

AD 710377

United States Naval Postgraduate School



THESIS

THE MODELING OF HUMAN INTELLIGENCE IN THE COMPUTER AS
DEMONSTRATED IN THE GAME OF DIPLOMAT

by

James Edward Collins

and

Thomas Dean Paulsen

June 1970

This document has been approved for public release and sale; its distribution is unlimited.

Reproduced by the
CLEARINGHOUSE
for Federal Scientific & Technical
Information Springfield Va. 22151

The Modeling of Human Intelligence in the Computer

As Demonstrated in the Game of DIPLOMAT

by

James Edward Collins
Lieutenant Commander, United States Navy
B. S., United States Naval Academy, 1959

and

Thomas Dean Paulsen
Lieutenant Commander, United States Navy
B. S., United States Naval Academy, 1960

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL
June 1970

Author

James E. Collins

Author

Thomas D. Paulsen

Approved by:

Robert C. Bolles

Thesis Advisor

Brooks J. Lockhart

Chairman, Committee for Computer Science

R. F. Rinehart

Academic Dean

ABSTRACT

The purpose of this thesis is a discussion of developing human-like behavior in the computer. A theory of the human learning processes is first described. This leads to the presentation of a computer game which simulates the human capabilities of reasoning and learning. The program is required to make intelligent decisions based on past experiences and critical analysis of the present situation.

TABLE OF CONTENTS

I.	INTRODUCTION	9
II.	THEORY	10
	A. THE LEARNING PROCESS OF THE HUMAN MIND	11
	B. REPRESENTATION OF THE HUMAN LEARNING PROCESS	13
	C. BACKGROUND OF THE GAME OF DIPLOMAT	14
III.	THE GAME	20
	A. THE REASONING PROCESSES	28
	1. Subroutine STRTGY	28
	2. Subroutine CPNDCN	34
	B. MEMORY STRUCTURE AND THE LEARNING PROCESS	37
	1. The Memory Structure	37
	2. The Learning Process	40
	C. ANALYSIS OF THE GAME	44
IV.	CONCLUSIONS	46
	A. EXTENSIONS AND CHANGES	46
	B. POSSIBLE USES OF DIPLOMAT	48
	APPENDIX A - THE MEMORY STRUCTURE OF DIPLOMAT	50
	APPENDIX B - EXAMPLE COMPUTER TERMINAL OUTPUT	52
	APPENDIX C - COMPUTER PROGRAM LISTING	74
	BIBLIOGRAPHY	160
	INITIAL DISTRIBUTION LIST	162
	FORM DD 1473	163

LIST OF TABLES

TABLE

- I. Wealth Changes Versus Strategies
As a Function of Player Strength 21

LIST OF FIGURES

FIGURE

1.	Macro-Flowchart of the Game DIPLOMAT.	24
2.	Approximate Payoff Matrix Presented to Player (SOUTH).	25
3.	Macro-Flowchart of Subroutine STRTGY Indicating the Determination of the Desired Function	29
4.	Basic Algorithm of STRTGY	30
5.	Basic Algorithm of CPNDCN	35
6.	Category Values of Strategies Used in Pattern-Matching	42
7.	Prediction of the Next Move Based on Strategy Type	43

I. INTRODUCTION

In the past few years, much attention in the computer world has been given to the study and development of Artificial Intelligence. One goal of artificial intelligence is to develop human-like qualities in the computer. One method of patterning human behavior is achieved by giving the computer the ability to (1) absorb data, (2) use inductive reasoning to make generalizations based on the data and abstract information from the data, (3) make decisions based on these abstractions and generalizations, and (4) learn from past experience. Admittedly this seems to be an enormous task, and in fact it is, yet this is how a child learns. To go one step further in modeling human behavior, and at the same time provide a reasonable limit on the size of the computer required, the ability to "forget" long-past experience could be developed in the system.

The theory of the human learning process is discussed first in this thesis. The theory leads to the development of a computer program that learns based on data gathered from past experiences. The program uses its library of stored knowledge in making decisions. The example which is presented is a game named DIPLOMAT which simulates the representatives of two nations interacting in strategic negotiations regarding relative strengths, i.e., armaments. The game is played between two opponents, the computer and a person.

The authors of the thesis consider that the interaction of the computer and a human in a decision-making environment is an excellent way of demonstrating the ability of the computer to develop human-like behavior.

II. THEORY

One of the objectives of artificial intelligence is to build a computer system that will effectively mimic the human learning process. This would be an enormous task if the system were to be so general in nature that it could handle any conceivable task, because it would involve representing the external world in a form adaptable for the computer. When restricted to a small subset of the world, i.e., confined to one or two specific tasks, attempts at this objective have been fairly successful in that the systems do mimic and sometimes surpass the human in their performance in these areas. However when trying to model the human brain in the general sense, the problem has been found to be extremely complex.

A possible approach to the problem would be to model the human brain using the physiological approach, i.e., to build the exact electrical network of sensors, storages, and connectors that form the physical makeup of the brain. Obviously this is impractical because the brain is composed of so many cells. Dr. R. L. Beurle, a noted English authority on artificial intelligence [Ref. 3] who has extensively studied the theory of brain models, estimates that the brain is composed of approximately 10^{10} neurons (nerve cells).

Another approach to the problem might be the psychological approach. In this approach, it is necessary to model the logical structure of the brain rather than the physical structure. However in order to use this approach, some knowledge of the process of psychological development of human behavior must be obtained. It would be best to trace the learning processes from their beginning in a child, since a knowledge of

total human behavior usually requires almost a lifetime of study and experience.

In the area of the development of the mental abilities of the child, the works of one man stand out above all others - the works of Jean Piaget. Jean Piaget is a noted Swiss child psychologist who was educated at Evychotel, Zurich, and the University of Paris. He has been a professor of child psychology and history of scientific thought at the University of Geneva since 1929, and is the director of the International Bureau of Education at the Institute J. J. Rousseau. Above all, he is noted for his research in the development of the mind from birth to adolescence.

The works of Jean Piaget, as presented in References 9 through 11, form the basis of this section on the theory of the development of the human learning process. They are supported by References 15 and 16, which are basic references used in the American medical profession.

A. THE LEARNING PROCESS OF THE HUMAN MIND

Piaget contends that the human mind consists of a finite number of structures, each consisting of a finite number of cells. Each of these cells contains an element of information which is a part of the human thought process. He theorizes that there is a separate set of information structures for each function of the central nervous system which is composed of the brain and spinal cord. Many of these separate sets of structures are developed before birth, for example, a set controlling the action of the heart, another the function of breathing, a third controlling the flexing of the arms and legs. Piaget further contends that as relationships build, these structures are linked together to produce automatic reflexes and in general the structures are reordered

and enlarged in successive phases. The central theme for the human learning process is then the process of manipulation of the information structures, i.e., the building, storing, linking, rebuilding and re-linking of elements of information.

It is easiest to describe the manipulating of these information structures by studying the development of the brain of the newborn baby. This is the approach presented by Piaget. At birth the cerebral cortex, which is composed principally of neurons, is largely undeveloped, and the infant is basically a reflex organism. The reflexes are the result of the manipulation of the information structures before birth. Other than the regular actions of the heart and other vital systems, most muscle activity is random, lacking direction from the brain.

One of the first learning processes of the infant is the learning of spatial relationships. While flexing his arms and legs, he touches the side of the crib. Relationships are built in his mind between the information structures for the sense of touch and those of muscle control, and a new structure built up regarding an awareness of the confining walls of the crib. These structures become libraries of information. Other libraries build up, for example he grows to associate his mother with warmth, comfort, and nourishment. The building of relationships between the structures, and the reordering of the structures and links, forms an associative memory within the brain.

It is important in the study of the human brain to understand that these activities are the result of relationships between the libraries even though the information in their structures is dissimilar. Once the links are established, the dissimilar sets function together. The act of crying which is the result of combining the structures of muscle control,

an awareness of a specific need, respiration control, and others is a good example. The operation that is accomplished constitutes some action of uniting or separating, placing or displacing, arranging or disarranging elements of information into sets of structures. As the relationships develop within the brain, the mind learns how to use the data which has been structured within its memory to produce a desired result, such as rolling over or sitting up.

B. REPRESENTATION OF THE HUMAN LEARNING PROCESS

The process of thought is the result of the human becoming aware of the relationships that link cells or structures in the associative memory of his brain. Piaget explains the mechanism of thought as a movement which evolves when an awareness of the relationships becomes sufficiently advanced to permit the individual to combine the information from several structures into a single idea. The resulting thought may cause other reactions such as body movement, and thus may cause new structures to be created or new links to be built. The actual process involved in the human brain in conceiving a thought is not definitely known, and must be extremely complicated. If such a scheme is to be computerized, however, it must be made deterministic.

In an attempt to make the process a deterministic one, it is necessary to summarize some facts deduced from the discussion of Piaget's works. One result of the thought process is the acquiring of an idea, which is the outcome of the interaction of cells or structures. The combination of old structures in a new way may lead to new and possibly improved concepts. The new relationships may be formed by applying deductive reasoning to some information extracted from the interconnected structures. In this way, the learning of a new concept may be achieved.

Thus learning is not merely an additive process, i.e., the piling of one disjoint piece of information atop another and another and another.

The number of cells or structures is not the criteria for learning, rather it is the effective combining of the stored information that results in intelligence. Once a certain point is reached in the process of human development, the physical size of the brain does not rapidly become larger and larger, rather information is restructured, old information may be forgotten and new information stored in its place, and new links established.

These two techniques, structuring of memory and learning by interaction of the structured information, are utilized in the structuring of a computer memory and the development of a program which will exhibit deductive reasoning based on learning. This is the example presented in this thesis. Considerable research and soul searching went into formulating a worthwhile application for the model of human intelligence. The application had to be general enough to apply to real-life situations, yet not overly complicated.

C. BACKGROUND OF THE GAME OF DIPLOMAT

Several recent periodicals have been devoting considerable attention to games as an application of artificial intelligence. One game in particular has been the subject of much of the attention, the game known as the Prisoner's Dilemma. The classic prisoner's dilemma is described in reference 7 as follows:

"Two suspects are taken into custody and separated. The district attorney is certain that they are guilty of a specific crime, but he does not have adequate evidence to convict them at a trial. He points out to each prisoner that each has two alternatives: to confess to the crime the police are sure they have done, or not to confess. If they both do not confess, then

the district attorney states he will book them on some minor trumped-up charge such as petty larceny and illegal possession of a weapon, and they will both receive minor punishment; if they both confess they will be prosecuted, but he will recommend less than the most severe sentence; but if one confesses and the other does not, then the confessor will receive lenient treatment for turning state's evidence, whereas the latter will get 'the book' slapped at him."

One reason the prisoner's dilemma has been discussed so much is that there are numerous situations in the world that have some of the characteristics of this game (or extensions from this game such as the addition of participants and/or strategies). Most economic situations that require a choice among a finite number of strategies have these characteristics. Consider, for example, gasoline service stations located close to one another, each of which can lower its prices. Regardless of the price one's competitors set, any one manager is better off, in the short run at least, cutting his price. If all cut prices, however, the total volume of business is the same as if none cut prices, but the total revenue is less. On a larger scale, consider wheat farmers in a country without governmental price and production controls. Any one farmer is better off producing wheat as long as his marginal cost is not greater than the price. He will be able to sell all he can produce at the going market rate without affecting the price. If all farmers produce maximum amounts, however, the price will be pushed down and all will be worse off than if each had restricted his production. On the worldwide scale, there is the problem of disarmament. One country can be more powerful (or secure) by arming, but nothing is gained if all arm. All countries would be better off if all disarmed in that the money not spent on defense could be spent for, say, consumer goods or for correcting social ills.

More interesting than the one-time classic prisoner's dilemma is the iterated game, i.e., a game composed of many moves. In the overall

picture of the iterated game, each player must (in general) forsake the possibility of maximizing his own short-run profit to enjoy the greatest payoff by maximizing his long-run profit. With a one-trial game and an unknown rival, it is difficult to imagine the wisdom of choosing a medium-gain, little-risk, cooperating type strategy, when more can be gained (or lost) by choosing the high-risk high-payoff strategy (nothing ventured, nothing gained). The single trial situation eliminates both the possibility of future cooperation and the possibility of punishing a rival for non-cooperative action in one trial. Dr. Lester B. Lave of the Carnegie Institute of Technology [Ref. 6] has studied factors affecting cooperation in prisoner's dilemma type games. He has found that the single-trial game and the multi-trial game are basically equivalent in the formal sense in that the expected values of the two games have the same range across different groups of opponents. However, the games are quite different with respect to negotiating cooperation among different participants. The expected values of the two games are not equal for a given rival, since certain forms of behavior can induce cooperation or competition. He based these results on experiments conducted using human competitors only, and did not introduce computer gaming into his research. He further found through experimentation that when a game was iterated, it was possible to display behavior that induces or stifles cooperation. He also found that it was possible for the players to develop communication between them using the choices in the game, but that this rudimentary form of communication took time to establish and function. He found that the longer the game, the more likely it was that a stable cooperative solution could be achieved.

Other experiments with a complex decision task showed that experience from previous tasks was a large factor in success. The conclusions

drawn from studies conducted at the Human Performance Center, Ohio State University, [Ref. 13] were that after gaining experience when tested under realistic circumstances, the experienced subjects were in general less conservative than naive subjects who received no such prior training. They were willing to take more risks to achieve higher overall gains.

The study of concept attainment by the machine has also been the subject of study. In order for a human to play a game of this nature, he must be able to form a few concepts of the game itself and of his opponent. It can be likened to the game of poker in that opponents must deduce the type of individuals playing the game. Dr. Frank B. Baker, a professor of educational psychology at the University of Wisconsin, has studied the theory of concept attainment and developed a computer program which demonstrated the theory in a simple decision task requiring the identification of common attributes among different sets [Ref. 1]. In Reference 2, Dr. Baker states:

"If computer programs are to serve as useful models of cognitive behavior, their creators cannot avoid coming to grips with the necessity for establishing an internal organization for their model which implements the higher level cognitive behavior associated with the human capacity for self-direction, autocriticism, and adaptation."

In computer game playing, the concept of the game itself is built into the game, however the concept of different methods of play or types of strategies that the computer may face is something that must be attained as it plays.

Most of the behavioral tests conducted to date have been between humans. However, in the past few years, more attention has been devoted to simulating these tests on the computer using interaction between man and the machine. Professor Roman J. Weil of the University of Chicago stated in Reference 18 the advantage of using the computer for this application quite well:

"The philosophy underlying the computer approach is this: If a program can be constructed that, when placed in a prisoner's dilemma situation, exhibits behavior like the behavior of people when placed in the same situation, then that program will be a powerful tool for generalizations."

Professor Weil goes on to say that if the computer can be made to simulate the human in organizing data and making decisions under all simulated conditions of risk and stress, it will be possible to accumulate more data and more accurately predict human behavior in the same environment.

The game of DIPLOMAT presented in this thesis is basically an extended version of the prisoner's dilemma. It incorporates an iterated game technique with the game lasting anywhere from ten to fifty moves, and two opponents choosing from among three strategies. Additional complications to the prisoner's dilemma basis are inserted by varying the payoffs to the participants as a function of previous moves, and by inserting an unknown random variance into the payoff table.

In order to be successful in this game, the participants must perform most of the tasks listed in the introduction as a goal of artificial intelligence, in addition to performing the task of concept attainment. In particular, the computer must analyze the situation of the game at the time of the move and refer to past games and past moves in the present game to determine its opponent's probable strategy. It must then abstract enough information from his prior and present knowledge to select a strategy, and correctly analyze the results of the move in order to store (remember) meaningful experience. In addition, it must form concepts regarding the reliability or honesty of its opponents. The participants in this game may or may not be completely truthful in their

negotiations, which is certainly characteristic of actual diplomats at the conference table. The computer, then, must form estimates of its opponent's reliability and factor this into its selection of a strategy.

In the first game played, the computer has no prior knowledge upon which to draw, and so must reason from an analysis of the present situation. With each move however, the system acquires more experience and thus has a better base from which to draw in selecting strategies. When the computer plays its second game, it has the experience of the first, with one winning and one losing strategy, to use for reference. In general, the more games it has played, the more experienced it is and the better it performs in making important decisions.

III. THE GAME

DIPLOMAT models the representatives of two nations interacting in strategic negotiations regarding armaments. It is basically a non-zero-sum two person rectangular game, using the phraseology of formal game theory. Each nation is given three choices of possible strategies regarding armaments:

- Strategy 1: Increase Armaments (Arm)
- Strategy 2: Maintain the Status Quo
- Strategy 3: Decrease Armaments (Disarm)

The following basic concepts govern the decision of the strategy to be followed by each side:

Each nation starts the game with zero strength and zero wealth, where "zero" implies a deviation from the average rather than absolute zero.

Arming increases strength by one unit, disarming decreases strength by one unit, and maintaining the status quo does not change strength. Arming costs money decreasing wealth, disarming gains wealth, and maintaining the status quo may increase or decrease wealth, depending on the strength of the player: If the player is strong in armaments, it will cost him more for upkeep and maintenance and hence decrease wealth; if the player is weak, armament upkeep is low and maintaining the status quo should gain some wealth. The basic changes in wealth for the different strategies are shown as a function of strength of the player in Table I.

Strength	STRATEGY		
	1	2	3
$S < -6$	0	3	3
$-6 \leq S < -4$	0	2	3
$-4 \leq S \leq -2$	-1	2	3
$-2 < S < +2$	-2	1	4
$+2 \leq S \leq +4$	-3	0	3
$+4 < S \leq +6$	-4	-2	3
$S > +6$	-6	-3	2

Table I
Wealth Changes Versus Strategies As
a Function of Player Strength

Changes to the basic values in Table I are generated by a random integer amount between the values of -2 and +2. These changes, which are to simulate economic conditions, are generated at random times during the game; hence, a given economic condition may last for only one move or for many moves. The economic condition for each player may be different, as each opponent uses a different random number generated from the same random number seed. The economic condition in effect for each player is not furnished to either opponent, but must be estimated based on the results of each move. The economic conditions of -2 and -1 mimic those times when prices are high, and the conditions of +1 and +2 mimic those times when the cost of living is relatively low. These values are added to the basic wealth change values of Table I to determine the actual wealth changes for the strategies chosen.

Point values for each participant are determined after each move according to the formula:

$$\text{POINTS} = (\text{Previous}) + 2 \times (\text{Wealth}) + 5 \times (\text{Strength})$$

Points Change Change

Bonus points are given for the following combinations of strategies:

If both opponents cooperate in disarming, each receives two bonus points as a reward for their cooperation.

If both opponents arm, each receives -1 bonus point, because both have spent money without acquiring any relative advantage.

If one side arms and the other disarms, the opponent who arms is awarded four bonus points for "outfoxing" the other.

The length of the game is at least ten and at most fifty moves. Between these values, a random selection is used for the decision to end the game; as the number of moves increases, the greater the chance of random termination. Experience has indicated that the average length of the game is twenty moves.

The winner of the game is decided by one of three methods:

Normal Termination: The participant with the most POINTS at the end of the game is declared the winner.

Abnormal Termination: The game may be abnormally terminated, even before ten moves have been completed, in two ways. The game is stopped if one nation's wealth becomes thirty units greater than the other, and the richer nation is declared the winner. Similarly, if one participant becomes stronger in armaments than the other by ten units, the game is stopped and the stronger nation is declared the winner.

In playing the game of DIPLOMAT, each side initially declares a proposed strategy called that participant's Concession Point, with each side taking turns declaring the first concession point. The proposal of each player is used as an aid in deciding that player's probable actual strategy. After both concession points have been declared, the computer's

move is locked into the system and the human participant is asked to declare his final strategy for that move. Of course, the concession point and the strategy need not be the same, but wisdom must be used in selecting proposals versus actual strategies in order to maintain a high degree of reliability, yet keep the opponent off guard as to the actual strategy to be chosen.

The program is written as a main routine which controls the running of the game itself, and forty-one subroutines. Seven of the subroutines assist in controlling the game and in performing list-processing chores, nine of the subroutines assist the program in accomplishing its reasoning capabilities, and the other twenty-five subroutines are necessary for accomplishing the task of learning. Among the tasks performed by these forty-one subroutines are setting up the memory, saving and restoring experience, selecting a strategy and a concession point, determining the opponent's reliability, pattern matching data from previous games and moves to take advantage of prior experience, and so forth. The program is written in the FORTRAN IV language, and is designed to operate on-line on a computer terminal using the Cambridge Monitor System. A complete listing of the program is contained in Appendix C.

Figure 1 is a macro-flowchart of the game. DIPLOMAT progresses in the following manner:

After initializing counters, point values, and payoff tables, the system sets up the memory cells in an associative memory structure, filling in the data from previous games. The initial economic conditions are also determined.

The player (hereafter called SOUTH) signs into the system with his name. The computer program (hereafter called NORTH) then

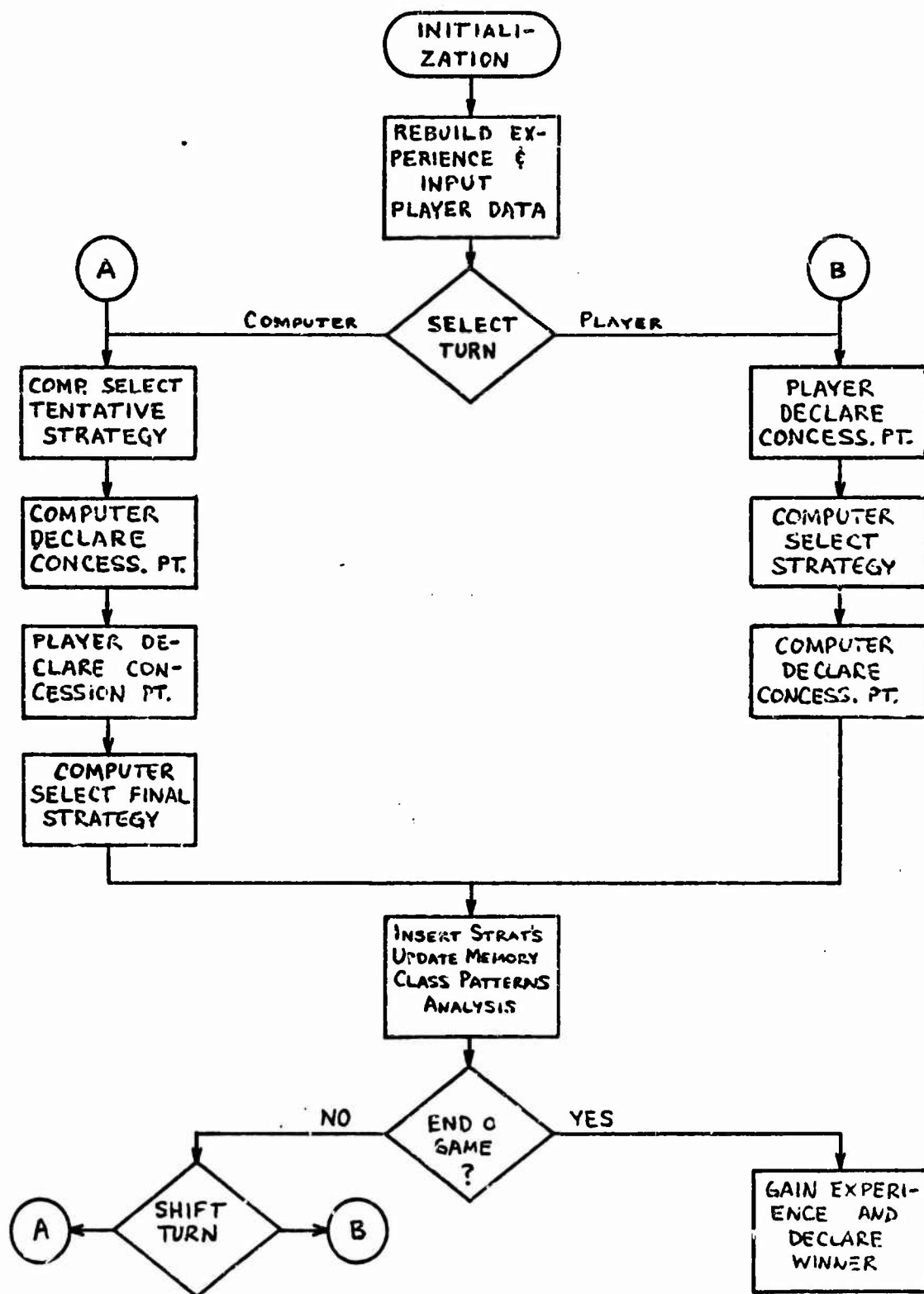


Figure 1. Macro Flowchart of the Game DIPLOMAT

checks to see if he has played this opponent before, and if so brings his history to the top of the catalog of players.

The player inputs a random integer number for use as a seed in generating random numbers throughout the program. The computer simulates flipping a coin to see who will make the first proposal during the first move. SOUTH calls the flip of the coin. (In the discussion which follows, it will be assumed that SOUTH won the toss).

Having won the toss of the coin, SOUTH must make the first concession point. To aid him in making his selection of a proposal and ultimately of a strategy, a payoff matrix is presented to him showing the approximate payoffs to SOUTH (relative to NORTH) for the various combinations of strategies. Approximate payoffs are shown rather than actual because neither SOUTH nor NORTH have access to the existing economic conditions. A sample payoff matrix as presented to SOUTH is shown in Figure 2.

STRATEGY		SOUTH		
		1	2	3

N	1	* -1	1	2
O		*****		
R	2	* 1	0	1
T		*****		
H	3	* 2	-1	2

Figure 2

Approximate Payoff Matrix Presented to Player (SOUTH)

The approximate payoff matrix changes according to the strengths of NORTH and SOUTH because of the differences in wealth for the various strategies as shown in Table I.

SOUTH may propose a concession point of 1 (arming), 2 (maintaining the status quo), or 3 (disarming). NORTH then considers SOUTH's concession point, attempts to determine if SOUTH is being honest, analyzes his payoff matrix, and responds to SOUTH with his concession point. NORTH also decides upon a strategy at this time.

SOUTH considers NORTH's concession point and decides upon a strategy for the move. Both sides then enter their strategies into the system and the results are tabulated and presented for analysis by both participants.

For the next move, NORTH will declare his concession point first. The game proceeds in this manner, alternating between NORTH and SOUTH as to who is first to declare a concession point. After each move, each side analyzes the results in order to determine the economic conditions in effect. After ten moves have been completed and after each move thereafter, a random number is generated and tested to determine if the game should end.

At the end of each move, changes to the payoff tables generated as a result of changing strengths of the participants are calculated and inserted into the system, and it is determined if it is time to change the economic conditions. If so, they are calculated and inserted into the game.

A sample output of the program as exhibited at the counter terminal is located in Appendix B.

In playing this game, the computer program maintains data regarding past moves and past games in order to draw upon this experience in selecting strategies and concession points in future moves and in future

games. It thus forms concepts of each player as it proceeds. At the end of each move, some of the learning subroutines are called upon to update the short-term memory in order to store data for use in playing the game in progress and for maintaining running totals. At the end of the game, others of these subroutines calculate the game totals and determine the characteristics exhibited by both participants for inclusion in the long-term memory. This is the experience gained by the program from this game.

The data which forms the experience is kept in an associative memory structure mainly for ease of manipulation, but it is considered that this patterns the human in organizing data and experiences in his mind. In studies conducted of neural nets, it has been found that the human will take data, organize it into logical structures based on determining relationships between units of the data, and store it in an associative net accordingly [Ref.s 2 and 3].

Diagrams of the memory structure of DIPLOMAT are contained in Appendix A, and were conceived by the authors after critical analysis of the structure of data maintained about players during hand simulation of the game.

Besides exhibiting the human attribute of learning by storing away past experiences, the program mimics the human in its reasoning capability in deciding upon strategies and concession points. The method of reasoning was also patterned after analysis of the mental reasoning used during hand simulation of the game. These two attributes of reasoning and learning are discussed in greater detail in the sections which follow.

A. THE REASONING PROCESSES

Two of the subroutines of the program are designed to mimic the reasoning processes of the human. These are the strategy decision subroutine (STRTGY) and the concession point decision subroutine (CPNDON). These subroutines in turn call on many other subroutines to determine optimum strategies, probable moves of the opponent, next moves of the computer, and so forth. Both the strategy and concession point decision subroutines were written based on the thought processes used by the authors in playing the game by hand.

1. Subroutine STRTGY

This subroutine has three main functions. The first is that of strategy decision based on an analysis of all the factors available. This decision is final if the player's concession point is known, otherwise it is a tentative decision until the declaration of SOUTH's concession point. The second function is that of reconsideration of the tentative strategy after SOUTH has declared his concession point, and results in the final selection of a strategy. This function is called upon only if NORTH was first to declare a concession point. The third function is analysis of the completed move in order to determine if a better choice of strategy could have been made. If so, an adjustment of the calculations performed in the decision portion of the routine is accomplished.

Macro-flowcharts of STRTGY are shown in Figures 3 and 4. The subroutine may be called upon either two or three times each move, depending upon who submits the first proposal. A series of flags are used to determine for which function the subroutine is being called. The algorithm for making this determination is shown in Figure 3.

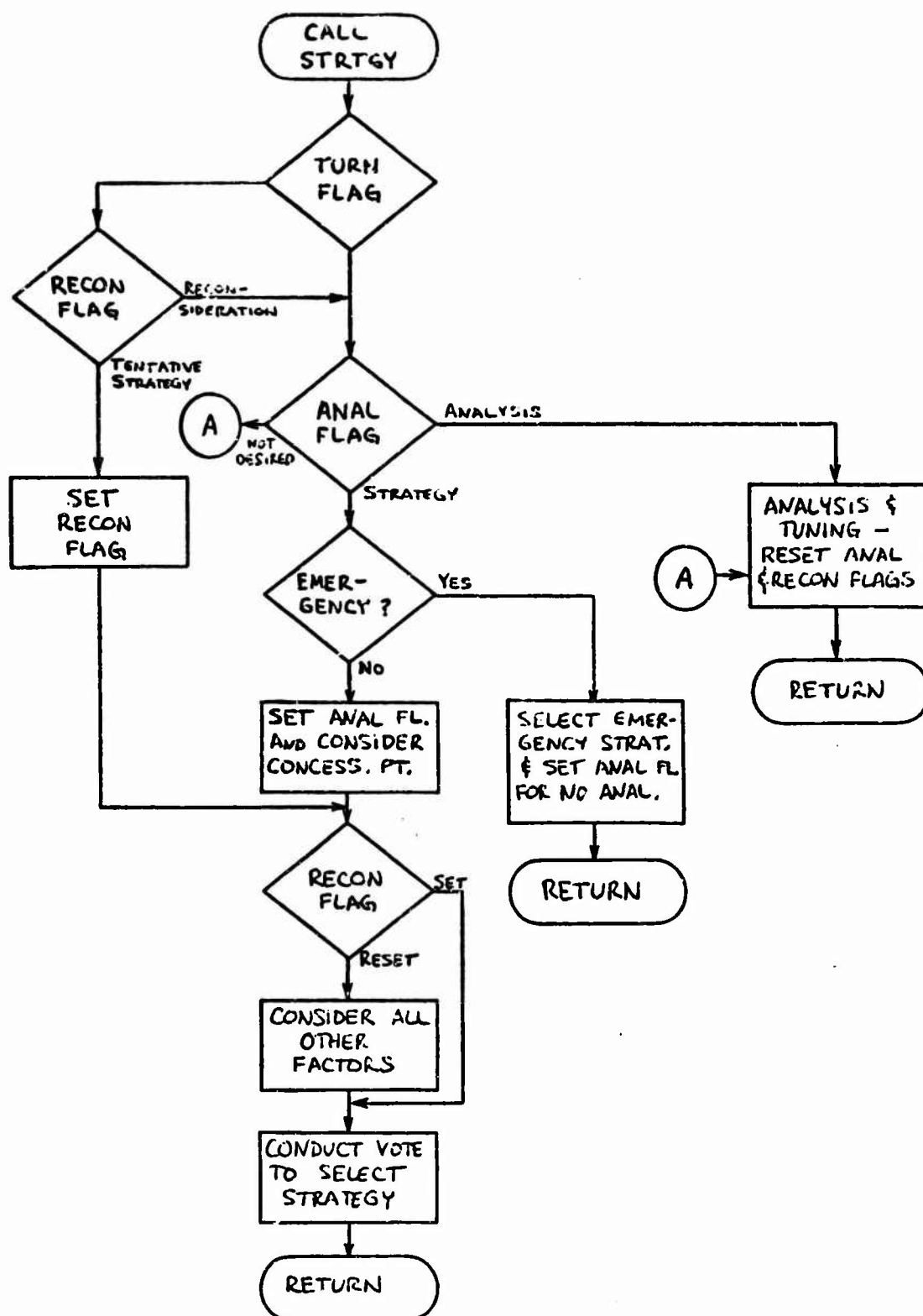


Figure 3. Macro Flowchart of Subroutine STRTG' Indicating Determination of the Desired Function

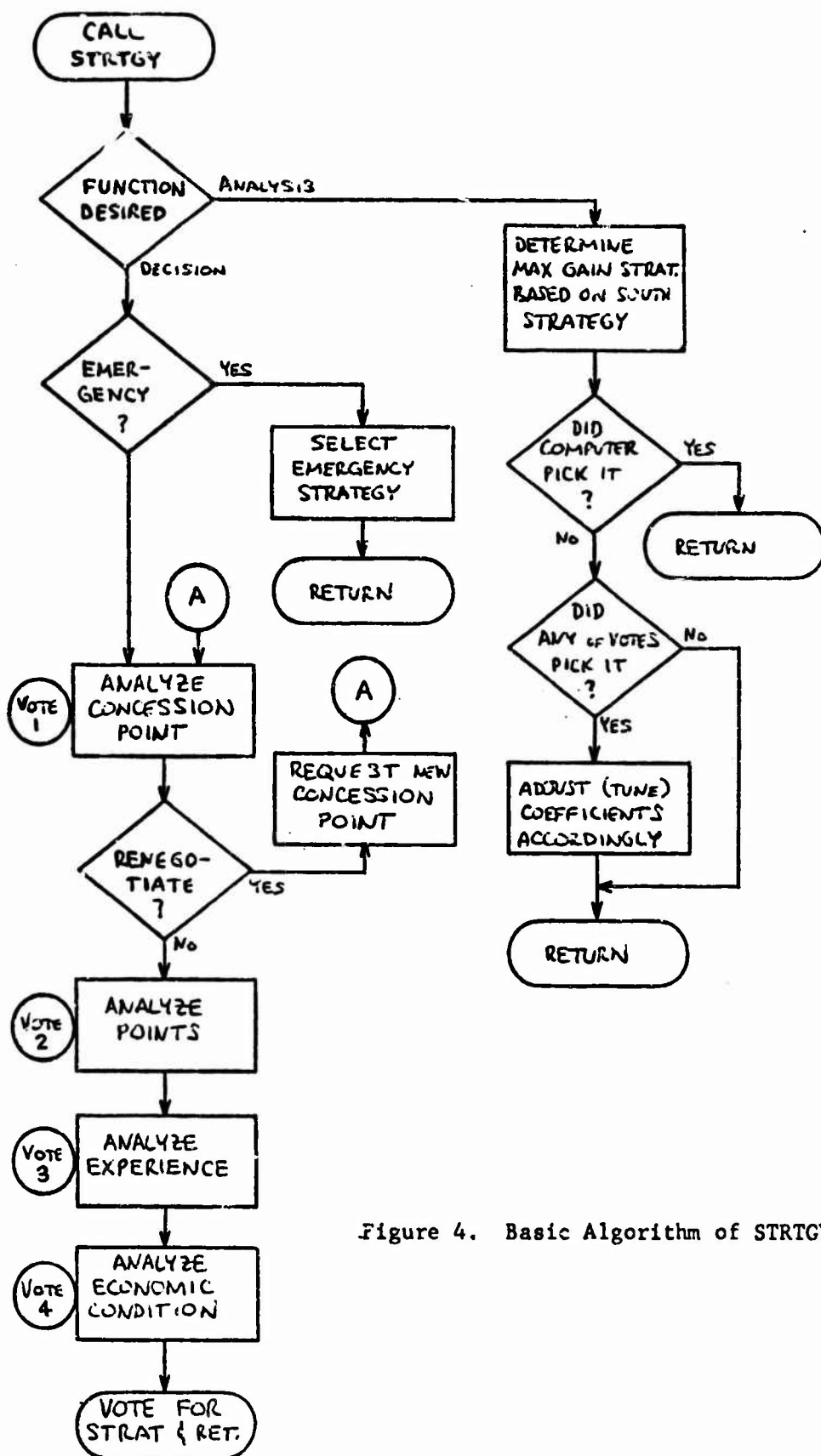


Figure 4. Basic Algorithm of STRIGY

Figure 4 provides the basic algorithm of STRTGY. The decision upon a strategy for the move is based upon the consideration of four factors, each of which are weighted in a "polynomial" fashion. These four terms of the polynomial are based upon:

- 1) SOUTH's concession point,
- 2) POINTS relative to SOUTH,
- 3) Previous experience, and
- 4) Economic conditions.

The coefficients assigned to each of these terms are inserted into the system at the start of the game, and may become modified as the game progresses, as discussed in the description of the analysis function.

Even before considering any of the four factors, the computer checks to ensure that it is not in danger of losing because of relative wealth or strength disadvantage. If it is, the system immediately selects as its strategy the one which will gain the most wealth or strength, as needed, and returns to the main program without considering any of the four factors. If the computer finds that this emergency action is necessary, later analysis of the move is not performed.

If emergency action is not necessary, each of the four factors are considered and used in determining a final strategy for the move.

The first factor to be considered is SOUTH's concession point. The computer first determines if it wants to call for a renegotiation. Renegotiation is requested if it finds that its opponent's proposal was to arm and that he is quite strong already; it requires SOUTH to submit a new concession point, although the new proposal may be the same as the original. The next step after considering (and possibly carrying out) renegotiation is to attempt to determine SOUTH's probable forthcoming strategy. The computer selects candidates for its opponent's

strategy based on weighted reliability estimates, patterns of dishonesty in SOUTH's proposals, history of previous games with this player, and formal game theory. After deciding upon SOUTH's probable strategy, the computer selects as the first term of the polynomial the strategy which maximizes NORTH's gain relative to SOUTH if SOUTH actually selects that strategy.

The second factor to be considered in selecting a strategy is based on NORTH's points relative to SOUTH. Depending upon whether the computer is behind, ahead, or even with its opponent, this term of the polynomial is set to the strategy which maximizes NORTH's possible gain or minimizes its possible loss. Formal game theory is used in selecting these possible strategies.

The third factor to be considered is the most difficult one, because it is based upon previous experience. If SOUTH has been a previous opponent (within the last ten opponents), there exists a history of his previous games in the computer's long term memory. NORTH can use this information in predicting SOUTH's probable moves. In addition, NORTH can search other strategy types contained in its libraries of past games in order to select previously successful strategies to use against its opponent. After several moves have been completed, the computer searches all the strategy listings in its "experience" in order to classify SOUTH's general pattern of strategies. Each of these listings may contain pointers to other strategy patterns which have proven successful against SOUTH's pattern in the past. NORTH can also pattern-match its own strategy pattern against those existing in the library in order to determine its own predicted move. The computer occasionally selects a

strategy opposite to the predicted one to avoid being stereotyped by SOUTH. This is called a "guess-opposite" selection of a strategy. It is the third factor of the polynomial which gains from the learning capabilities of the machine. The method of learning and pattern-matching is discussed in greater detail in a later section.

The fourth and final factor to be considered in selecting a strategy is based on an estimate of the economic condition in effect. After the completion of each move, the program analyzes the results to determine if they match the expected values. If not, it estimates the economic conditions and sets this term of the polynomial to the strategy which takes the most advantage of the state of the economy. For example, if prices are low, it is probably the best time to arm, but if prices are abnormally high it may be too expensive to arm at that time.

After all four factors have been considered in selecting a strategy, the system conducts a vote to determine the choice for the move. If SOUTH's concession point has not been declared, the coefficient of the first term is set to zero, and a tentative strategy is chosen based on the other three terms. When it becomes time for the reconsideration, the vote is taken of all four terms for deciding the final strategy. The vote is accomplished by summing the coefficients of the terms voting for each of the three possible strategies. The strategy which receives the greatest sum is the "winning" strategy; in case of a tie, the computer selects from the tying strategies the one which will result in the greatest absolute point gain.

Having selected a strategy, control is returned to the main program which either calls upon Subroutine CLDCN to decide NORTH's concession point, or else locks the strategy into the system awaiting SOUTH's indication of a strategy.

The third function of Subroutine STRTGY is that of analyzing the completed move in order to determine if a better strategy could have been chosen. The system does this by entering NORTH's relative payoff table with the actual strategy chosen by SOUTH to determine the strategy which provides the maximum relative gain. If the computer determines that the correct strategy was chosen, no adjustments are performed and control is returned to the main program. If, however, the best strategy was not picked by NORTH, the computer determines if any of the votes cast for the different strategies matched the best possible strategy. If any are found, the coefficients corresponding to those terms are then increased (tuned up), and those assigned to the wrong terms tuned down. If no votes are found, the analysis was unsuccessful for that move.

If the computer ultimately wins the game, the final values of the coefficients for each of the terms are inserted into the long-term memory of the player for use as the initial values in the succeeding game with that player. This is done because the system found these coefficients successful and they would probably provide a better base from which to start the next time. This is but another part of the learning process of the system.

Thus, Subroutine STRTGY attempts to determine a strategy to be followed in the game of DIPLOMAT in much the same way that a human reasons through the game. Similar to a human, if the reasoning process results in a wrong answer, the system attempts to improve itself.

2. Subroutine CPNDCN

This is the second of the program's subroutines which employs human-like reasoning. Its purpose is to determine a concession point to be proposed to SOUTH after having selected a tentative (or final) strategy. Figure 5 provides a macro-flowchart of this subroutine.

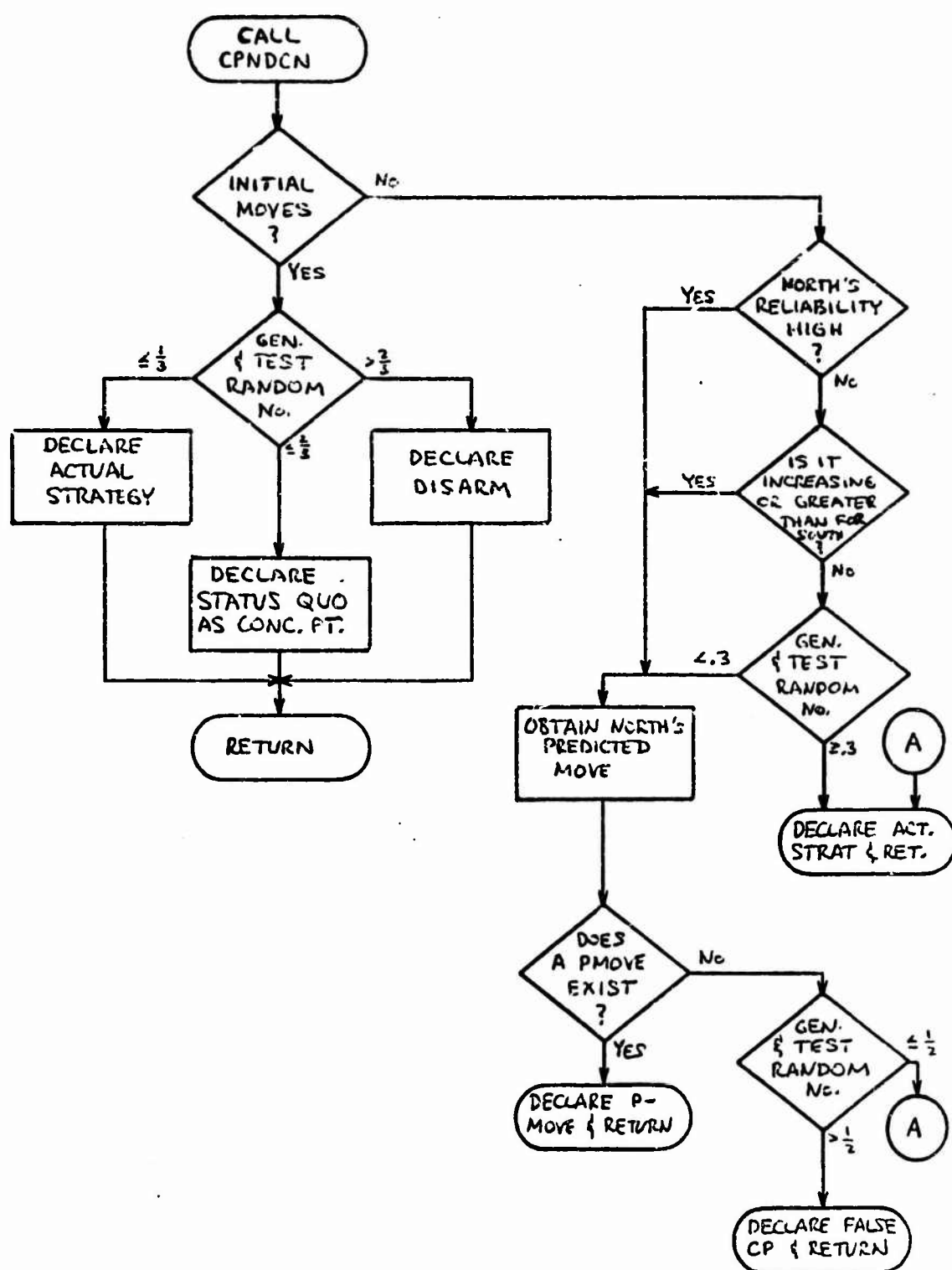


Figure 5. Basic Algorithm of CPNDCN

NORTH's concession point must be carefully chosen. In order to keep the player off guard, the computer must try to maintain at least a facade of reliability, yet not be so reliable as to be "read like a book." If NORTH and SOUTH are each following a pattern of moves, the computer should indicate that it expects to follow the pattern, even if it has no such intention.

For the initial moves, the computer uses a semi-random pattern of selecting proposals. Weight is given to choosing either the actual strategy proposed, or to choosing a passive (i.e., disarming or status quo) declaration. After the initial moves have been completed, however, the computer utilizes much of the data it has been accumulating on the player during the game. In addition to pattern-matching previous moves, the computer maintains estimates of the player's reliability as well as its own. EN is NORTH's estimate of SOUTH's honesty, and ES is what NORTH thinks SOUTH estimates for the computer's reliability. These are both weighted values, with the most weight given to the recent moves. EN and ES are calculated after each move by comparing the actual strategy with the concession point. The values of EN and ES drop accordingly each time there is not a match.

If ES is high, is increasing, or at least is greater than EN, the system determines its predicted move from pattern-matching its own strategies, and uses a predicted move as its concession point. If, on the other hand, NORTH's reliability needs improving, the subroutine normally selects the actual strategy determined by STRTGY as the computer's concession point.

As with STRTGY, CPNDCN was written based on hand simulation of the game. Notes were kept on the reasons behind each decision, and

used to help determine the methods of reasoning developed in these subroutines.

B. MEMORY STRUCTURE AND THE LEARNING PROCESS

The information within this section is somewhat more detailed because it describes the attempts to exemplify the concepts of Piaget in structuring the mind. Computer list-processing techniques are used in structuring and manipulating the memory for this task. The reason for this is that these techniques model the theoretical structures of the brain and are efficient in coding requirements.

1. The Memory Structure

The memory structure of the program, depicted in Appendix A, is divided into two segments, the "temporary" or short-term memory and the "permanent" or long-term memory. The short-term memory contains data regarding the current game and the long-term memory contains the data which comprises the experience gained from previous games.

The short-term memory is used to maintain data necessary for determining the pattern of play being used by both the human player and the computer. The information within this segment of memory grows as the game progresses. At the end of the game, pertinent data is summarized and transferred to the long-term memory, and the short-term memory is deleted or "forgotten." The temporary memory consists of arrays and lists for determining the reliability estimates used in the reasoning processes of the system. In addition, this segment of memory contains a pattern of strategies used by each participant. It also maintains a total of the number of times each strategy is selected. The former is used in generally classifying the method of play which is being employed,

and the latter is used in determining the overall aggressiveness being exhibited.

The long-term memory, the "experience," consists of arrays and lists arranged into four libraries. Libraries are interconnected as they are developed, in order to maintain continuity in the flow of information and facilitate access to specific data. These four libraries are labeled the "Catalogue of Players," the "Library of Initial Moves," the "Library of Types of Moves," and the "Library of Sequences of Moves." For simplicity these names are abbreviated the "Catalogue," the "Initial Library," the "Type Library," and the "Sequence Library."

To provide a basic understanding of the construction and maintenance of these libraries, it is necessary to define various terms associated with them. A move is a played strategy. Groups of three moves are categorized into five levels of aggressiveness. Groups are thus combinations of any three strategies played. The following example illustrates the construction of groups:

Suppose an opponent picked the following moves (strategies):

1 2 1 1 3 2 2 1 3 3 1 1 1 2 1 3

Group (1) then consists of 1 2 1, Group (2) is 2 1 1, Group (3) is 1 1 3, Group (4) is 1 3 2, and so on.

Patterns are determined in the Initial Library according to individual moves, and are determined in the Type Library by a sequence of groups. The sequence of groups is called a "TYPE." Further discussion will be given to the utilization of categories of aggressiveness, and their associated groups, in the explanation of the learning process of the program.

The Catalogue contains information on one to ten players. As a player is first introduced into the game, a unique set of nodes (sequential set of computer words) is established for him which at the end of the game will be filled with data and pointers regarding his history of play. The most recent player is maintained at the top of the catalogue and the oldest player at the bottom. As players participate in additional games, a reordering of the Catalogue is accomplished to maintain this perspective. If the Catalogue reaches its capacity of ten players and a new player is introduced into the system, the information on the oldest player is deleted (forgotten) and the new player established in the Catalogue.

Information maintained on each player includes pointers to the initial strategies and a sequence of "TYPE" patterns used by both the player and the computer in previous games. It also contains the set of parameters (the foundation of the coefficients considered in the strategy polynomial) which have proven most successful against that player, the average values of reliability demonstrated by both the player and the computer when playing against that player, and the player's aggressiveness in previous games.

The INIT Library contains the initial three moves (first group) used in various games in the past, and pointers to the Type Library. Information on the exact initial strategies used in previous games is maintained so that the computer can try to get a "jump" on the player at the beginning of the game. It is used to predict SOUTH's initial strategy and to aid the computer in selecting the best offense against this strategy.

The Type Library contains up to ten structures. A structure consists of a header cell followed by a TYPE of eight to ten groups. The header cell contains pointers to other structures in the Type Library which have demonstrated a successful defense against this TYPE, and against which this TYPE has proven to be a good offense.

The Sequence Library is a history of the TYPEs and associated initial strategies for each game played by a particular player in the Catalogue. It also contains the computer's TYPEs used against any player in the Catalogue. Each entry in the Sequence Library contains a pointer to a structure in the Type Library and a pointer to the initial strategy associated with that structure.

When the capacity of each of these libraries is filled, the oldest information is deleted, or updated, to make room for the new information. This conforms to the contention that the brain grows to a finite size, then information becomes restructured or forgotten as it grows out of date.

2. The Learning Process

The learning process progresses as the memory structures develop. During the first game the computer plays, there is nothing in long-term memory so the computer must reason through the game as best it can. During succeeding games, however, the computer searches long-term memory for strategy predictions and pattern-matching types of strategies.

When the player signs into the system at the start of a game, the computer searches the Catalogue to determine if this player has been played before. If so, the computer assumes that he will follow

the same initial strategy and determines the best moves against that strategy. After two moves the computer pattern-matches these against the INIT Library to find the closest match. The computer's third move is selected as the one which is best against the third move listed in the closest matched INIT group. After the third move, the computer again pattern-matches the initial three moves against the INIT Library, finds the closest match, and predicts that the player will use the TYPE to which that entry in the INIT Library points.

The Type Library provides general categories of strategies to simplify pattern-matching. In a twenty-move game, there could be an almost infinite number of sequences of exact strategies used, but by categorizing exact strategies into general classifications, pattern-matching may be done at a meta-level. These categories, as stated earlier, are determined by arranging the moves into groups of three and classifying the group into one of five levels of aggressiveness. Figures 6 and 7 illustrate the techniques of categorizing strategies and making predictions based on the types obtained. The overlap of the groups provides continuity in classifying and predicting.

As the game progresses, the computer pattern-matches the TYPES used by both the player and the computer, which are maintained in short-term memory, against the Type Library. This provides NORTH with predicted moves for both participants, and also with an indication of the TYPE that is best to use against SOUTH's TYPE.

The Sequence Library provides the computer with the ability to make better predictions of the TYPES that a player will use. After each game with a player, a pointer to the TYPE employed in that game is

CATEGORY	1	A1	2	A2	3
Combination of Groups of Three Moves	111	113	123	133	332
	112	131	132	331	323
	121	311	213	313	233
	211	122	231	322	333
		212	312	232	
		221	321	223	
			222		

The sum of the strategies for each category is:

Category 1 - 3 or 4

Category A1 - 5

Category 2 - 6

Category A2 - 7

Category 3 - 8 or 9

Figure 6. Category Values of Strategies Used in
Pattern-Matching

By totaling the first two moves of any predicted category, it is possible to predict the next move:

	If First Two Moves Total:	For Category Number:	The Move Predicted Is:
3 2 1 2 1 1 3	2	1	2
	3	1	1
2 1 A1	2	A1	3
	3	A1	2
Overlapping of Category Groups Shown Above	4	A1	1
	3	2	3
	4	2	2
Example Strategies and Categories are Inserted	5	2	1
	4	A2	3
	5	A2	2
	6	A2	1
	5	3	3
	6	3	2

Figure 7. Prediction of the Next Move Based on Strategy Type

inserted in the Sequence Library. The computer can pattern-match these lists to determine the expected type which the player will use during the next game.

At the end of a game, corresponding data between the short-term memory and the long-term memory are compared. This is accomplished by the same methods of pattern-matching used during the game. If similarity exists, the structures of the short term memory are combined with the closest Initial and Type Library structures to form more up-to-date information. If no similarities exist, the information from temporary memory is transferred to permanent memory as new Initial and Type Library entries. If these libraries are at capacity, the oldest information is deleted (forgotten) and the new information is inserted.

As suggested by the theory, past experience is utilized to obtain the best prediction of events, but if the computer finds no matching experience upon which to draw, it must reason through the problem as best it can. Knowledge grows when new structures are formed or old ones reconstructed.

C. ANALYSIS OF THE GAME

After a rather poor start, the computer has gone ahead of its opponents in total points for all games, and the gap is widening. This is due partly to some minor changes in the strategy decision routines, however evidence indicates that most of the credit can be given to the building of the libraries. This contention is supported by the fact that the analysis portion of the strategy routine tends to increase (tune up) the basis of the coefficient assigned to the experience factor, and tune down some of the others.

Based on observation of the game and discussions with those playing it, it is evident that the players and the computer use many of the same factors in deciding upon a strategy and a concession point. In addition, concepts built by the human are similar to the concepts formed by the computer and stored within its memory. The average values for the reliability estimates that the computer maintains on both itself and its opponent remain relatively constant as the game progresses. The same is true of the weighted reliability.

IV. CONCLUSIONS

"If a program can be constructed that, when placed in a prisoner's dilemma situation, exhibits behavior like the behavior of people when placed in the same situation, then that program will be a powerful tool for generalizations."

The above statement by Professor Weil is repeated from an earlier section of this thesis for emphasis. In its existing form, this program does provide a powerful tool for extracting generalizations regarding human behavior in a medium risk decision making task. With but minor changes in wording or by incorporating extensions to the game, the program can be made applicable to almost any field of corporate or governmental endeavor requiring a psychological understanding of human behavior in making decisions where different gains can be achieved at varying risks.

A. EXTENSIONS AND CHANGES

DIPLOMAT is in itself an extension to the classic prisoner's dilemma game. It may be extended or changed to broaden its applicability to real world situations and make the game considerably more interesting.

The simplest change to the program would be to change the name assigned to the three strategies. For example, the applicability of the program could be changed by transforming the words arm/status-quo/disarm into the words buy/wait/sell, or perhaps raise-prices/no-change/lower-prices.

The whole outcome of the game can be changed drastically by changing

the POINTS formula. This is done by simply assigning different coefficients to the factors of WEALTH and STRENGTH, or by changing the values in Table I (the wealth changes for the various strategies shown as a function of player strength). Thus the game may be altered to match actual conditions encountered in, say, the business community.

The number of strategies from which to choose could be changed in either direction. Decreasing the choice to two would render the game closer to the prisoner's dilemma situation, but even this has many applications in the real world. On the other hand, it would be more interesting to increase the number of strategies. For example, there could be two levels of arming and two levels of disarming, or for buying or selling. Some implications of increasing the number of strategies, however, would be that the pattern-matching routines may not be feasible in their present state. It would probably require a pattern-matching scheme which placed more emphasis on the meta-level, i.e., looking at the broad spectrum of the pattern from a higher level, rather than pattern-matching individual strategies or small groups of strategies.

Increasing the number of players is probably the most difficult of the possible extensions to the game, but provides the most interesting possibilities. If the number of players is increased, treaties between the players and alliances among groups of players may be proposed and formed. An infinite number of situations may arise out of this idea, for example, one nation might wonder whether his ally will abide by the alliance or possibly turn on him several moves hence; or, a player might hesitate to sign an agreement when a better one might be offered from a different player. In the diplomacy situation, several nations may disarm to gain wealth, then form an alliance against a stronger nation which

has armed to gain strength at the cost of wealth. Increasing the number of players would also make the concepts of reasoning and learning more difficult, but more fascinating. For example, will one nation risk an alliance with another when it remembers prior treachery? Instead of merely analyzing what one's opponent thinks of him, a player will have to analyze several players' estimates of all the opponents. The reliability considerations become almost overwhelming.

B. POSSIBLE USES OF DIPLOMAT

It is considered that Professor Weil was correct in assuming that the prisoner's dilemma computer game would be a powerful tool for generalizations. The game of DIPLOMAT or its extensions would be an invaluable aid in the training of executives prior to stepping into positions requiring the art and finesse of personal contact. The speed of the computer permits the game to proceed rapidly and permits many different situations to be established by changing payoffs and formulas. The performance of different players against standard setups could then be analyzed.

In addition, personnel in the study of behavioral science and psychology could develop a better understanding of the nature of the human decision-making process and of the risk of striking out on an independent path as opposed to the benefits and security gained by cooperating. The fact that the risky path may lead to greater gains may be more important to some people than to others.

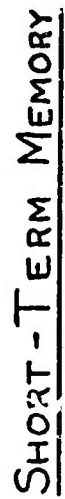
The training of college students in the theory of marketing and analysis would also be enhanced by applying textbook concepts to the difficulties of the everchanging conditions and types of people with

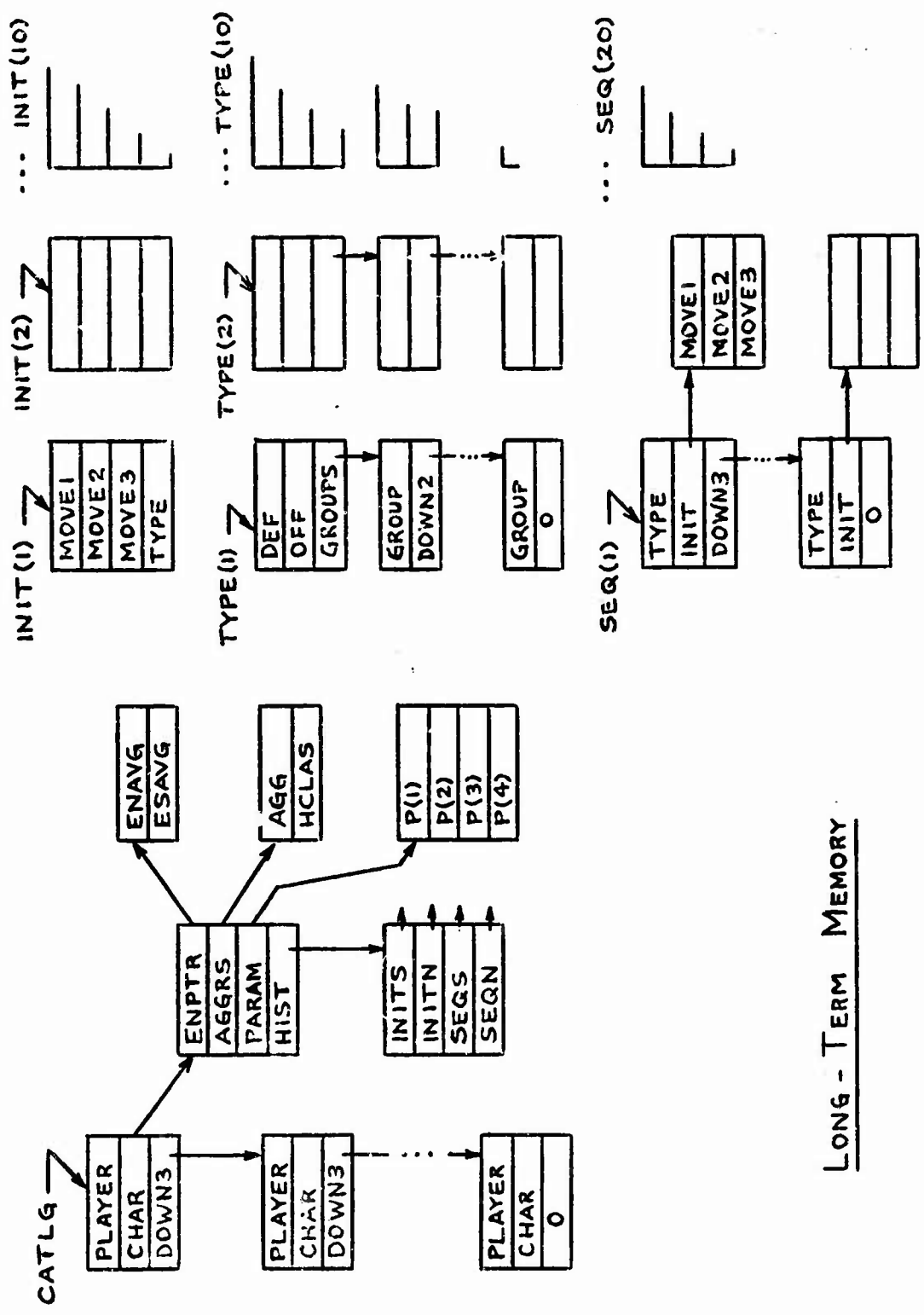
which they may deal. This training might prove useful for young officers in the diplomatic corps or even at junior level armed forces colleges.

In order to enhance the development of computer application to this type of work, it is recommended that consideration be given to the possibility of adapting this program to marketing or financial decision-making courses in the management curriculum. This type of program could be extremely useful as a tool in teaching management students the power and usefulness of the computer. It demonstrates the interface capabilities of man and machine. It can also be used as a teaching aid in artificial intelligence, game theory, and basic management-decision courses. The first part of Section III describing the game could be reproduced for use as a handout for potential players.

The concept and performance of DIPLOMAT appears to be good. It is ahead of the human players in POINTS and performs well in adapting to different human strategies that have been tried against it. However, similar to a human, it does not win every game, indicating that even it has more to learn.

THE MEMORY STRUCTURE OF DIPLOMAT





LONG - TERM MEMORY

APPENDIX B
EXAMPLE COMPUTER TERMINAL OUTPUT

DO YOU WISH TO READ MEMORY AND REBUILD THE HISTORY OF PLAYERS, YES OR NO "A3" FORMAT.
YES

THIS PROGRAM REQUIRES THE FOLLOWING INPUTS:

1. YOUR NAME IN A4 FORMAT, RIGHT JUSTIFIED IF LESS THAN FOUR CHARACTERS IN LENGTH.
2. INRAND, AN ODD INTEGER OF NINE OR LESS DIGITS, OR RANDOM ORIGIN, FOR USE IN GENERATING RANDOM NUMBERS.

YOUR NAME:
JIM

INRAND:
555

A TOSS OF A COIN WILL BE USED TO DETERMINE WHO GOES FIRST.
YOU MAY CALL HEAD OR TAIL:
HEAD

SORRY YOU LOST THE TOSS OF THE COIN.
THEREFORE I WILL MAKE THE FIRST PROPOSAL (CONCESSION POINT).

PARAMETERS AT START OF GAME ARE: 10 10 10 20

MY PROPOSAL (CONCESSION POINT) WHERE 1=ARM, 2=MAINTAIN THE STATUS QUO, AND 3=DISARM IS: 2.

THE APPROXIMATE PAYOFF MATRIX FOR THE EXISTING CONDITIONS OF STRENGTH AND WEALTH IS SHOWN BELOW.
INDICATED PAYOFFS ARE TO SOUTH (YOU).

STRATEGY	SOUTH		
	1	2	3

N	-1	1	2
O	*	*	*
R	-1	0	1
T	*	*	*
H	2	-1	2

WHAT IS YOUR PROPOSAL, STRATEGY 1(ARM), 2(MAINTAIN THE STATUS QUO), OR 3(DISARM): 3

IT IS NOW TIME TO CARRY OUT THE STRATEGY OF EACH SIDE.
THE COMPUTER'S MOVE IS LOCKED INTO THE SYSTEM.
PLEASE INDICATE YOUR STRATEGY AS 1, 2, OR 3:

THE ACTUAL STRATEGIES CHOSEN BY EACH OPPONENT ARE AS FOLLOWS:
NORTH(COMPUTER): 1
SOUTH(PLAYER): 3

THE RESULTS OF THE LAST MOVE ARE AS FOLLOWS:

	WEALTH	STRENGTH	*POINTS*
COMPUTER(NORTH)	-1	1	7
PLAYER(SOUTH)	5	-1	5

IT IS NOW MOVE NUMBER 2. IT IS YOUR TURN TO GO FIRST.

THE APPROXIMATE PAYOFF MATRIX IS UNCHANGED FROM THE LAST MOVE.
YOU MAY USE IT AS AN AID IN DECIDING YOUR STRATEGY AND CP.

WHAT IS YOUR PROPOSAL, STRATEGY 1(ARM), 2(MAINTAIN THE STATUS QUO), OR 3(DISARM) ?

3 MY PROPOSAL (CONCESSION POINT) WHERE 1=ARM, 2=MAINTAIN THE STATUS QUO, AND 3=DISARM IS: 1.

NOTE THAT MY CP IS TO ARM.

PLEASE INDICATE WITH A "YES" OR "NO" (RIGHT JUSTIFIED IN A3 FORMAT) IF YOU WISH TO RENEGOTIATE:
NO

IT IS NOW TIME TO CARRY OUT THE STRATEGY OF EACH SIDE.
THE COMPUTER'S MOVE IS LOCKED INTO THE SYSTEM.

PLEASE INDICATE YOUR STRATEGY AS 1, 2, OR 3:

1

THE ACTUAL STRATEGIES CHOSEN BY EACH OPPONENT ARE AS FOLLOWS:

NORTH(COMPUTER): 1

SOUTH(PLAYER): 1

THE RESULTS OF THE LAST MOVE ARE AS FOLLOWS:

	WEALTH	STRENGTH	*POINTS*
COMPUTER (NORTH) *	-2	2	9 *
PLAYER (SOUTH) *	4	0	7 *

IT IS NOW MOVE NUMBER 3. IT IS MY TURN TO GO FIRST.

MY CP IS: 2.

THE APPROXIMATE PAYOFF MATRIX IS UNCHANGED FROM THE LAST MOVE.
YOU MAY USE IT AS AN AID IN DECIDING YOUR STRATEGY AND CP.

WHAT IS YOUR PROPOSAL, (1,2,OR 3) ?

2

WHAT IS YOUR STRATEGY ?

2

THE ACTUAL STRATEGIES CHOSEN BY EACH OPPONENT ARE AS FOLLOWS:
NORTH(COMPUTER): 1
SOUTH(PLAYER): 2

THE RESULTS OF THE LAST MOVE ARE AS FOLLOWS:

COMPUTER (NORTH) *	WEALTH	*	STRENGTH	*	*POINTS*
PLAYER (SOUTH) *	-6	*	3	*	6
	7	*	0	*	13
					**

IT IS NOW MOVE NUMBER 4. IT IS YOUR TURN TO GO FIRST.

THE APPROXIMATE PAYOFF MATRIX FOR THE EXISTING CONDITIONS OF STRENGTH AND WEALTH IS SHOWN BELOW.
INDICATED PAYOFFS ARE TO SOUTH (YOU).

STRATEGY	SOUTH			
	1	2	3	4

N	1	3	4	
O	*	*	*	
R	1	2	3	
T	*	*	*	
H	4	1	4	
	*	*	*	

WHAT IS YOUR PROPOSAL, (1,2,OR 3) ?

3

MY CP IS: 2.

WHAT IS YOUR STRATEGY ?

THE ACTUAL STRATEGIES CHOSEN BY EACH OPPONENT ARE AS FOLLOWS:

NORTH(COMPUTER): 1
SOUTH(PLAYER): 1

THE RESULTS OF THE LAST MOVE ARE AS FOLLOWS:

COMPUTER (NORTH) *	WEALTH	STRENGTH	*POINTS*
PLAYER (SOUTH) *	-11	4	0
	5	1	13
		*	*

IT IS NOW MOVE NUMBER 5. IT IS MY TURN TO GO FIRST.

MY CP IS: 3.

THE APPROXIMATE PAYOFF MATRIX IS UNCHANGED FROM THE LAST MOVE.
YOU MAY USE IT AS AN AID IN DECIDING YOUR STRATEGY AND CP.

WHAT IS YOUR PROPOSAL, (1,2,OR 3) ?

2

WHAT IS YOUR STRATEGY ?

1

THE ACTUAL STRATEGIES CHOSEN BY EACH OPPONENT ARE AS FOLLOWS:
NORTH(COMPUTER): 3
SOUTH(PLAYER): 1

THE RESULTS OF THE LAST MOVE ARE AS FOLLOWS:

COMPUTER (NORTH) *	WEALTH	*	STRENGTH	*	*POINTS*
PLAYER (SOUTH) *	-9	*	3	*	-1
	2	*	2	*	16
					*

IT IS NOW MOVE NUMMER 6. IT IS YOUR TURN TO GO FIRST.
 THE APPROXIMATE PAYOFF MATRIX IS UNCHANGED FROM THE LAST MOVE.
 YOU MAY USE IT AS AN AID IN DECIDING YOUR STRATEGY AND CP.

WHAT IS YOUR PROPOSAL, (1,2,OR 3) ?
 2
 MY CP IS: 3.

WHAT IS YOUR STRATEGY ?
 3
 THE ACTUAL STRATEGIES CHOSEN BY EACH OPPONENT ARE AS FOLLOWS:
 NORTH(COMPUTER): 3
 SOUTH(PLAYER): 3

THE RESULTS OF THE LAST MOVE ARE AS FOLLOWS:

COMPUTER (NORTH) *	WEALTH	*	STRENGTH	*	*POINTS*
PLAYER (SOUTH) *	-7	*	2	*	0
	3	*	1	*	15
					*

IT IS NOW MOVE NUMBER 7. IT IS MY TURN TO GO FIRST.

MY CP IS: 2.

THE APPROXIMATE PAYOFF MATRIX FOR THE EXISTING CONDITIONS OF STRENGTH AND WEALTH IS SHOWN BELOW.
INDICATED PAYOFFS ARE TO SOUTH (YOU).

STRATEGY	SOUTH		
	1	2	3

N	-1	1	2
O	*	*	*
R	-1	0	1
T	*	*	*
H	2	-1	2
	*	*	*

WHAT IS YOUR PROPOSAL, (1,2,OR 3) ?

2
WHAT IS YOUR STRATEGY ?

1
THE ACTUAL STRATEGIES CHOSEN BY EACH OPPONENT ARE AS FOLLOWS:
NORTH(COMPUTER): 3
SOUTH(PLAYER): 1

THE RESULTS OF THE LAST MOVE ARE AS FOLLOWS:

COMPUTER (NORTH)	WEALTH	STRENGTH	*POINTS*
PLAYER (SOUTH)	-5	1	-1
	-1	2	16
	*	*	*

IT IS NOW MOVE NUMBER 8. IT IS YOUR TURN TO GO FIRST.

THE APPROXIMATE PAYOFF MATRIX FOR THE EXISTING CONDITIONS OF STRENGTH AND WEALTH IS SHOWN BELOW.
 INDICATED PAYOFFS ARE TO SOUTH (YOU).

STRATEGY	SOUTH			
	1	2	3	

N	1	3	4	
O				
R	1	2	3	
T				
H	4	1	4	

WHAT IS YOUR PROPOSAL, (1,2,OR 3) ?

1
 MY CP IS: 2.

WHAT IS YOUR STRATEGY ?

2
 THE ACTUAL STRATEGIES CHOSEN BY EACH OPPONENT ARE AS FOLLOWS:
 NORTH(COMPUTER): 3
 SOUTH(PLAYER): 2

THE RESULTS OF THE LAST MOVE ARE AS FOLLOWS:

COMPUTER (NORTH) *	WEALTH	STRENGTH	*POINTS*
PLAYER (SOUTH) *	-3	0	-2
	-2	2	14
	*	*	*

IT IS NOW MOVE NUMBER 9. IT IS MY TURN TO GO FIRST.

MY CP IS: 1.

NOTE THAT MY CP IS TO ARM.
PLEASE INDICATE WITH A "YES" OR "NO" (RIGHT JUSTIFIED IN A3 FORMAT) IF YOU WISH TO RENEGOTIATE:

YES

BASED ON YOUR REQUEST FOR RENEGOTIATION, I HAVE CONSIDERED MY CONCESSION POINT. MY NEW CP IS TO: 2.

THE APPROXIMATE PAYOFF MATRIX FOR THE EXISTING CONDITIONS OF STRENGTH AND WEALTH IS SHOWN BELOW.
INDICATED PAYOFFS ARE TO SOUTH (YOU).

STRATEGY	SOUTH		
	1	2	3

N	-3	-1	0
O	*	*	*
R	-3	-2	-1
T	*	*	*
H	0	-3	0

61

WHAT IS YOUR PROPOSAL, (1,2,OR 3) ?

3

WHAT IS YOUR STRATEGY ?

THE ACTUAL STRATEGIES CHOSEN BY EACH OPPONENT ARE AS FOLLOWS:

NORTH(COMPUTER): 3
SOUTH(PLAYER): 3

THE RESULTS OF THE LAST MOVE ARE AS FOLLOWS:

COMPUTER (NORTH) *	WEALTH	STRENGTH	*POINTS*
PLAYER (SOUTH) *	0	-1	1
			15
			*

IT IS NOW MOVE NUMBER 10. IT IS YOUR TURN TO GO FIRST.
 THE APPROXIMATE PAYOFF MATRIX IS UNCHANGED FROM THE LAST MOVE.
 YOU MAY USE IT AS AN AID IN DECIDING YOUR STRATEGY AND CP.

WHAT IS YOUR PROPOSAL, (1,2,OR 3) ?

3

MY CP IS: 3.

WHAT IS YOUR STRATEGY ?

2

THE ACTUAL STRATEGIES CHOSEN BY EACH OPPONENT ARE AS FOLLOWS:
 NORTH(COMPUTER): 3
 SOUTH(PLAYER): 2

THE RESULTS OF THE LAST MOVE ARE AS FOLLOWS:

COMPUTER (NORTH) *	WEALTH	STRENGTH	*POINTS*
PLAYER (SOUTH) *	3	-2	2
	0	1	15
	*	*	*

IT IS NOW MOVE NUMBER 11. IT IS MY TURN TO GO FIRST.

MY CP IS: 2.

THE APPROXIMATE PAYOFF MATRIX FOR THE EXISTING CONDITIONS OF STRENGTH AND WEALTH IS SHOWN BELOW.
INDICATED PAYOFFS ARE TO SOUTH (YOU).

STRATEGY	SOUTH		
	1	2	3

N	-1	1	2
O	*	*	*
R	-1	0	1
T	*	*	*
H	2	-1	2
	*	*	*

WHAT IS YOUR PROPOSAL, (1,2,OR 3) ?

63

WHAT IS YOUR STRATEGY ?

THE ACTUAL STRATEGIES CHOSEN BY EACH OPPONENT ARE AS FOLLOWS:
NORTH(COMP): 3
SOUTH(PLAYER): 3

THE RESULTS OF THE LAST MOVE ARE AS FOLLOWS:

	WEALTH	* STRENGTH	*POINTS*
COMPUTER (NORTH) *	5	-3	3
PLAYER (SOUTH) *	3	0	18
			*

IT IS NOW MOVE NUMBER 12. IT IS YOUR TURN TO GO FIRST.
 THE APPROXIMATE PAYOFF MATRIX FOR THE EXISTING CONDITIONS OF STRENGTH AND WEALTH IS SHOWN BELOW.
 INDICATED PAYOFFS ARE TO SOUTH (YOU).

STRATEGY	SOUTH		
	1	2	3

N	-3	-1	0
C	-3	-2	-1
T			
H	4	1	4

WHAT IS YOUR PROPOSAL, (1,2,OR 3) ?

MY CP IS: 2.

WHAT IS YOUR STRATEGY ?

THE ACTUAL STRATEGIES CHOSEN BY EACH OPPONENT ARE AS FOLLOWS:
 NORTH(COMPUTER): 3
 SOUTH(PLAYER): 1

THE RESULTS OF THE LAST MOVE ARE AS FOLLOWS:

	WEALTH	*STRENGTH	*POINTS*
COMPUTER (NORTH)	7	-4	2
PLAYER (SOUTH)	0	1	21
			*
			*

IT IS NOW MOVE NUMBER 13. IT IS MY TURN TO GO FIRST.

MY CP IS: 2.

THE APPROXIMATE PAYOFF MATRIX IS UNCHANGED FROM THE LAST MOVE.
YOU MAY USE IT AS AN AID IN DECIDING YOUR STRATEGY AND CP.

WHAT IS YOUR PROPOSAL, (1,2,OR 3) ?

2

WHAT IS YOUR STRATEGY ?

3

THE ACTUAL STRATEGIES CHOSEN BY EACH OPPONENT ARE AS FOLLOWS:
NORTH(COMPUTER): 3
SOUTH(PLAYER): 3

THE RESULTS OF THE LAST MOVE ARE AS FOLLOWS:

COMPUTER (NORTH) *	WEALTH	STRENGTH	*POINTS*
PLAYER (SOUTH) *	3	-5	3
		0	24
			*

IT IS NOW MOVE NUMRER 14. IT IS YOUR TURN TO GO FIRST.
 THE APPROXIMATE PAYOFF MATRIX IS UNCHANGED FROM THE LAST MOVE.
 YOU MAY USE IT AS AN AID IN DECIDING YOUR STRATEGY AND CP.

WHAT IS YOUR PROPOSAL, (1,2,OR 3) ?

3
 MY CP IS: 2.

WHAT IS YOUR STRATEGY ?
 2 THE ACTUAL STRATEGIES CHOSEN BY EACH OPPONENT ARE AS FOLLOWS:
 NORTH(COMPUTER): 2
 SOUTH(PLAYER): 2

THE RESULTS OF THE LAST MOVE ARE AS FOLLOWS:

COMPUTER (NORTH) *	WEALTH	STRENGTH	*POINTS*
PLAYER (SOUTH) *	12	-5	11
	3	0	24
			**

IT IS NOW MOVE NUMBER 15. IT IS MY TURN TO GO FIRST.

MY CP IS: 3.

THE APPROXIMATE PAYOFF MATRIX FOR THE EXISTING CONDITIONS OF STRENGTH AND WEALTH IS SHOWN BELOW.
INDICATED PAYOFFS ARE TO SOUTH (YOU).

STRATEGY	SOUTH		
	1	2	3
	*****	*****	*****
N	* -5	* -3	* -2
C	* -3	* -2	* -1
T	* *	* *	* *
H	* 4	* 1	* 4

WHAT IS YOUR PROPOSAL, (1,2,OR 3) ?
3

WHAT IS YOUR STRATEGY ?

THE ACTUAL STRATEGIES CHOSEN BY EACH OPPONENT ARE AS FOLLOWS:
NORTH(COMPUTER): 1
SOUTH(PLAYER): 3

THE RESULTS OF THE LAST MOVE ARE AS FOLLOWS:

COMPUTER (NORTH) *	WEALTH	STRENGTH	*POINTS*
PLAYER (SOUTH) *	13	-4	20
	7	-1	27
			*

IT IS NEW MOVE NUMBER 16. IT IS YOUR TURN TO GO FIRST.

THE APPROXIMATE PAYOFF MATRIX IS UNCHANGED FROM THE LAST MOVE.
YOU MAY USE IT AS AN AID IN DECIDING YOUR STRATEGY AND CP.

WHAT IS YOUR PROPOSAL, (1,2,OR 3) ?

1

MY CP IS: 1.

NOTE THAT MY CP IS TO ARM.

PLEASE INDICATE WITH A "YES" OR "NO" (RIGHT JUSTIFIED IN A3 FORMAT) IF YOU WISH TO RENEGOTIATE:

YES

BASED ON YOUR REQUEST FOR RENEGOTIATION, I HAVE CONSIDERED MY CONCESSION POINT. MY NEW CP IS TO: 2.

WHAT IS YOUR STRATEGY ?

2

THE ACTUAL STRATEGIES CHOSEN BY EACH OPPONENT ARE AS FOLLOWS:

NORTH(COMPUTER): 2

SOUTH(PLAYER): 2

THE RESULTS OF THE LAST MOVE ARE AS FOLLOWS:

COMPUTER (NORTH) *	WEALTH	*	STRENGTH	*	*POINTS*
PLAYER (SOUTH) *	16	*	-4	*	26
	6	*	-1	*	25
					*

IT IS NOW MOVE NUMBER 17. IT IS MY TURN TO GO FIRST.

MY CD IS: 2.

THE APPROXIMATE PAYOFF MATRIX FOR THE EXISTING CONDITIONS OF STRENGTH AND WEALTH IS SHOWN BELOW.
INDICATED PAYOFFS ARE TO SOUTH (YOU).

STRATEGY	SOUTH		
	1	2	3

N	-3	-1	0
O	*	*	*
R	-3	-2	-1
T	*	*	*
H	4	1	4
	*	*	*

WHAT IS YOUR PROPOSAL, (1,2,OR 3) ?

69

WHAT IS YOUR STRATEGY ?

THE ACTUAL STRATEGIES CHOSEN BY EACH OPPONENT ARE AS FOLLOWS:
NORTH(COMPUTER): 1
SOUTH(PLAYER): 1

THE RESULTS OF THE LAST MOVE ARE AS FOLLOWS:

COMPUTER (NORTH) *	WEALTH	*	STRENGTH	*	*POINTS*
PLAYER (SOUTH) *	16	*	-3	*	30
	2	*	0	*	21
					*

IT IS NOW MOVE NUMBER 18. IT IS YOUR TURN TO GO FIRST.

THE APPROXIMATE PAYOFF MATRIX IS UNCHANGED FROM THE LAST MOVE.
 YOU MAY USE IT AS AN AID IN DECIDING YOUR STRATEGY AND CP.

WHAT IS YOUR PROPOSAL, (1,2,OR 3) ?

1

MY CP IS: 2.

WHAT IS YOUR STRATEGY ?

3

THE ACTUAL STRATEGIES CHOSEN BY EACH OPPONENT ARE AS FOLLOWS:

NORTH(COMPUTER): 2

SOUTH(PLAYER): 3

THE RESULTS OF THE LAST MOVE ARE AS FOLLOWS:

COMPUTER (NORTH) *	WEALTH	STRENGTH	*POINTS*
PLAYER (SOUTH) *	17	-3	32
	6	-1	24
			* *

IT IS NOW MOVE NUMBER 19. IT IS MY TURN TO GO FIRST.

MY CP IS: 2.

THE APPROXIMATE PAYOFF MATRIX IS UNCHANGED FROM THE LAST MOVE.
YOU MAY USE IT AS AN AID IN DECIDING YOUR STRATEGY AND CP.

WHAT IS YOUR PROPOSAL, (1,2,OR 3) ?
3

WHAT IS YOUR STRATEGY ?
1

THE ACTUAL STRATEGIES CHOSEN BY EACH OPPONENT ARE AS FOLLOWS:
NORTH(COMPUTER): 1
SOUTH(PLAYER): 1

THE RESULTS OF THE LAST MOVE ARE AS FOLLOWS:

COMPUTER (NORTH) *	WEALTH	STRENGTH	*POINTS*
PLAYER (SOUTH) *	15	-2	32
	4	0	24
			* *

IT IS NOW MOVE NUMBER 20. IT IS YOUR TURN TO GO FIRST.

THE APPROXIMATE PAYOFF MATRIX IS UNCHANGED FROM THE LAST MOVE.
 YOU MAY USE IT AS AN AID IN DECIDING YOUR STRATEGY AND CP.

WHAT IS YOUR PROPOSAL, (1,2,OR 3) ?

2

MY CP IS: 2.

WHAT IS YOUR STRATEGY ?

1

THE ACTUAL STRATEGIES CHOSEN BY EACH OPPONENT ARE AS FOLLOWS:

NCRTH(COMPUSER): 2

SOUTH(PLAYER): 1

THE RESULTS OF THE LAST MOVE ARE AS FOLLOWS:

COMPUTER (NORTH) *	WEALTH	*	STRENGTH	*	*POINTS*
PLAYER (SOUTH) *	16	*	-2	*	34
	2	*	1	*	25
					*
					*

IT IS NOW MOVE NUMBER 21. IT IS MY TURN TO GO FIRST.

MY CP IS: 2.

THE APPROXIMATE PAYOFF MATRIX IS UNCHANGED FROM THE LAST MOVE.
YOU MAY USE IT AS AN AID IN DECIDING YOUR STRATEGY AND CP.

WHAT IS YOUR PROPOSAL, (1,2,OR 3) ?

WHAT IS YOUR STRATEGY ?

THE ACTUAL STRATEGIES CHOSEN BY EACH OPPONENT ARE AS FOLLOWS:

NORTH(COMPUTER): 3

SOUTH(PLAYER): 3

*** BY A RANDOM SELECTION, THE UMPIRE HAS DECIDED TO END THIS GAME AFTER 21 MOVES.
THE PARTICIPANT WITH THE MOST POINTS (POINTS = (2*WEALTH)+(5*STRENGTH)) IS DECLARED THE WINNER.

THE FINAL GRAND TOTALS FOR THIS GAME ARE:

	WEALTH	STRENGTH	POINTS*
COMPUTER(NORTH)	18	-3	35
PLAYER(SOUTH)	6	0	30

THE COMPUTER WON THE GAME. THANK YOU FOR PLAYING.

PARAMETERS AT END OF GAME ARE: 14 6 16 4

COMPUTER PROGRAM LISTING

```

*****
** THE MAIN PROGRAM CONTROLS THE OPERATION OF THE GAME OF
** DIPLOMAT. IT IS SUPPORTED IN THIS TASK BY THE SEVEN
** SUBROUTINES LISTED WITHIN THIS SECTION.
*****
IMPLICIT INTEGER (A-W)
COMMON /TDJE/APN,APS,STACK(40)
COMMON /PAR/P(4),S(4),PRBLS(5)
COMMON /ENP/ENARRY(50),ESARRY(50),TOPEN,TOPES
COMMON /CLS/DEFES,BDEFES,BOFFEN,PMOVEN,PMOVES
COMMON /JEC/VALUE,PW,NC,ISSE,FCNXX,PN,PS,EACTXN,FACXS
1SN,SS,WN,WS,RS,FCNXX,GO,NELAG,TURN,ANAL,RECON
2PNOLD,ISN,ISS,FCNXX,GO,NELAG,TURN,ANAL,RECON
COMMON /ST/STTRUE,LASTAIL4,ATABN(3),ATABN(3),ATABS(3),
COMMON /T/AVAIL4,ATM,CATLG,CHAR,HIST,PLAYER,CATLGN,PATRNS,PATPNN
COMMON /TD/TOP,RK(10),STKM(10),STKNU,TYNCR
COMMON /TDP2/STK(10),TYPE1,TYSOU,TYNCR
COMMON /TDP8/RLHIST,NAHIST
COMMON /TDP3/TYPE(20),INIT(20)
COMMON /TDP4/SEQ(20),SCONTR
COMMON /TDP7/GRNUPS,GRUPN,TMOVES,TMOVEN,MCNT,CONTR
DIMENSION ION,RLINK4(1000),LLINK4(1000),DOWN4(1000),DOWN2(1000)
1(1000),DOWN2(1000)
EQUIVALENCE (NAME4(4),RLINK4(3),LLINK4(2),DOWN4(1)),(NAME2(2),DOWN
2(1)),(NAME3(2),RLINK3(2),DOWN3(1))
DATA HEADS,'HEAD',TAILS,'TAIL',YES,'YES',NO,'NO',
DATA AI,'A1',A2,'A2',A1,'A1',A2,'A2',
APN=0
APS=0
CATLGN=0
NC=1
SCONTR=0
INCNTR=0
INCNTR=10
CONTR=0
DO 180 I=1,4
PRBLS(I)=0
180 S(I)=0
PRBLS(5)=0

```

```

FACTXN=0
FACTXS=0
VALUE(1,1)=0
VALUE(2,1)=-1
VALUE(3,1)=-2
VALUE(4,1)=-3
VALUE(5,1)=-4
VALUE(6,1)=-6
VALUE(7,1)=3
VALUE(1,2)=2
VALUE(2,2)=1
VALUE(3,2)=0
VALUE(4,2)=-2
VALUE(5,2)=-3
VALUE(6,2)=3
VALUE(7,2)=3
VALUE(1,3)=3
VALUE(2,3)=4
VALUE(3,3)=3
VALUE(4,3)=3
VALUE(5,3)=3
VALUE(6,3)=2
VALUE(7,3)=2
(INITIALIZING ROUTINE)
CALL INITL
CALL ECALC(ENACT,ESACT)
CALL REMEMB(INCNT,INCNT)

C
C
C      (INPUTS)
399  WRITE (6,399) THIS PROGRAM REQUIRES THE FOLLOWING INPUTS:'.',EX,
1. YOUR NAME IN AN A4 FORMAT, RIGHT JUSTIFIED IF LESS THAN FOUR
2 CHARACTERS IN LENGTH.'. ',5X,'2. INRAND, AN ODD INTEGER OR
3 LESS DIGITS, OF RANDOM ORIGIN, FOR USE IN GENERATING RANDOM NUMPE
4RS.')
6001 WRITE (6,3991)
3991 FORMAT (//,' YOUR NAME ?')
      READ (5,204,END=6000) PLAYER
3992 WRITE (6,3992)
      FORMAT (//,' INRAND = ?')
3993 READ (5,3993) INRAND
      FORMAT (//,' INRAND = ?')
C
      SN=0
      SS=0
      WN=0
      WS=0

```

```

RS=0
RN=0
PN=0
PS=0
TYPE1=0
TYPE1=C
TYSQR=0
TYNOR=0
ECNXN=0
ECNXS=0
ANAL=0
RECON=-10
ISSN=4
ISS=4
ENAVG=0
ESAVG=0
LASTGO=0
EFLAG=0
NFLAG=0
WHOCAL=0
NC=1
PCNTT=0
PCNTM=0
PCNTN=0
PCNTNM=0
TOPL1=C
TOPL2=C
TOPLIN=0
TOPL2N=0
PCOUNT=0
MOV=0
CALL TABCHG
CALL CATLOG
CALL PATRN

```

CCCCCCCC

THE PROGRAM NOW DETERMINES WHO MAKES THE FIRST MOVE. SELECTION IS BY A RANDOM PROCESS.

TURN=+1 INDICATES THAT THE PLAYER MAKES THE FIRST PROPOSAL.
TURN=-1 INDICATES THAT THE COMPUTER MAKES THE FIRST PROPOSAL.

```

I=HEADS
CALL RANDOM
IF (YRAND:GE:0.5) I=TAILS
WRITE (6,203)
203 FORMAT (/, 'A TOSS OF A COIN WILL BE USED TO DETERMINE WHO GOES
1 FIRST. YOU MAY CALL "HEAD" OR "TAIL":')
READ (5,204) J

```

```

204 FORMAT (A4)
   IF (J.NE.I) GO TO 250
C
205 WRITE (6,205)
   FORMAT (//, 'CONGRATULATIONS, YOU WON THE TOSS OF THE COIN. YOU
   1MAY MAKE THE FIRST PROPOSAL (CONCESSION POINT).')
   GO TO 200
C
201 WRITE (6,202) NC
202 FORMAT (//, 'IT IS NOW MOVE NUMMER',I3,'. IT IS YOUR TURN TO GO
   1FIRST.')
200 IF (NC.EQ.1) GO TO 206
   DO 2007 I=1,3
   DO 2007 J=1,3
   IF (RTABN(I,J).NE.COMP(N(I,J)) GO TO 2006
2007 CONTINUE
   GO TO 2008
2006 WRITE (6,2000) ((RTABS(I,J),J=1,3),I=1,3)
2000 FORMAT (//, 'THE APPROXIMATE PAYOFF MATRIX FOR THE EXISTING CONDIT
   1IONS OF STRENGTH AND WEALTH IS SHOWN BELOW.//, 'INDICATED PAYOFF
   2ES ARE TO SOUTH (YOU).//, 'T28, 'SOUTH.//, 'T6, 'STRATEGY.//, '1, '5X,
   3, '2, '5X, '3, '1, '19X, '*****//, '19X, '3, '7X, 'N, '3X,
   4, '1, '7X, '1, '14, '216, '7X, '1, '11X, '1, '7X, '2, '7X, '1, '14, '216,
   5, '7X, '1, '11X, '1, '7X, '3, '7X, '1, '14, '216, '//)
   GO TO 2010
2008 WRITE (6,2009)
2009 FORMAT (//, 'THE APPROXIMATE PAYOFF MATRIX IS UNCHANGED FROM THE L
   1AST MOVE.//, 'YOU MAY USE IT AS AN AID IN DECIDING YOUR STRATEGY
   2 AND CP.//)
2010 IF (NC.GT.2) GO TO 2002
   WRITE (6,2001)
2001 FORMAT (//, 'WHAT IS YOUR PROPOSAL, STRATEGY 1 (ARM), 2 (MAINTAIN THE S
   1TATUS QUO), OR 3 (DISARM) ?')
   GO TO 2005
2002 WRITE (6,2003)
2003 FORMAT (//, 'WHAT IS YOUR PROPOSAL, (1,2,OR 3) ?')
C
2005 READ (5,206) CPS
2006 FORMAT (//)
   IF (CPS.NE.1.AND.CPS.NE.2.AND.CPS.NE.3) GO TO 410
C
   (SINCE CPS HAS BEEN DECLARED, THE SYSTEM MAY DECIDE ITS
   STRAIN AND CPN NOW, WITHOUT FURTHER RECONSIDERATION.)
   CALL STRGCM(MOV, INCNT, PCNT, PCNTM, INCNTR, PCNTN, PCNTNM)
   CALL CPNDCN(MOV, INCNT, PCNTN, PCNTNM, INCNTR)
C
   IF (NC.GT.2) GO TO 2061

```

```

WRITE (6,207) CPN
GO TO 2065
2061 WRITE (6,2062) CPN
2062 FORMAT (//, 'MY CP IS:', I2, ' ')
C 2065 IF (CPN.GT.1) GO TO 300
WRITE (6,2071)
2071 FORMAT (//, 'NOTE THAT MY CP IS TO ARM.', /, 'PLEASE INDICATE WITH
1A "YES" OR "NO" (RIGHT JUSTIFIED IN A3 FORMAT) IF YOU WISH TO RENE
2GOTIATE:')
3 READ (5,2072) I
2072 FORMAT (A3)
IF (I.EQ.NO) GO TO 300
WHOCAL=1
CALL RENEG(PROBLM,WHOCAL)
GO TO 300
C 250 WRITE (6,251)
251 FORMAT (//, 'SORRY, YOU LOST THE TOSS OF THE COIN, THEREFORE I WILL
1 MAKE THE FIRST PROPOSAL (CONCESSION POINT).')
GO TO 252
C 255 WRITE (6,256) NC
256 FORMAT (//, 'IT IS NOW MOVE NUMBER', I3, '. IT IS MY TURN TO GO FIR
252 TUPN=-1
C THE PROGRAM NOW DECIDES UPON A STRATEGY, BUT THE STRATEGY MAY BE
C CHANGED IN LIGHT OF SOUTH CONCESSION POINT AFTER IT IS DECLARED.
C CALL STRTGY(MOV,INCNT,PCNT,PCNTM,INCNT,PCNTN,PCNTNM)
C HAVING DECIDED UPON A TENTATIVE STRATEGY, THE SYSTEM NOW
C DETERMINES THE COMPUTERS CONCESSION POINT.
C CALL CPNDCN(MOV,INCNT,PCNTN,PCNTNM,INCNT)
C IF (NC.GT.2) GO TO 2075
WRITE (6,207) CPN
207 FORMAT (//, 'MY PROPOSAL (CONCESSION POINT) WHERE 1=ARM, 2=MAINTAI
1 IN THE STATUS QUO, AND 3=DISARM IS:', I2, ' ')
GO TO 2079
2075 WRITE (6,2076) CPN
2076 FORMAT (//, 'MY CP IS:', I2, ' ')
C (DETERMINE IF SOUTH WANTS TO RENEGOTIATE:)
2079 IF (CPN.GT.1) GO TO 2089
WRITE (6,2071)

```

```

4      READ (5,2072) I
      IF (I.EQ.NO) GO TO 2089
      WHOCAL=1
      CALL RENEG(PROBLM,WHOCAL)
      (NOW ASK FOR SOUTH'S CONCESSION POINT.)
C 2089  IF (NC.EQ.1) GO TO 208
      DO 2081 I=1,3
      DO 2081 J=1,3
      IF (RTARN(I,J).NE.COMPNI(I,J)) GO TO 208
2081  CONTINUE
      WRITE (6,2009)
      GO TO 2084
208  WRITE (6,2000) ((RTABS(I,J),J=1,3),I=1,3)
2084  IF (NC.GT.2) GO TO 2085
      WRITE (6,2001)
      GO TO 209
2085  WRITE (6,2003)
209  READ (5,206) CPS
      IF (CPS.NE.1.AND.CPS.NE.2.AND.CPS.NE.3) GO TO 420
C      THE PROGRAM NOW RECONSIDERS ITS PREVIOUSLY CHOSEN STRATEGY SINCE
C      SOUTH HAS NOW DECLARED HIS CPS.
      NC=NC+1
      CALL STRTGY(MOV,INCNT,PCNT,PCNTM,INCNT,PCNTN,PCNTNM)
      NC=NC-1
C
300  IF (NC.GT.2) GO TO 3001
      WRITE (6,301)
301  FORMAT (//,IT IS NOW TIME TO CARRY OUT THE STRATEGY OF EACH SIDE
1      THE COMPUTER'S MOVE IS LOCKED INTO THE SYSTEM.,/,PLEASE IND
2      ICATE YOUR STRATEGY AS 1, 2, OR 3:!)
      GO TO 3005
3001  WRITE (6,3002)
3002  FORMAT (//,WHAT IS YOUR STRATEGY ?)
3005  READ (5,206) STRATS
3011  IF (STRATS.NE.1.AND.STRATS.NE.2.AND.STRATS.NE.3) GO TO 430
302  FORMAT (//,THE ACTUAL COMPSTRATEGIES CHOSEN BY EACH OPPONENT ARE AS F
1      FOLLOWS:TN.EQ.1.AND.STRATS.EQ.1) GO TO 3021
      IF (STRATN.EQ.1.AND.STRATS.EQ.1) GO TO 3022
      IF (STRATN.EQ.3.AND.STRATS.EQ.3) GO TO 3023
      IF (STRATN.EQ.3.AND.STRATS.EQ.1) GO TO 3024
      GO TO 302
3021  FACTXN=-1
302  FACTXS=-1

```

```

GO TO 303
FACTXN=4
GO TO 303
3022 GO TO 303
FACTXN=2
3023 FACTXS=2
GO TO 303
FACTXS=4
3024 CA'L PATUPD
303
C 317 PNOLD=PN
C (THIS VALUE IS LATER USED TO DETERMINE ECONOMIC CONDITIONS.)
C DETERMINE ISN
C IF (SN) 101,102,103
C SN NEGATIVE
C 101 IF (SN.LT.-4) GO TO 1011
IF (SN.LE.-2) GO TO 1012
ISN=4
GO TO 104
1012 ISN=3
GO TO 104
1011 IF (SN.LT.-6) GO TO 1013
ISN=2
GO TO 104
1013 ISN=1
GO TO 104
C SN ZERO
C 102 ISN=4
GO TO 104
C SN POSITIVE
C 103 IF (SN.GT.4) GO TO 1031
IF (SN.GE.2) GO TO 1032
ISN=4
GO TO 104
1032 ISN=5
GO TO 104
1031 IF (SN.GT.6) GO TO 1033
ISN=6
GO TO 104
1033 ISN=7
C

```

```

C
C
C 104      DETERMINE ISS
C 111      IF (SS) 111,112,113
C          IF (SS.LT.-4) GO TO 1111
C          IF (SS.LT.-2) GO TO 1112
C          ISS=4
C          GO TO 105
C 1112     ISS=3
C          GO TO 105
C 1111     IF (SS.LT.-6) GO TO 1113
C          ISS=2
C          GO TO 105
C 1113     ISS=1
C          GO TO 105
C          SS ZERO
C 112      ISS=4
C          GO TO 105
C          SS POSITIVE
C 113      IF (SS.GT.4) GO TO 1131
C          IF (SS.GE.2) GO TO 1132
C          ISS=4
C          GO TO 105
C 1132     ISS=5
C          GO TO 105
C 1131     IF (SS.GT.6) GO TO 1133
C          ISS=6
C          GO TO 105
C 1133     ISS=7
C 105      WN=WN+VALUE(I SN,STRATN)+ECNXN
C          WS=WS+VALUE(I SS,STRATS)+ECNXS
C          JSTN=STRATN-2
C          JSTS=STRATS-2
C          IF (JSTN) 106,107,108
C          SN=SN+1
C          GO TO 107
C 106      SN=SN-1
C 108      IF (JSTS) 109,120,110
C 107      SS=SS+1
C 109      GO TO 120
C 110      SS=SS-1
C 120      PN=PN+2*(VALUE(I SN,STRATN)+ECNXN)-5*(STRATN-2)+FACTXN
C          PS=PS+2*(VALUE(I SS,STRATS)+ECNXS)-5*(STRATS-2)+FACTXS
C          RW=WN-WS

```

```

C      RS=SN-SS
C      (CALL ANALYSES PORTION OF STRTG TO ANALYZE THE RESULTS OF THE
C      LAST MOVE TO DETERMINE IF STRTG COULD HAVE DONE BETTER.)
C      NC=NC+1
C      CALL STRTG(MOV,INCNT,PCNT,PCNTM,INCNT,PCNTN,PCNTNM)
C      NC=NC-1
C      WHOCAL=0
C      NFLAG=0
C      (RENEGOTIATE FLAGS CLEARED SINCE MOVE IS COMPLETE.)
C      CALL ECONOMY
C      (DETERMINE ECONOMIC CONDITIONS BY CALLING SUBROUTINE ECONOMY.)
C      DO 121 I=1,3
C      DO 121 J=1,3
C      COMPN(I,J)=RTABN(I,J)
C      CALL TABCHG
C      FACTXN=0
C      FACTXS=0
C      NC=NC+1
C      CALL ECALC(ENACT,ESACT)
C      NC=NC-1
C      ENAPRY(NC)=ENACT
C      ESAPRY(NC)=ESACT
C      ENAVG=ENAVG+ENACT
C      ESAVG=ESAVG+ESACT
C      IS IT END OF GAME YET?
C      THE NUMRER OF MOVES IN THE GAME VARIES RANDOMLY BETWEEN 10 AND 50.
C      IF (NC.LT.10) GO TO 480
C      IF (NC.GE.50) GO TO 490
C      CALL RANDOM
C      IF (YRAND.GE.0.9) GO TO 490
C      IF (NC.GT.40.AND.YRAND.GE.0.4) GO TO 490
C      IF (NC.GT.30.AND.YRAND.GE.0.5) GO TO 490
C      IF (NC.GT.20.AND.YRAND.GE.0.6) GO TO 490
C      NC=NC+1
C      480 WRITE (6,4801) WN,SN,PN,WS,SS,PS
C      4801 FORMAT (//,1 THE RESULTS OF THE LAST MOVE ARE AS FOLLOWS:,,//,
C      121X,WEALTH,5X,STRENGTH,5X,*POINTS*,//,
C      2, COMPUTER (NORTH) *,17,3X,*,*,18,4X,*,*,18,3X,*,*,//,
C      3, PLAYER (SOUTH) *,17,3X,*,*,18,4X,*,*,18,3X,*,*,//,
C      IF (IABS(RS).GE.10) GO TO 492
C      IF (IABS(RW).GE.30) GO TO 493
C      IF (TURN) 201,201,255

```

```

C 492 IF (RS) 494,494,495
C 493 IF (RW) 496,496,497

410 WRITE (6,411)
411 FORMAT (//, '*** YOUR SELECTION OF A CONCESSION POINT WAS NOT CORRE
      1CT. THE CP MUST BE A SINGLE INTEGER 1, 2, OR 3.', //, 'PLEASE INDIC
      2ATE YOUR CP AS 1 (ARM), 2 (STATUS QUD), OR 3 (DISARM):')
      GO TO 2005
420 WRITE (6,411)
430 GO TO 209
431 WRITE (6,431)
431 FORMAT (//, '*** YOUR SELECTION OF A STRATEGY MUST BE A SINGLE INTE
      1GER 1, 2, OR 3.', //, 'PLEASE INDICATE YOUR STRATEGY AS 1 (ARM), 2 (ST
      2ATUS QUD), OR 3 (DISARM):')
      READ (5,206) STRATS
      GO TO 3011

C 494 WRITE (6,4941)
4941 FORMAT (//, '*** CONGRATULATIONS, YOU WON THIS GAME BASED ON YOUR
      1STRENGTH SUPERIORITY.')
      WON=-1
      NC=NC-1
      GO TO 491
495 WRITE (6,4951)
4951 FORMAT (//, '*** SORRY, YOU LOST THIS GAME BASED ON YOUR STRENGTH
      1DEFICIENCY.')
      WON=1
      NC=NC-1
      GO TO 491
496 WRITE (6,4961)
4961 FORMAT (//, '*** CONGRATULATIONS, YOU WON THIS GAME BASED ON YOUR
      1WEALTH SUPERIORITY.')
      WON=-1
      NC=NC-1
      GO TO 491
497 WRITE (6,4971)
4971 FORMAT (//, '*** SORRY, YOU LOST THIS GAME BASED ON YOUR WEALTH DEF
      1ICIENCY.')
      WON=1
      NC=NC-1
      GO TO 491
499 WRITE (6,4991)
4991 FORMAT (//, '*** THE GAME HAS NOW ENDED. THE PARTICIPANT WITH THE
      1MOST POINTS (POINTS = (2*WEALTH)+(5*STRENGTH)) IS DECLARED THE WIN
      2NER.')
      IF (PN-PS) 391,392,393
391 WON=-1

```



```

NAME4(PARAM)=10
RLINK4(PARAM)=10
LLINK4(PARAM)=10
DOWN4(PARAM)=20
162 WRITE (6,1605) (P(I),I=1,4)
1605 FORMAT (/,' PARAMETERS AT END OF GAME ARE:',4I5)
C
CHAR=RLINK2(CATLG)
TOTAL1=NAME4(LLINK4(PATRNS))
TOTAL2=RLINK4(LLINK4(PATRNS))
TOTAL3=LLINK4(LLINK4(PATRNS))
IF (TOTAL1.GT.TOTAL2.AND.TOTAL1.GT.TOTAL3) GO TO 150
IF (TOTAL3.GT.TOTAL2.AND.TOTAL3.GT.TOTAL1) GO TO 151
NAME2(RLINK4(CHAR))=2
GO TO 152
150 NAME2(RLINK4(CHAR))=1
GO TO 152
151 NAME2(RLINK4(CHAR))=3
152 ENAVG=ENAVG/NC
ESAVG=ESAVG/NC
NAME2(NAME4(CHAR))=ENAVG
DOWN2(NAME4(CHAR))=ESAVG
J=0
K=0
DO 157 I=1,NC
IF (ENARRY(I).EQ.10.OR.ENARRY(I).EQ.20) J=J+1
IF (ENARRY(I).EQ.30.OR.ENARRY(I).EQ.40) K=K+1
157 CONTINUE
IF (J.GT.K) HCLAS=2
IF (J.LT.K) HCLAS=3
IF (J.EQ.K) HCLAS=1
DOWN2(RLINK4(CHAR))=HCLAS
CALL TYPELB(INCNTR,WON,INCNT)
CALL SEQLB(INCNTR,WON)
CALL DPATNS
GO TO 6001
6000 WRITE (6,702)
702 FORMAT(' DO YOU WANT TO PRINT OUT THE PERMANENT MEMORY. YES OR NO
1. USE A3 FORMAT RIGHT JUSTIFIED,')
703 READ(5,703) DU
FORMAT(A3)
CALL SAVE(INCNTR,INCNT,DU)
C
STOP
END
C

```

```

SUBROUTINE TABCHG
THIS SUBROUTINE UPDATE THE TABLES AS A RESULT OF THE LAST MOVE.
IT ALSO DETERMINES IF AN "ECONOMIC CONDITIONS" CHANGE IS DUE.
ECONOMIC CHANGES ARE RANDOMLY GENERATED AT RANDOM TIMES.

IMPLICIT INTEGER (A-W)
COMMON/JEC/VALUE(7,3), RTARN(3,3), RTABS(2,3), ATARN(3), ATARS(3),
1 SN, SS, WN, WS, RS, RW, NC, INRAND, YRAND, EFLAG, CPN, CPS, STRATN, STRATS,
2 PNOLD, ISN, ISS, ECNXN, ECNXS, PN, PS, FACTXN, FACTXS
TI=1
TJ=1
TTN=2*VALUE(ISN,TI)-5*(TI-2)
TTS=2*VALUE(ISS,TJ)-5*(TJ-2)
RTABS(TI,TJ)=TTN-TTS
IF(TJ.EQ.3) GO TO 3
TJ=TJ+1
GO TO 2
3 ATARN(TI)=TTN
ATABS(TI)=2*VALUE(ISS,TI)-5*(TI-2)
IF(TI.EQ.3) GO TO 10
TI=TI+1
TJ=1
GO TO 1
10 RTARN(1,1)=RTARN(1,1)-1
RTABS(1,1)=RTABS(1,1)-1
RTABS(1,3)=RTABS(1,3)+4
RTABS(3,1)=RTABS(3,1)+2
RTABS(3,3)=RTABS(3,3)+2
ATABS(1)=ATARN(1)+2
ATABS(3)=ATARN(3)+1
AT (THE ATARN(S) FACTORS APPROXIMATE THE GAINS FOR COOPERATING
IN DISARMING, OR FOR ONE SIDE ARMING WHILE OTHER DISARMS.)

THIS COMPLETES THE TABLE CHANGES. THE ECONOMIC CONDITION IS NOW
DETERMINED. THE FIRST STEP IS TO DETERMINE IF A CHANGE IS DUE.
IF IT IS, THE CONDITIONS FOR THE OPPONENTS (COMPUTER IS "NORTH",
OPPONENT IS "SOUTH") ARE PLACED INTO PARAMETERS ECNXN AND ECNXS
RESPECTIVELY.

CALL RANDOM
IF (YRAND*GE.0.6) GO TO 30
1 (STMT 30 IS "RETURN", BRANCH THERE IF NO CHANGE IS DUE)

```

```

C C C C C
C C C C C
      IN ORDER TO DETERMINE THE ECONOMIC CONDITIONS, A RANDOM NUMBER
      MUST FIRST BE GENERATED.
      CALL RANDOM
      THE ECONOMIC CONDITION FOR THE COMPUTER (NORTH) IS CALCULATED
      FIRST AND PLACED INTO ECNXN.
      YRAND IS THE RANDOM NUMBER USED TO DETERMINE THE ECONOMIC CHANGE.
      IF (YRAND.GE.0.7) GO TO 35
      IF (YRAND.LE.0.3) GO TO 33
      ECNXN=0
      GO TO 40
33  IF (YRAND.LE.0.1) GO TO 34
      ECNXN=-1
      GO TO 40
34  ECNXN=-2
      GO TO 40
35  IF (YRAND.GE.0.9) GO TO 36
      ECNXN=1
      GO TO 40
36  ECNXN=2

C C C C C
      THE ECONOMIC CONDITION FOR SOUTH IS THEN CALCULATED AND PLACED
      INTO ECNXS. IT USES A DIFFERENT RANDOM NO. THAN FOR NORTH.
      CALL RANDOM
      IF (YRAND.GE.0.7) GO TO 45
      IF (YRAND.LE.0.3) GO TO 43
      ECNXS=0
      RETURN
43  IF (YRAND.LE.0.1) GO TO 44
      ECNXS=-1
      RETURN
44  ECNXS=-2
      RETURN
45  IF (YRAND.GE.0.9) GO TO 46
      ECNXS=1
      RETURN
46  ECNXS=2
      RETURN
      END

```

```

SUBROUTINE RANDOM
  THE PURPOSE OF THIS SUBROUTINE IS TO GENERATE A FLOATING POINT
  RANDOM NUMBER IN THE RANGE 0-1.0. THE PROCEDURE IS ESSENTIALLY
  IDENTICAL TO THE "SCIENTIFIC SUBROUTINE PACKAGE" PROCEDURE "RANDIIM".

  PASSING PARAMETER INRAND IS USED TO START THE PROCESS OF RANDOM
  NUMBER GENERATION. IT IS ANY ODD INTEGER NUMBER WITH NINE OR LESS
  DIGITS INPUT IN A (HOPEFULLY) RANDOM MANNER WHEN THE PLAYER OR LESS
  INITIALLY SIGNS INTO THE SYSTEM. AFTER THE FIRST ENTRY TO THIS
  SUBROUTINE IN ANY ONE GAME, INRAND BECOMES THE PREVIOUS VALUE OF
  INRAND COMPUTED BY THIS SUBROUTINE.

  IMPLICIT INTEGER (A-W)
  COMMON/JEC/VALUE(7,3),RTABN(3,3),RTABS(3,3),ATABN(3),ATARS(3),
  1SN,SS,WN,WS,RS,RW,NC,INRAND,YRAND,EFLAG,CN,CPS,STRATN,STRATS,
  2PN,D,I,ISN,ISS,ECN,N,ECNXS,PN,PS,FACTXN,FACTXS
  INRAND=INRAND*65535
  IF (INRAND) 31,32,32
  31 INRAND=INRAND+2147483647+1
  32 YRAND=YRAND*.4656613F-9
  RETURN
END

```

CCCCCCCCCCCC

```

SUBROUTINE INITL
  THIS SUBROUTINE INITIALIZES AVAILABLE CELLS FOR 250,333,
  AND 500 NODES

  IMPLICIT INTEGER (A-W)
  COMMON/T/AVAIL4,AVAIL3,AVAIL2,NAME4(1000),NAME2(1000),NAME3(1000)
  COMMON/TO/TOP,BOTTOM,CATLG,CHAR,HIST,PLAYER,CATLGN,PATRNS,PATRNN
  DIMENSION RLINK4(1000),LLINK4(1000),DOWN4(1000),RLINK3(1000),DOWN3
  1(1000),DOWN2(1000)
  EQUIVALENCE (NAME4(4),RLINK4(4),RLINK4(2),DOWN4(1)),(NAME2(2),DOWN
  12(1)),(NAME3(3),RLINK3(2),DOWN3(1))
  N=500
  M=999
  P=999
  DO 10 I=4,N,4
    NAME4(I)=I+1
    AVAIL4=1
  NAME4(1000)=0
  DO 11 I=2,M,2
    NAME2(I)=I+1
  NAME2(1000)=0

```

CCCC

```

12      AVAIL2=1
        DO 12 I=3,P,3
          NAME3(I)=I+1
          AVAIL3=1
          NAME3(999)=0
        RETURN
      END

```

```

CCCCC
SUBROUTINE GET2(K)
      THIS SUBROUTINE GETS A CELL FROM THE LIST OF AVAILABLE
      CELLS IN AVAIL2. "WK" IS THE PASSING PARAMETER FOR SUBSCRIPT
      OF CELL OBTAINED. "AVAIL2" IS THE POINTER TO THE NEXT
      CELL IN THE LIST OF AVAIL2 CELLS.

```

```

      IMPLICIT INTEGER (A-W)
      COMMON/T/AVAIL4,AVAIL3,AVAIL2,NAME4(1000),NAME2(1000),NAME3(1000)
      COMMON/TD/TOP,BOTTOM,CATLG,CHAR,HIST,PLAYER,CATLGN,PATRNS,PATRNN
      DIMENSION RLINK4(1000),LLINK4(1000),DOWN4(1000),PLINK3(1000),DOWN3
      1(1000),DOWN2(1000)
      EQUIVALENCE (NAME4(4),RLINK4(3),LLINK4(2),DOWN4(1)),(NAME2(2),DOWN
      12(1)),(NAME3(3),RLINK3(2),DOWN3(1))
      IF(AVAIL2.EQ.0) GOTO 2
      K=AVAIL2
      AVAIL2=DOWN2(AVAIL2)
      RETURN
2      WRITE(6,3)
3      FORMAT(IH0,5X,'UNDERFLOW EXISTS IN AVAIL2 LIST OF AVAILABLE CELL
      IS.')
      RETURN
      END

```

```

CCCCC
SUBROUTINE GET3(L)
      THIS SUBROUTINE GETS A CELL FROM THE LIST OF AVAILABLE
      CELLS IN AVAIL3. "WL" IS THE PASSING PARAMETER FOR SUBSCRIPT
      OF CELL OBTAINED. "AVAIL3" IS THE POINTER TO THE NEXT
      CELL IN THE LIST OF AVAIL3 CELLS.

```

```

      IMPLICIT INTEGER (A-W)
      COMMON/T/AVAIL4,AVAIL3,AVAIL2,NAME4(1000),NAME2(1000),NAME3(1000)
      COMMON/TD/TOP,BOTTOM,CATLG,CHAR,HIST,PLAYER,CATLGN,PATRNS,PATRNN
      DIMENSION RLINK4(1000),LLINK4(1000),DOWN4(1000),PLINK3(1000),DOWN3
      1(1000),DOWN2(1000)
      EQUIVALENCE (NAME4(4),RLINK4(3),LLINK4(2),DOWN4(1)),(NAME2(2),DOWN
      12(1)),(NAME3(3),PLINK3(2),DOWN3(1))

```

```

IF(AVAIL3.EQ.0) GOTO 2
L=AVAIL3
AVAIL3=DOWN3(AVAIL3)
RETURN
WRITE(6,3)
FORMAT(1H0,5X,'UNDERFLOW EXISTS IN AVAIL3 LIST OF AVAILABLE CELL
1S.')
```

```

2
3
```

SUBROUTINE GET4(J)

THIS SUBROUTINE GETS A CELL FROM THE AVAILABLE CELLS IN
AVAIL4. "J" IS THE PASSING PARAMETER FOR SUBSCRIPT OF
CELL OBTAINED. "AVAIL4" IS THE POINTER TO NEXT CELL IN
THE LIST OF AVAIL4 CELLS.

```

IMPLICIT INTEGER (A-W)
COMMON/T/AVAIL4,AVAIL3,AVAIL2,NAME4(1000),NAME2(1000),NAME3(1000)
COMMON/TD/TOP,BOTM,CATLG,CHAR,HIST,PLAYER,CATLGN,PATRNS,PATRNN
DIMENSION RLINK4(1000),LLINK4(1000),DOWN4(1000),PLINK3(1000),DOWN3
1(1000),DOWN2(1000)
EQUIVALENCE (NAME4(4),RLINK4(3),LLINK4(2),DOWN4(1)),(NAME2(2),DOWN
12(1)),(NAME3(3),RLINK3(2),DOWN3(1))
IF(AVAIL4.EQ.0) GOTO 2
J=AVAIL4
AVAIL4=DOWN4(AVAIL4)
RETURN
WRITE(6,3)
FORMAT(1H0,5X,'UNDERFLOW EXISTS IN AVAIL4 LIST OF AVAILABLE CEL
1S.')
```

```

2
3
```

SUBROUTINE COUNT(K1,J1,L1)

THIS SUBROUTINE COUNT THE NUMBER OF CELLS STILL AVAILABLE
"K1,J1,L1" ARE PASSING PARAMETERS THE PROVIDE THE COUNT
FOR EACH LIST. K1=>AVAIL4,J1=>AVAIL2,L1=>AVAIL3.

```

IMPLICIT INTEGER (A-W)
COMMON/T/AVAIL4,AVAIL3,AVAIL2,NAME4(1000),NAME2(1000),NAME3(1000)
COMMON/TD/TOP,BOTM,CATLG,CHAR,HIST,PLAYER,CATLGN,PATRNS,PATRNN
DIMENSION RLINK4(1000),LLINK4(1000),DOWN4(1000),RLINK3(1000),DOWN3
1(1000),DOWN2(1000)
EQUIVALENCE (NAME4(4),RLINK4(3),LLINK4(2),DOWN4(1)),(NAME2(2),DOWN
```

```

CCCCC

```

```

CCCCC

```

```

12(1)), (NAME3(3),RLINK3(2),DOWN3(1))
K1=0
J1=0
L1=0
A4=AVAIL4
A3=AVAIL3
A2=AVAIL2
1 IF(A2.EQ.0)GOTO 2
A2=DOWN2(A2)
J1=J1+1
GOTO 1
2 IF(A4.EQ.0)GOTO 3
A4=DOWN4(A4)
K1=K1+1
GOTO 2
3 IF(A3.EQ.0)GOTO 4
A3=DOWN3(A3)
L1=L1+1
GOTO 3
4 RETURN
END

```

```

SUBROUTINE STRIGY(MOV,INCNT,PCNT,PCNTM,INCNT,PCNTN,PCNTNM)
*****
* THE NINE SUBROUTINES WITHIN THIS SECTION SUPPORT THE
* REASONING CAPABILITIES OF THE PROGRAM.
*
*****

```

THIS IS ONE OF THE KEY SUBROUTINES OF THE SYSTEM IN THAT IT IS THE SUBROUTINE IN WHICH THE "THOUGHT PROCESSES" OCCUR. THIS SUBROUTINE HAS THE FOLLOWING FUNCTIONS: AND DECIDE UPON A STRATEGY FOR THE MOVE. THIS IS DONE BY USING A VOTING PROCESS ON A POLYNOMIAL.

(1) ANALYZE ALL FACTORS AVAILABLE AND DECIDE UPON A STRATEGY FOR THE MOVE.

(2) RECONSIDER THE STRATEGY CHOSEN IF THE COMPUTER WAS FIRST TO DECLARE HIS CONCESSION POINT. THE RECONSIDERATION IS DONE IN LIGHT OF THE PROPOSED STRATEGY OF SOUTH. THE COMPUTER'S OPPONENT.

(3) ANALYSIS OF THE COMPLETED MOVE TO DETERMINE IF THE PARAMETERS USED FOR THE POLYNOMIAL COEFFICIENTS NEED BE TUNED UP OR DOWN TO GIVE A BETTER SOLUTION.

```

IMPLICIT INTEGER (A-W)
COMMON/TDP4/SEQ(20),SCONTR

```

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

```

COMMON/TDP3/TYPE(20),INIT(20)
COMMON/PAR/P(4),S(4),PRBLS(5)
COMMON/JEC/VALUE(7,3),RTARN(3,3),RTABS(3,3),ATARN(3),ATABS(3),
1SN,SS,WN,WS,RS,RW,NC,INRND,YR AND,EFLAG,CPN,CPS,STRATN,STRATS,
2PNOLD,ISN,ISS,ECNXXN,ECNXXS,PN,PS,FACTXN,FACTXS
COMMON/EEEE/EN,ES,TOPEN,TOPEX
COMMON/ENP/ENARRY(50),ESARRY(50)
COMMON/ST/STREUE,LASTGO,NFLAG,TURN,ANAL,RECON
COMMON/CLS/DEEES,BDEFS,OFFEN,ROFEN,PMOVEN,PMOVES
COMMON/TDPG/TYPENU,TYPE1,TYSOU,TYNOR
COMMON/T/AVAIL4,AVAIL3,AVAIL2,NAME4(1000),NAME2(1000),NAME3(1000)
COMMON/TD/TOP,ROTM,CATLG,CHAR,HIST,PLAYER,CATLGN,PATRNS,PATRNN
DIMENSION RLINK4(1000),LLINK4(1000),RLINK3(1000),DOWN3
1(1000),DOWN2(1000)
EQUIVALENCE (NAME4(4),RLINK4(2),DOWN4(1)),(NAME2(2),DOWN
12(1)),(NAME3(3),RLINK3(1))
DIMENSION CF(4)

```

(IF THE SYSTEM REMEMBERS HAVING PLAYED THIS OPPONENT, THE
PARAMETERS USED FOR THE COEFFICIENTS FOR THE TERM OF THE
POLYNOMIAL ARE TAKEN FROM THE HISTORY OF THIS PLAYER IN
LONG TERM MEMORY. IF NOT, THE PARAMETERS FOUND BELOW ARE
USED. -- THE FIRST TIME THIS SUBROUTINE IS CALLED IN A
GAME, THE PARAMETERS DESCRIBED ABOVE ARE USED. AFTER THE
FIRST MOVE, THESE PARAMETERS MAY BE MODIFIED BY THE ANALYSIS
PORTION OF THIS SUBROUTINE.)

```

IF (NC.GT.1) GO TO 7
CHAR=RLINK3(CATLG)
IF (NAME4(LLINK4(CHAR)),EQ.0) GO TO 5000
P(1)=NAME4(LLINK4(CHAR))
P(2)=RLINK4(LLINK4(CHAR))
P(3)=LLINK4(LLINK4(CHAR))
P(4)=DOWN4(LLINK4(CHAR))
WRITE (6,61) (P(I),I=1,4)
FORMAT (//, ' PARAMETERS AT START OF GAME ARE:',4I5)
GO TO 7
61
P(1)=1C
P(2)=1C
P(3)=1C
P(4)=20
WRITE (6,61) (P(I),I=1,4)
6000

```

(THE FIRST STEP IS TO DETERMINE IF SOUTH'S CONCESSION POINT WAS
DECLARED FIRST. IF SO, THIS CPS IS USED IN DETERMINING THE


```

C      COEFFICIENT "CF1" ASSIGNED TO THE 1ST TERM OF THE POLYNOMIAL.)
C
10 ANAL=10
C      (THIS SETS THE ANAL FLAG SO THAT ANALYSIS WILL
C      BE PERFORMED WHEN REQUESTED.)
C      GO TO (11,12,12),CPS
C
C      (A BRANCH TO 11 INDICATES CPS WAS TO ARM.)
11 IF (SS.LE.4) GO TO 12
C      (NO BRANCH INDICATES THAT THE COMPUTER'S OPPONENT IS QUITE
C      STRONG ALREADY, YET HE IS ARMING EVEN FURTHER: CONSIDER
C      RENEGOTIATION. BRANCHING TO 12 INDICATES NO RENEGOTIATE.)
C
C      IF (NFLAG.EQ.1) GO TO 12
C      (NFLAG=1 INDICATES THAT WE HAVE RENEGOTIATED, BUT SOUTH
C      PERSISTS IN ARMING.)
C      NFLAG=1
C      IF (RS.LT.-2) GO TO 1111
C      (NO BRANCH INDICATES THAT BOTH SIDES HAVE BEEN ARMING GREATLY.)
C      PROBLM=1
C      (I.E., "LET'S QUIT THE ARMS RACE.")
C      CALL RENEG(PROBLM,WHOCAL)
C      GO TO 10
1111 Z=NAME4(LLINK4(PATRNS))/NC
C      IF (Z.LE.0.5) GO TO 1112
C      PROBLM=2
C      (I.E., SOUTH IS QUITE STRONG, HAS MUCH STRENGTH SUPERIORITY,
C      AND HAS ARMED OVER 50% OF THE TIME.)
C      CALL RENEG(PROBLM,WHOCAL)
C      GO TO 10
1112 J=0
C      MOVES=DCWN4(PATRNS)
1113 IF (NAME2(MOVES).EQ.1) J=J+1
C      IF (DOWN2(MOVES).EQ.0) GO TO 1114
C      MOVES=DOWN2(MOVES)
C      GO TO 1113
1114 MOVES=DCWN4(PATRNS)
C      IF (J.LT.2) GO TO 12
C      J=0
C      PROBLM=3
C      (I.E., SOUTH IS QUITE STRONG AND HAS MUCH STRENGTH SUPERIORITY,
C      YET HAS ARMED AT LEAST 2 OUT OF THE LAST 3 MOVES.)
C      CALL RENEG(PROBLM,WHOCAL)
C      GO TO 10
C      (END CHECK FOR RENEGOTIATE BLOCK.)
C

```



```

15 CONTINUE
151 WRITE (6,151) *** S/R STRTGY ERROR #1 ***
152 FORMAT (//, ' *** S/R STRTGY ERROR #1 *** ')
153 RETURN
154 PRBLS(4)=1
155 HISMAX=MAXO(RTABS(1,1),RTABS(1,2),RTABS(1,3),RTABS(2,1),
1 RTABS(2,2),RTABS(2,3),RTABS(3,1),RTABS(3,2),RTABS(3,3))
DO I=1,3
DO J=1,3
IF (RTABS(I,J).EQ.HISMAX) GO TO 155
CONTINUE
153 WRITE (6,154)
154 FORMAT (//, ' *** S/R STRTGY ERROR #2 *** ')
155 RETURN
156 PRBLS(5)=J
C
C (NOW DETERMINE HIS PROBABLE STRATEGY BASED ON CPS.)
S(1)=0
S(2)=0
S(3)=0
DO I=1,5
IF (PRBLS(I).EQ.1) S(1)=S(1)+1
IF (PRBLS(I).EQ.2) S(2)=S(2)+1
IF (PRBLS(I).EQ.3) S(3)=S(3)+1
CONTINUE
16 HISMAX=MAXO(S(1),S(2),S(3))
DO I=1,3
IF (HISMAX.EQ.S(I)) GO TO 161
CONTINUE
161 WRITE (6,163)
RETURN
1611 S(1)=0
1612 S(2)=0
1613 S(3)=0
HISMAX=1
C
C (PICK COMPUTER'S VOTE FOR STRATEGY SELECTION BASED ON CPS TO
C MAXIMIZE RELATIVE GAIN)
S(1)=MAXO(RTABN(1,HISMAX),RTABN(2,HISMAX),RTABN(3,HISMAX))
DO I=1,3
IF (S(1).EQ.RTABN(I,HISMAX)) GO TO 164
CONTINUE
162 WRITE (6,163)
163 FORMAT (//, ' *** S/R STRTGY ERROR #3 *** ')
164 RETURN
164 S(1)=I
CF(1)=P(1)

```

```

C      IF (HISMAX.EQ.3.AND.S(1).EQ.1) CF(1)=P(1)+2
C      17 IF (RECON) 20,20,50
C      (THIS COMPLETES THE RECONSIDERATION PHASE)
C
C      THE SECOND TERM OF THE POLYNOMIAL IS BASED ON "GOAL". THIS IS AND
C      BASED ON ANALYSIS OF THE STRENGTH, WEALTH, AND POINTS SO FAR, AND
C      SELECTION OF THE GOAL OF THIS MOVE: OPTIMUM STRATEGY, MAXIMUM
C      GAIN, OR MINIMUM LOSS. A COEFFICIENT CF2 IS THEN ASSIGNED TO THIS
C      TERM OF THE POLYNOMIAL.
C      20 RP=PN-PS
C      IF (RP) 22,23,24
C
C      (IF COMPUTER'S POINTS LESS THAN PLAYER'S:)
C      22 IF (SN.LT.-4) GO TO 221
C      242 CALL MAXSTR(MAXGN)
C      S(2)=MAXGN
C      GO TO 29
C      221 S(2)=1
C      GO TO 29
C
C      (IF COMPUTER'S POINTS = PLAYER'S POINTS:)
C      23 IF (SN.LT.-4) GO TO 221
C      CALL OPTSTR(STRATO)
C      S(2)=STRATO
C      GO TO 29
C
C      (IF COMPUTER'S POINTS GREATER THAN PLAYER'S:)
C      24 IF (SN) 241,242,243
C      241 IF (RS) 221,242,243
C      243 CALL MINSTR(MINLOS)
C      S(2)=MINLOS
C
C      (DETERMINE CF2.)
C      29 CF(2)=P(2)
C
C      THE THIRD TERM OF THE STRATEGY POLYNOMIAL IS BASED ON
C      "LOOK-AHEAD" AND "GUESS-OPPOSITE".
C      LOOK-AHEAD IS DETERMINED BY PATTERN MATCHING THE STRATEGIES
C      EMPLOYED BY THE COMPUTER'S OPPONENT AGAINST THE STRATEGY LIBRARY
C      IN LONG-TERM MEMORY, THEN FOLLOWING THE BEST DEFENSE AS ALREADY
C      DETERMINED THEREIN.
C      A "GUESS-OPPOSITE" AGAINST THIS STRATEGY IS INFREQUENTLY
C      EMPLOYED WHERE IT CAN PROVIDE A GOOD RELATIVE GAIN AND SERVE TO
C      CONFUSE SOUTH. IT IS NOT USED VERY OFTEN, BECAUSE THE SYSTEM
C      STRIVES TO MAINTAIN A GOOD IMAGE OF RELIABILITY.

```



```

DO 3041 I=1,3
IF (S(3).EQ.RTABN(1,MOVE2)) GO TO 3043
CONTINUE
WRITE (6,3042)
FORMAT (//,' *** S/R STRTG Y ERROR #5 ***')
RETURN
3043 S(3)=I
CF(3)=P(3)
GO TO 40
31 IF (NC.GT.4)GOTO 32
CALL CLSMVS(MOV,INCNT,PCNT,PCNTM,INCNTR)
IF (DEFES.GT.0) GO TO 311
IF (BDEFS.GT.0) GO TO 311
CF(3)=0
GO TO 40
C 311 S(3)=MAXO(RTABN(1,PMOVES),RTABN(2,PMOVES),RTABN(3,PMOVES))
DO 3111 I=1,3
IF (S(3).EQ.RTABN(1,PMOVES)) GO TO 312
CONTINUE
3111 WRITE (6,3143)
3143 FORMAT (//,' *** S/R STRTG Y ERROR #6 ***')
CF(3)=0
GO TO 40
C 312 CALL CLSMVN(MOV,INCNT,PCNTN,PCNTNM,INCNTR)
IF (OFFEN.GT.0) GO TO 313
IF (BOFFEN.GT.0) GO TO 313
S(3)=I
CF(3)=P(3)/2
GO TO 39
C
C (HAVE BOTH PMOVES AND PMOVEN:)
C 313 IF (I.EQ.PMOVEN) GO TO 314
S(3)=I
CF(3)=P(3)/2
GO TO 39
314 S(3)=PMOVEN
CF(3)=P(3)
GO TO 39
C 32 NORSO=C
CALL CLSTYP(NORSO)
IF (PMOVES.GT.0) GO TO 321
CF(3)=C
GO TO 40
321 S(3)=MAXO(RTABN(1,PMOVES),RTABN(2,PMOVES),RTABN(3,PMOVES))
DO 3211 I=1,3

```

```

3211 IF (S(3).EQ.RTABN(I,PMOVES)) GO TO 322
      CONTINUE
3243 WRITE (6,3243)
      FORMAT (//, '*** S/R STRTGY ERROR #7 ***')
      CF(3)=C
      GO TO 40
C 322 NORSO=1
      CALL CLSTYP(NORSO)
      IF (PMOVEN.GT.0) GO TO 323
      S(3)=I
      CF(3)=P(3)/2
      GO TO 30
C      (HAVE BOTH PMOVES AND PMOVEN:)
C 323 IF (I.EQ.PMOVEN) GO TO 324
      S(3)=I
      CF(3)=P(3)/2
      GO TO 38
324 S(3)=PMOVEN
      CF(3)=P(3)
      GO TO 39
C
C
C      (NOW DETERMINE IF WE SHOULD GUESS OPPOSITE.)
C 38 I=NC-LASTGO
      IF (I.LT.5) GO TO 39
      (DON'T GUESS OPPOSITE MORE THAN EVERY FIVE MOVES.)
      (ALSO DON'T GUESS OPPOSITE IF COMPUTER'S RELIABILITY IS LOW.)
      TOPE=ES
      IF (NAME3(TOPE).LT.40) GO TO 39
      IF (GUESS.OPPOSITE) IF USED, IS AGAINST OUR PATTERN OF MOVES:)
      IF (PMOVEN.EQ.0) GO TO 30
      STRUE=PMOVEN
      LASTGO=NC
      CF(3)=P(3)*2
      GO TO 40
C
C      (IF GUESS OPPOSITE NOT CHOSEN:)
C 39 CALL RANDOM
      IF (YRAND.GT.0.2) GO TO 40
      IF (YRAND.GT.0.1) GO TO 395
      IF (S(3).EQ.1) S(3)=3
      IF (S(3).EQ.3) S(3)=1
      CF(3)=P(3)*3
      LASTGO=NC
      GO TO 40

```

```

395 S(3)=2
    CF(3)=P(3)*3
    LASTGO=NC

THE FOURTH TERM OF THE POLYNOMIAL IS BASED ON THE EXPECTED ECONOMIC CONDITIONS. COEFFICIENT CF4 IS ASSIGNED TO THIS TERM.

40 IF (EFLAG) 41,42,43

(EFLAG NEGATIVE INDICATES THAT A "BOOM" IS PROBABLY IN PROGRESS AND THAT PRICES ARE HIGH, AND THIS IS A BAD TIME TO ARM.)

41 S(4)=3
    CF(4)=P(4)
    GO TO 50

(EFLAG ZERO INDICATES THAT THE PROBABLE STATE OF THE ECONOMY IS "NORMAL". THUS NO SPECIAL CONSIDERATION FOR ECONOMIC CONDITIONS NEED BE TAKEN INTO ACCOUNT.)

42 CF(4)=C
    GO TO 50

(EFLAG POSITIVE INDICATES THAT A "CRISIS" IS PROBABLY IN PROGRESS, AND THAT PRICES ARE LOW. IF STRENGTH IS NOT ESPECIALLY HIGH, THIS IS A GOOD TIME TO ARM.)

43 IF (SN.GT.4) GO TO 431
    S(4)=1
    CF(4)=P(4)
    GO TO 50

J=MAX(C(ATABN(2), ATABN(3))
DO 432 I=2,3
IF (J.EQ.ATABN(I)) GO TO 434
CONTINUE
WRITE(6,433)
FORMAT(/,' *** S/R STRATEGY ERROR #9 ***')
S(4)=1
CF(4)=P(4)

(NOW COMBINE ALL THE TERMS INTO THE POLYNOMIAL AND DETERMINE STRAIGHTWAY A VOTING PROCESS.)

50 PRBL S(1)=0
PRBL S(2)=0
PRBL S(3)=0
DO 501 I=1,4
IF (S(I).EQ.1) PRBL S(1)=PRBL S(1)+CF(I)
IF (S(I).EQ.2) PRBL S(2)=PRBL S(2)+CF(I)
IF (S(I).EQ.3) PRBL S(3)=PRBL S(3)+CF(I)
CONTINUE
501 CONTINUE

```

```

J=MAXO(PRBL(1),PRBL(2),PRBL(3))
DO 502 I=1,3
IF (J.EQ.PRBL(I)) GO TO 504
502 CONTINUE
WRITE (6,503) *** S/R STRTG ERROR #11 ***
503 FORMAT (//, ' *** S/R STRTG ERROR #11 *** ')
504 IF (PRBL(1).EQ.PRBL(2).OR.PRBL(1).EQ.PRBL(3).OR.
1 PRBL(2).EQ.PRBL(3)) GO TO 51
505 STRATN=I
RETURN
51 IF (PRBL(1).EQ.PRBL(2)) GO TO 511
IF (PRBL(1).EQ.PRBL(3)) GO TO 512
IF (J.NE.PRBL(2)) GO TO 505
STRATN=MAXO(ATABN(2),ATABN(3))
DO 510 K=2,3
IF (STRATN.EQ.ATABN(K)) GO TO 5103
510 CONTINUE
5101 WRITE (6,5102)
5102 FORMAT (//, ' S/R STRTG ERROR #12 *** ')
5103 STRATN=K
RETURN
511 IF (J.NE.PRBL(1)) GO TO 505
STRATN=MAXO(ATABN(1),ATABN(2))
DO 5110 K=1,2
IF (STRATN.EQ.ATABN(K)) GO TO 5103
5110 CONTINUE
5111 GO TO 5101
512 IF (J.NE.PRBL(1)) GO TO 505
STRATN=MAXO(ATABN(1),ATABN(3))
DO 5120 K=1,3,2
IF (STRATN.EQ.ATABN(K)) GO TO 5103
5120 CONTINUE
5121 GO TO 5101
(THIS COMPLETES THE STRATEGY DECISION AND RECONSIDERATION
PHASES OF THIS SUBROUTINE.)
CCCCCCCCCCCC
( THE THIRD FUNCTION OF THIS SUBROUTINE IS TO ANALYZE THE
RESULTS OF THE LAST MOVE AND DETERMINE IF A CORRECT CHOICE
OF STRATEGY WAS MADE. IF SO, WELL AND GOOD, BUT IF NOT
THE PARAMETERS P(I) ARE CHANGED TO REFLECT WHAT MAY HAVE
BEEN A BETTER CHOICE. ANALYSIS IS NOT DONE IF THE "EMERGENCY"
PROCEDURES OF STATEMENT 1 OF THIS SUBROUTINE ARE CARRIED OUT.)
7C ANAL=0
(THIS RESETS THE ANAL FLAG.)
C

```



```

C
DOWN3(L)=0
NAME3(L)=50
ROTEN=L
RLINK3(L)=1
CALL GET3(L)
DOWN3(L)=ROTEN
NAME3(L)=50
MIDEN=L
RLINK3(L)=3
CALL GET3(L)
DOWN3(L)=MIDEN
TOPEN=L
NAME3(L)=50
RLINK3(L)=6
EN=L
  THEN, ES:)
  CALL GET3(L)
  DOWN3(L)=0
  ROTES=L
  NAME3(L)=50
  RLINK3(L)=1
  CALL GET3(L)
  DOWN3(L)=ROTEN
  NAME3(L)=50
  MIDES=L
  RLINK3(L)=3
  CALL GET3(L)
  DOWN3(L)=MIDES
  TOPES=L
  NAME3(L)=50
  ES=L
  RLINK3(L)=6
  IF (NC.EQ.1) RETURN
1  NAME3(ROTEN)=NAME3(MIDEN)
   NAME3(MIDEN)=NAME3(TOPEN)
   NAME3(ROTES)=NAME3(MIDES)
   NAME3(MIDES)=NAME3(TOPES)
   J=CPN-STRATN+3
   K=CPN-STRATN+3
   GO TO (2,3,4,5,6),J
C
2  (SOUTH WAS PASSIVELY DISHONEST BY 2:)
   NAME3(TOPEN)=30
   J=4
   GO TO 10
C
3  (PASSIVELY DISHONEST BY 1:)
   NAME3(TOPEN)=40

```

```

C      J=2
      GO TO 10
      GO (COMPLETELY TRUTHFUL:)
      NAME3(TOPEN)=50
C      GO TO 12
      GO (AGGRESSIVELY DISHONEST BY 1:)
      NAME3(TOPEN)=20
C      J=8
      GO TO 10
      GO (AGGRESSIVELY DISHONEST BY 2:)
      NAME3(TOPEN)=10
C      J=10
      EVAL=NAME3(TOPEN)
      ENACT=NAME3(TOPEN)
      E=(6*EVAL+3*NAME3(MIDEN)+NAME3(ROTES))/10
      IF (E.LT.NAME3(MIDEN)) GO TO 15
      E=NAME3(MIDEN)-J
      GO TO 15
C      E=(6*NAME3(TOPEN)+3*NAME3(MIDEN)+NAME3(ROTES))/10
      ENACT=E
      NAME3(TOPEN)=E
C      GO TO (21,22,23,24,25),K
C
C      NAME3(TOPES)=30
      K=4
      GO TO 30
      NAME3(TOPES)=40
      K=2
      GO TO 30
      NAME3(TOPES)=50
      GO TO 32
      NAME3(TOPES)=20
      K=8
      GO TO 30
      NAME3(TOPES)=10
      K=10
C
      EVAL=NAME3(TOPES)
      ESACT=NAME3(TOPES)
      E=(6*EVAL+3*NAME3(MIDES)+NAME3(ROTES))/10
      IF (E.LT.NAME3(MIDES)) GO TO 35
      E=NAME3(MIDES)-K
      GO TO 35
C      E=(6*NAME3(TOPES)+3*NAME3(MIDES)+NAME3(ROTES))/10
      ESACT=E
      NAME3(TOPES)=E

```

```

C
RETURN
END

SUBROUTINE CPNDCN(MOV, INCNT, PCNTN, PCNTNM, INCNTR)

THIS SUBROUTINE DETERMINES THE CONCESSION POINT TO BE DECLARED
BY THE COMPUTER (NORTH), CPN. IT DOES THIS AFTER STRATN, THE
STRATEGY TO BE FOLLOWED, HAS BEEN DECIDED UPON BY S/P STRGY.

IMPLICIT INTEGER (A-W)
COMMON/ENP/ENAPRY(50), ESARRY(50)
COMMON/JEC/VA:UE(7,3), RTARN(3,3), RTABS(3,3), ATARN(3), ATABS(3),
1SN, SS, WN, WS, RS, RN, W, NC, INRAND, YPAND, EFLAG, CPN, CPS, STRATN, STRATS,
2PNOLD, ISN, ISS, ECNXN, ECNXS, PN, PS, FACTXN, FACTXS
COMMON/IDP4/SEQ(20), SCNTR
COMMON/IDP3/TYP(20), INIT(20)
COMMON/EEE/EN, ES, TOPEN, TOPES
COMMON/ST/STRUE, LASTGO, NFLAG, TURN, ANAL, RECON
COMMON/IDP9/TYPNU, TYPE1, TYSON, TYNCR
COMMON/CLS/DEFES, BDEFES, OFFEN, RGEFN, PMOVEN, PMOVES
COMMON/T/AVAIL4, AVAIL3, AVAIL2, NAME4(1000), NAME2(1000), NAME3(1000)
COMMON/TD/TOP, ROTM, CATLG, CHAR, HIS, PLAYER, CATLGN, PATRNS, PATRNN
DIMENSION RLINK4(1000), LLINK4(1000), DOWN4(1000), RLINK3(1000), DOWN3
1(1000), DOWN2(1000)
EQUIVALENCE (NAME4(4), RLINK4(3), LLINK4(2), DOWN4(1)), (NAME2(2), DOWN
12(1)), (NAME3(3), RLINK3(2), DOWN3(1))

(FIRST DECIDE IF NORTH'S RELIABILITY IS HIGH ENOUGH TO AFFORD
A NON-TRUTHFUL DECLARATION. IF NOT, USE A RANDOM NUMBER TO
DECIDE IF A FALSE DECLARATION SHOULD BE MADE ANYWAY. THIS
IS TO PREVENT BUILD-UP OF A PATTERN OF UNRELIABILITY.)

IF (NC.GT.2) GO TO 3
CALL RANDOM
IF (YRAND.GT.0.33) GO TO 1
CPN=STRATN
RETURN
1 IF (YRAND.GT.0.67) GO TO 2
CPN=2
RETURN
2 CPN=3
RETURN
3 IF (NC.GT.4) GO TO 10
IF (DEFES.GT.0.OR.BDEFES.GT.0) GO TO 4
CALL CLSMVN(MOV, INCNT, PCNTN, PCNTNM, INCNTR)
IF (PMOVEN.EQ.0) GO TO 5

```

```

25 CPN=PMOVEN
RETURN
10 TOPE=ES
IF (NAME3(TOPE).GE.40) GO TO 20
TOPE=DOWN3(TOPE)
IF (NAME3(ES).LE.NAME3(TOPE)) GO TO 15
TOPE=TOPE
TOPE=DOWN3(TOPE)
IF (NAME3(TOPE).GT.NAME3(TOPE)) GO TO 20
IF (GO AHEAD IF RELIABILITY INCREASING.)
TOPE=EN
TOPE=ES
CALL RANDOM
IF (YRAND.LT.0.3) GO TO 20
CPN=STRATN
RETURN
20 TOPE=EN
TOPE=ES
IF (PMOVES.GT.0) GO TO 21
NORS=1
CALL CLSTYP(NORS)
IF (PMOVEN.GT.0) GO TO 25
CALL RANDOM
IF (YRAND.LT.0.5) GO TO 6
IF (STRATN.EQ.1) CPN=3
IF (STRATN.EQ.2) CPN=2
IF (STRATN.EQ.3) GO TO 5
RETURN
END

SURROUTINE PENEGR(PROBLM,WHOCAL)
THIS SUBROUTINE IS CALLED UPON IF EITHER NORTH OR SOUTH WISH TO
RENEGOTIATE THE OTHER'S CONCESSION POINT CALLING FOR ARMING.

IMPLICIT INTEGER (A-W)
COMMON/ST/STRUE, LASTCD, NFLAG, TURN, ANAL, RECON
COMMON/JEC/VALUE(7,3), RTABN(3,3), RTARS(3,3), ATABN(3), ATAPS(2),
1SN, SS, WN, WS, RS, RW, NC, INRANC, YRAND, EFLAG, CPN, CPS, STRATN, STRATS,
2PNOLD, ISN, ISS, ECNXN, ECNXS, PN, PS, FACTXN, FACTXS
IF (WHOCAL.EQ.1) GO TO 100
(A BRANCH INDICATES THAT THE PLAYER ASKED FOR RENEGOTIATION,
NO BRANCH INDICATES THAT THE COMPUTER ASKED.)
GO TO (10,20,30), PROBLM
10 WRITE (6,11)
11 FORMAT (//, OUR STRENGTHS AND MOVES SO FAR HAVE INDICATED THAT WE

```

```

1 ARE CONDUCTING AN ARMS RACE. A RENEGOTIATION IS REQUESTED.')
  WRITE (6,12)
12 FORMAT (/,12) AFTER RENEGOTIATING, WHAT IS YOUR NEW CONCESSION PT?')
13 READ (5,13) CPS
13 FORMAT (11)
20 RETURN (6,21)
21 WRITE (6,21) IT IS REQUESTED THAT WE RENEGOTIATE. YOU HAVE ARMED
  1 OVER 50% OF THE MOVES, AND IF THIS KEEPS UP, WE WILL BOTH GO BROKE
  2 //.. CONDUCTING AN ARMS RACE.')
  WRITE (6,12)
  READ (5,13) CPS
  RETURN
30 WRITE (6,31)
31 FORMAT (//,12) IT IS REQUESTED THAT WE RENEGOTIATE. YOU ARE ALREADY
  1 QUITE STRONG, YET YOU HAVE ARMED AT LEAST TWO OUT OF THE LAST',
  2 //, THREE MOVES.')
  WRITE (6,12)
  READ (5,13) CPS
  RETURN
C 100 CALL MAXSTR(MAXGN)
C (THE PROGRAM ASKS FOR THE MAXIMUM GAIN STRATEGY IN ORDER TO
C APPEASE THE PLAYER, YET BENEFIT AS MUCH AS POSSIBLE.)
  STRATN=MAXGN
  CALL RANDOM
  IF (YRAND.GT.0.8.AND.STRATN.EQ.1) STRATN=2
  CALL RANDOM
  IF (YRAND.GT.0.67) GO TO 110
  IF (YRAND.GT.0.33) GO TO 120
  CPN=STRATN
  GO TO 40
110 CPN=2
  GO TO 40
120 CPN=3
  GO TO 40
40 WRITE (6,101) CPN
101 FORMAT (//,12) BASED ON YOUR REQUEST FOR RENEGOTIATION, I HAVE CONST
  1 DERED MY CONCESSION POINT. MY NEW CP IS TO:',12,')
  WHOICAL=0
  RETURN
  END
SUBROUTINE OPTSTR(STRATO)
THIS SUBROUTINE DETERMINES THE OPTIMUM STRATEGY BASED ON ZERO-SUM
TWO PERSON RECTANGULAR GAME THEORY. THE OPTIMUM STRATEGY IS
RETURNED BY THE PARAMETER STRATO.
C C C C

```

```

C
COMMON/JEC/VALUE(7,3),RTABN(3,3),RTABS(3,3),ATARN(3),ATARS(3),
1SN,SS,WN,WS,RS,RW,NC,INRAND,YRAND,EELAG,CN,CPS,STRATN,STRATS,
2PNOLD,ISN,ISS,ECNXN,ECNXS,PN,PS,EACTXN,EACTXS
DIMENSION Q(3),W(3)

THE SUBROUTINE FIRST DETERMINES IF A SADDLE POINT EXISTS.

DO 10 I=1,3
  Q(I)=MINO(RTABN(I,1),RTARN(1,2),RTABN(1,3))
10  W(I)=MAXO(RTABN(1,1),RTABN(2,1),RTABN(3,1))
  QMAX=MAXO(Q(1),Q(2),Q(3))
  WMIN=MINO(W(1),W(2),W(3))
  IF (QMAX.NE.WMIN) GO TO 30
  DO 11 I=1,3
  IF (QMAX.EQ.Q(I)) GO TO 13
11  CONTINUE
  WRITE (6,12)
12  FORMAT (//, ' S/R OPTSTR ERROR #1 *****')
  STOP
  DO 14 J=1,3
  IF (WMIN.EQ.W(J))GO TO 22
14  CONTINUE
  WRITE (6,15)
15  FORMAT (//, ' S/R OPTSTR ERROR #2 *****')

C
L=1
IF (RTABN(L,J).EQ.QMAX) GO TO 40
16  IF (L.EQ.3) GO TO 19
  K=L+1
  DO 18 L=K,3
  IF (QMAX.EQ.Q(L))GO TO 16
18  CONTINUE
  IF (J.EQ.3) GO TO 30
  J=J+1
  DO 20 L=J,3
  IF (WMIN.EQ.W(L))GO TO 21
20  CONTINUE
  GO TO 30
  J=L
  GO TO 22
40  STRATO=L
  RETURN

IF NO SADDLE POINT EXISTS, THE CONCEPT OF DOMINANCE IS USED TO
  ATTEMPT TO REDUCE THE SIZE OF THE MATRIX.
C

```

```

30 L=3
   I=1
   K=2
31 IF (RTABN(I,1).GE.RTABN(K,1).AND.RTABN(I,2).GE.RTABN(K,2).AND.
   1 RTABN(I,3).GE.RTABN(K,3)) GO TO 35
   GO TO 351
35 IF (RTABN(I,1).GE.RTABN(L,1).AND.RTABN(I,2).GE.RTABN(L,2).AND.
   1 RTABN(I,3).GE.RTABN(L,3)) GO TO 36
   GO TO (32,33,34),I
32 I=2
   K=1
   GO TO 31
33 I=3
   L=2
   GO TO 31
36 STRATO=I
   RETURN

```

THE STMT 34 BRANCH INDICATES THAT NO OPTIMUM STRATEGY CAN BE
EASILY DETERMINED. IN THIS CASE, THE S/R RETURNS A STRATEGY
RECOMMENDATION BASED ON STRENGTH.

```

34 IF (SN.LE.-2) GO TO 39
   STRATO=3
   RETURN
39 STRATO=1
   RETURN
   END

```

SUBROUTINE MAXSTR(MAXGN)

THIS SUBROUTINE DETERMINES THE MAXIMUM GAIN STRATEGY FROM THE
RELATIVE TABLE (RTABN) OF POINTS. THE SELECTED STRATEGY NUMBER
IS RETURNED BY PARAMETER MAXGN.

```

IMPLICIT INTEGER (A-W)
COMMON/JEC/VALUE(7,3), RTABN(3,3), RTARS(3,3), ATABS(3),
1 SN, SS, WN, WS, RS, RW, NC, INRAND, YRAND, EFLAG, CPN, CPS, STRATN, STRATS,
2 PNOLD, ISN, ISS, ECNXN, ECNXS, PN, PS, FACTXN, FACTXS
DIMENSION Q(3)

```

```

C
DO 12 I=1,3
12 Q(I)=MAXO(RTABN(I,1),RTABN(I,2),RTABN(I,3))
   MAXGN=MAXO(Q(1),Q(2),Q(3))
DO 14 I=1,3
14 IF (MAXGN.EQ.Q(I)) GO TO 13
   CONTINUE

```

```

1  WRITE (6,1)
   FORMAT (//, ' S/R MAXSTR ERROR *****' )
   RETURN
13  MAXGN=I
   RETURN
   END

```

SUBROUTINE MINSTR(MINLCS)

THIS SUBROUTINE DETERMINES THE STRATEGY FOR MINIMUM RISK OF LOSS OF POINTS RELATIVE TO THE OPPONENT. THE SELECTED STRATEGY IS RETURNED BY THE PARAMETER MINLCS.

```

IMPLICIT INTEGER (A-W)
COMMON/JEC/VALUE(7,3), RTABN(3,3), RTABS(3,3), ATABN(3), ATABS(3),
1  ISN, SS, WN, WS, RS, RW, NC, INRAND, YRAND, EFLAG, CPN, CPS, STRATN, STRATS,
2  PNOLD, ISN, ISS, ECNXN, ECNXS, PN, PS, FACTXN, FACTXS
DIMENSION Q(3)

```

```

C
DO 10 I=1,3
10  Q(I)=MINO(RTABN(I,1),RTABN(I,2),RTABN(I,3))
   QMAX=MAXO(Q(1),Q(2),Q(3))
   DO 12 I=1,3
   12  IF (QMAX.EQ.Q(I)) GO TO 13
   CONTINUE
   WRITE (6,1)
1  FORMAT (//, ' S/R MINSTR ERROR *****' )
   RETURN
13  MINLOS=I
   RETURN
   END

```

SUBROUTINE ECONMY

THIS SUBROUTINE DETERMINES IF THE STATE OF THE ECONOMY HAS CHANGED, AND IF SO, ATTEMPTS TO DETERMINE THE ECONOMIC CONDITIONS AND SET A FLAG (EFLAG) ACCORDINGLY.

THE ECONOMIC CONDITION IS RANDOMLY GENERATED AT RANDOM TIMES, BUT THE AMOUNT OF CHANGE TO COSTS IS NOT FACTORED INTO THE TABLES, AND HENCE IS UNKNOWN TO THE OPPONENTS (COMPUTER AND PLAYER).

```

IMPLICIT INTEGER (A-W)
COMMON/JEC/VALUE(7,3), RTABN(3,3), RTABS(3,3), ATABN(3), ATABS(3),
1  ISN, SS, WN, WS, RS, RW, NC, INRAND, YRAND, EFLAG, CPN, CPS, STRATN, STRATS,
2  PNOLD, ISN, ISS, ECNXN, ECNXS, PN, PS, FACTXN, FACTXS

```

```

C      PCHG=PNOLD+RTABN(STRATN,STRATS)+FACTXN-PN
C      IF (PCHG) 3,2,1
C      1  EFLAG=-10
C          (INDICATES COSTS ARE EXPENSIVE - A 900M)
C          RETURN
C      2  EFLAG=C
C          (INDICATES COSTS ARE NORMAL)
C          RETURN
C      3  EFLAG=10
C          (INDICATES COSTS ARE CHEAP - A BUST)
C          RETURN
C          END
C
C      SUBROUTINE ENPAT(PTRUTH)
C
C      THE PURPOSE OF THIS SUBROUTINE IS TO DETERMINE PATTERNS IN THE
C      EN (NORTH'S ESTIMATE OF SOUTH'S RELIABILITY) ARRAY, IN ORDER TO
C      PREDICT THE TRUTHFULNESS OF SOUTH'S DECLARATION.
C
C      PTRUTH=0 ==> SOUTH'S DECLARATION IS PROBABLY NOT TRUE.
C      PTRUTH=1 ==> SOUTH'S DECLARATION IS PROBABLY TRUE.
C
C      IMPLICIT INTEGER (A-W)
C      COMMON/JEC/VALUE(7,3),RTABN(3,3),RTABS(3,2),ATABN(3),ATABS(3),
C      1SN,SS,WN,WS,RS,RW,NC,INRAND,YRAND,EFLAG,CPS,STRATN,STRATS,
C      2PNOLD,ISN,ISS,ECN,N,ECN,N,PS,FACTXN,FACTXS
C      COMMON/ENP/ENARY(50),ESARRY(50)
C      COMMON/EEF/EN,ES,TOPEN,TOPE
C      COMMON/T/AVAIL4,AVAIL3,AVAIL2,NAME4(1000),NAME2(1000),NAME3(1000)
C      COMMON/TD/TOP,ROTM,CATLG,CHAR,HIST,PLAYER,CATLGN,PATRNS,PATRNN
C      EQUIVALENCE (NAME4(3),RLINK4(3)),LLINK4(2),DOWN4(1)),(NAME2(2),DOWN
C      12(1)),(NAME3(3),RLINK3(2),DOWN3(1))
C      DIMENSION RLINK4(1000),LLINK4(1000),DOWN4(1000),RLINK3(1000),DOWN3
C      1(1000),DOWN2(1000)
C
C      L=1
C      IF (NC.GT.4) GO TO 5
C      GO TO (1,2,3,4),NC
C      1  CALL RANDOM
C      IF (YRAND.GT.0.7) GO TO 101
C      (ASSUME=1
C      100  PTRUTH=1
C      GO TO 90

```

```

101 PTRUTH=0
   GO TO 99
2   IF (NAME3(EN).EQ.50) GO TO 100
   PTRUTH=0
   GO TO 99
3   IF (NAME3(EN).EQ.50) GO TO 101
   (IF HE HAS TOLD TRUTH THE 1ST 2 TIMES, ASSUME HE WILL NOT 3RD.)
   PTRUTH=1
   GO TO 99
4   IF (NAME3(EN).EQ.50) GO TO 101
   IF (ENARRY(2).EQ.50.AND.ENARRY(3).EQ.50) GO TO 101
   (ASSUME PATTERN OF 1-2-1 UNTRUTH/TRUTH/UNTRUTH.)
   PTRUTH=1
   GO TO 99
5   IF (NAME3(EN).EQ.50) GO TO 100
   (HAS HE ALWAYS TOLD TRUTH?)
   N=NC-1
   IF (ENARRY(N).EQ.50) GO TO 500
   (IF TOLD TRUTH LAST TIME, DETERMINE PATTERN.)
   DO 6 I=2,N
   IF (ENARRY(NC-I).EQ.50) GO TO 75
   (IF HE HAS EVER TOLD TRUTH, DETERMINE UNTRUTH PATTERN.)
   CONTINUE
   PTRUTH=0
   GO TO 99
75  J=NC-1
   DO 76 I=J,N
   IF (ENARRY(NC-I).LT.50) GO TO 77
   CONTINUE
   PTRUTH=1
   (HISTORY OF COMPLETE TRUTHFULNESS PRIOR LAST UNTRUTH, CAN NOT
   DETERMINE UNTRUTH PATTERN, ASSUME TRUTH.)
   GO TO 99
77  K=I-J
   (K IS TRUTH PATTERN.)
   DO 78 I=K,N
   IF (ENARRY(NC-I).EQ.50) GO TO 79
   CONTINUE
   PTRUTH=0
   (VERY POOR HONESTY DEMONSTRATED, ASSUME UNTRUTH.)
   GO TO 99
79  L=I-J-K+1
   (L IS UNTRUTH PATTERN.)
   (PATTERN IS APPROXIMATELY K/L/K TRUTH/UNTRUTH/TRUTH.)
   I=J-1
   IF (I.GE.L) GO TO 100

```

```

C      (IS PATTERN OF UNTRUTHS SATISFIED?)
C      PTRUTH=0
C      GO TO 55
C 500 DO 51 I=2,N
C      IF (ENARRY(NC-I).LT.50) GO TO 52
C 51 CONTINUE
C      PTRUTH=1
C      (ASSUME HE MUST BE COMPLETELY HONEST.)
C      GO TO 50
C 52 IF (I.EQ.N) GO TO 101
C      (ASSUME HE IS AT END OF PATTERN.)
C      J=I-1
C      (J IS NUMBER OF TRUTHS IN THIS SERIES.)
C      I=I+1
C      DO 53 L=I,N
C 53 IF (ENARRY(NC-L).LT.50) GO TO 54
C      CONTINUE
C      PTRUTH=1
C      (HISTORY OF COMPLETE TRUTHFULNESS PRIOR LAST UNTRUTH ==> TRUTH.)
C      GO TO 55
C 54 K=L-1
C      (K IS TRUTH PATTERN LENGTH.)
C      IF (J.GE.K) GO TO 101
C      (IS PATTERN OF TRUTHS SATISFIED?)
C      PTRUTH=1
C 90 CONTINUE
C      RETURN
C      END

```

SUBROUTINE CATLOG

```

*****
*      THE TWENTY-FIVE SUBROUTINES WITHIN THIS SECTION.
*      SUPPORT THE LEARNING CAPABILITIES OF THE PROGRAM.
*      *****

```

```

SUBROUTINE TO BUILD THE LIBRARY OF CATALOGUE OF PLAYERS.
IMPLICIT INTEGER (A-W)
COMMON/T/AVAIL4,AVAIL3,AVAIL2,NAME4(1000),NAME2(1000),NAME3(1000)
COMMON/TD/TOP,BOTM,CATLG,CHAR,HIST,PLAYER,CATLGN,PATRNS,PATRN

```

```

DIMENSION RLINK4(1000),LLINK4(1000),DOWN4(1000),PLINK3(1000),DOWN2
1(1000),DOWN2(1000)
EQUIVALENCE (NAME4(4),RLINK4(3),LLINK4(2),DOWN4(1)),(NAME2(2),DOWN
12(1)),(NAME3(3),RLINK3(2),DOWN3(1))
IF(CATLGN.EQ.0) GOTO 101
CATLGN=CATLGN+1
CALL GET3(L)
C FIRST PLAYER CELL
TOP=L
DOWN3(TOP)=0
ROTM=TOP
CATLG=TOP
NAME3(TOP)=PLAYER
GOTO 50
101 CALL SEARCH(PLY)
C CHECK TO SEE IF LISTED. IF IT IS MOVE PLAYER TO TOP OF CATLG.
IF(PLY.EQ.1)GOTO 200
C CHECK FOR 10 PLAYERS IN THE CATALOGUE. IF THERE IS DELETE THE
C BOTTOM PLAYER.
IF(CATLGN.EQ.10)GOTO 300
C UPDATE THE NUMBER OF PLAYERS IN THE CATALOGUE COUNTER
CATLGN=CATLGN+1
C GET ADDITIONAL PLAYER CELL
400 TOP=CATLG
CALL GET3(L)
DOWN3(L)=TOP
TOP=L
CATLG=TOP
NAME3(TOP)=PLAYER
C BUILD ADDITIONAL CELLS FOR EXPERIENCE LIBRARY ON NEW PLAYER.
50 CALL GET4(J)
C CELL FOR CHARACTERISTICS HEADER.
RLINK3(TOP)=J
CHAR=J
C CELL FOR AGGRESSIVENESS HEADER
CALL GET2(K)
RLINK4(CHAR)=K
AGGRS=K
NAME2(AGGRS)=0
DOWN2(AGGRS)=0
C CELL FOR PARAMETERS
CALL GET4(J)

```

```

LLINK4(CHAR)=J
PARAM=J
NAME4(PARAM)=0
RLINK4(PARAM)=0
LLINK4(PARAM)=0
DOWN4(PARAM)=0

C
C
C
CALL GET4(J)
CELL FOR HISTORY HEADER

DOWN4(CHAR)=J
HIST=J
NAME4(HIST)=0
RLINK4(HIST)=0
LLINK4(HIST)=0
DOWN4(HIST)=0

C
C
C
CELL FOR RELIABILITY REFERENCE

CALL GET2(K)
NAME4(CHAR)=K
ENPTR=K
NAME2(ENPTR)=0
DOWN2(ENPTR)=0
RETURN

C
C
C
GET SUBROUTINE TO DELETE A PLAYER FROM THE BOTTOM OF CATALOGUE.
300 CALL DELETP
GOTO 400

C
C
C
GET SUBROUTINE TO MOVE EXISTING PLAYER TO THE TOP OF THE CATALOGUE
200 CALL MOVEPL

C
C
C
RETURN
END

SUBROUTINE DELETP
SUBROUTINE TO DELETE A PLAYER FROM THE BOTTOM OF THE CATALOGUE.
C
C
IMPLICIT INTEGER (A-W)
COMMON/T/AVAIL4,AVAIL3,AVAIL2,NAME4(1000),NAME2(1000),NAME3(1000)
COMMON/TD/TOP,BOTM,CATLG,CHAR,HIST,PLAYER,CATLGN,PATRNS,PATRNN
COMMON/TDP3/TYDE(20),INIT(20)
COMMON/TDP4/SEQ(20),SCNTR

```

```

DIMENSION RLINK(1000), LLINK4(1000), DOWN4(1000), RLINK3(1000), DOWN3
1(1000), DOWN2(1000), PLINK4(3), LLINK4(2), DOWN4(1), (NAME2(2), DOWN
EQUIVALENCE (NAME4(4), PLINK4(3), LLINK4(2), DOWN3(1))
12(1)), (NAME3(3), PLINK3(2), DOWN3(1))
IF(RLINK3(BOTM).EQ.0) GOTO 10

RETURN CELL TO AVAIL2 LIST FROM ENPTR

CHAR=PLINK3(BOTM)
K2=NAME4(CHAR)
DOWN2(K2)=RLINK4(CHAR)
DOWN2(RLINK4(CHAR))=AVAIL2
AVAIL2=K2

RETURNS CELLS TO AVAIL4 FROM CHAR AND HIST.

HIST=DOWN4(CHAR)
CALL DSEQLB
PARAM=LLINK4(CHAR)
DOWN4(HIST)=PARAM
DOWN4(PARAM)=AVAIL4
AVAIL4=CHAR
TOP=CATLG

10 RETURNS PLAYER CELL TO AVAIL3

11 IF(DOWN3(TOP).EQ.BOTM) GOTO 12
TOP=DOWN3(TOP)
GO TO 11

12 DOWN3(BOTM)=AVAIL3
AVAIL3=BOTM
BOTM=TOP
DOWN3(BOTM)=0
RETURN
END

SURROUTINE INTLIB(INCNT, MOVE, TOPI, INCNTR, MOV)
INITIAL STRATEGY LIBRARY. THIS SUBROUTINE SETS UP CELLS FOR THE
INITIAL STRATEGY LIBRARY OF CELLS.

IMPLICIT INTEGER (A-W)
COMMON/JEC/VALUE(7,3), RTABN(3,3), RTABS(3,3), ATARN(3), ATARS(3),
1SN, SS, WN, WS, RS, RW, NC, INRAND, YRAND, EFLAG, CPN, CPS, STRATN, STRATS,
2PNOLD, ISN, ISS, ECNXN, ECNXS, PN, PS, EACTXN, EACTXS
COMMON/T/AVAIL4, AVAIL3, AVAIL2, NAME4(1000), NAME2(1000), NAME3(1000)
COMMON/TO/TOPT, BOTM, CATLG, CHAR, HIST, PLAYER, CATLGN, PATRNS, PATRNN

```

```

COMMON/TDP3/TYPE(20),INIT(20)
DIMENSION RLINK4(1000),LLINK4(1000),DOWN4(1000),RLINK3(1000),DOWN3
1(1000),DOWN2(1000)
EQUIVALENCE (NAME4(4),RLINK4(3),LLINK4(2),DOWN4(1)),(NAME2(2),DOWN
12(1)),(NAME3(3),RLINK3(2),DOWN3(1))
DATA A1/A1//,A2/A2//

"INCNT" IS THE COUNT OF CELLS IN THE LIBRARY. "INCNT" IS THE
ALLOWABLE NUMBER OF CELLS IN THE LIBRARY. (IE. INITIALLY 10 CELLS
WERE IN THE LIBRARY. "MOV" INDICATES WHETHER THE INITIAL
MOVES ARE IN THE LIBRARY OR IF A NEW CELL IS REQUIRED.

IF (INCNT.EQ.INCNT) GOTO 10
INCNT=INCNT+1
CALL GET4(J)
INIT(INCNT)=J
TOPI=J
RETURN
10 CALL DINTLB(INCNT,MOVE,TOPI)
RETURN
END

SUBROUTINE DINTLB(INCNT,MOVE,TOPI)
SUBROUTINE TO DELETE OR COMBINE SIMILAR ELEMENTS IN
THE INITIAL MOVE LIBRARY.

IMPLICIT INTEGER (A-W)
COMMON/JEC/VALUE(7,3),RTARN(3,3),RTABS(3,3),ATARN(3),ATARS(3),
1SN,SS,WN,WS,RS,RW,NC,INRAND,YRAND,EFLAG,CN,CPS,STRAIN,STRATS,
2PNOLD,ISN,ISS,ECNXN,FCNXS,PN,PS,FACITXN,FACITXS
COMMON/T/AVAIL4,AVAIL3,AVAIL2,NAME4(1000),NAME2(1000),NAME3(1000)
COMMON/TDP/ROTM,ROTM,CATLG,CHAR,HIST,PLAYER,CATLG,PTATNS,PTATNN
COMMON/TDP/MOVES,MOVEN
COMMON/TDP3/TYPE(20),INIT(20)
DIMENSION RLINK4(1000),LLINK4(1000),RLINK3(1000),DOWN3
1(1000),DOWN2(1000)
EQUIVALENCE (NAME4(4),RLINK4(3),LLINK4(2),DOWN4(1)),(NAME2(2),DOWN
12(1)),(NAME3(3),RLINK3(2),DOWN3(1))
DATA A1/A1//,A2/A2//
SUBROUTINE TO DELETE OR COMBINE INITIAL MOVES LIBRARY.
IF (MOVE.EQ.MOVEN) GOTO 20
IF (NAME4(HIST).EQ.0) GOTO 11
TOPI=NAME4(HIST)
VAL20=TOPI
RUNTCP=TOP
8 IF (RUNTCP.EQ.BOTM) GOTO 13
15 IF (RUNTOP.EQ.BOTM) GOTO 13

```

```

13  RUNTOP=DOWN3(RUNTOP)
    CHARR=RLINK3(RUNTOP)
    HISTR=DOWN4(CHARR)
    IF(MOVE.EQ.MOVEN) GOTO 22
    IF(NAME4(HISTR).EQ.VAL20) GOTO 14
    GOTO 15
    TOPMN=MOVE
    NAME4(TOPI)=NAME2(TOPMN)
    TOPMN=DOWN2(TOPMN)
    RLINK4(TOPI)=NAME2(TOPMN)
    TOPMN=DOWN2(TOPMN)
    LLINK4(TOPI)=NAME2(TOPMN)
    DOWN4(TOPI)=0
    RETURN
14  NAME4(HISTR)=0
    GOTO 15
11  MCNT=1
12  TOPI=INIT(MCNT)
    IF(MCNT.EQ.INCNT) GOTO 100
    RUNTOP=TOP
    VAL20=TOPI
    GOTO 15
20  IF(RLINK4(HISTR).EQ.0) GOTO 21
    TOPI=RLINK4(HISTR)
    GOTO 8
21  MCNT=2
    GOTO 12
22  IF(RLINK4(HISTR).EQ.VAL20) GOTO 24
    GOTO 15
24  RLINK4(HISTR)=0
    GOTO 15
100 RETURN
    END

```

```

CCCC
SUBROUTINE TYPELB(INCNTR,MON,INCNT)
SUBROUTINE TO BUILD NEW STRUCTURES IN THE TYPE LIBRARY AT THE
END OF A GAME.
IMPLICIT INTEGER (A-W)
COMMON/TDP7/GROUPS,GRUOIPN,TMOVES,TMOVEN,MCNT,CONTR
COMMON/T/AVAIL4,AVAIL3,AVAIL2,NAME4(1000),NAME2(1000),NAME3(1000)
COMMON/TD/TOP,RTI,CATLG,CHAR,HIST,PLAYER,CATLGN,PATRNS,PATRNN
COMMON/TDP4/SEQ(20),SCNTR
COMMON/TDP3/TYPE(20),INIT(20)
COMMON/TDP8/RLH,ST,NAH,ST
COMMON/CLS/DEFS,BOFFS,OFFEN,BOFFEN,PMOVEN,PMOVES

```

```

COMMON/TPJEC/DONOR,DOSOU,CKCNT1,CKCNT2
COMMON/JEC/VALUE(7,3),RTABN(3,3),RTABS(3,3),ATABS(3),
1SN,SS,WN,WS,RS,RW,NC,INRAND,YRAND,CBN,CPS,STRATN,STRATS,
2PNOLD,ISN,ISS,ECNXN,ECNXP,PS,FACTXN,FACTXS
DIMENSION RLINK4(1000),LLINK4(1000),DOWN4(1000),DOWN2
1(1000),DOWN2(1000)
EQUIVALENCE (NAME4(4),RLINK4(3),LLINK4(2),DOWN4(1)),(NAME2(2),DOWN
12(1)),(NAME3(3),RLINK3(2),DOWN2(1))
CATA A1/A1//,A2/A2//
CKCNT1=0
CKCNT2=0
DOSOU=0
DONOR=0
NAME4(HIST)=NAHIST
RLINK4(HIST)=RLHIST
1E(NC,LT,10)GOTO 101
IF(CONTR.EQ.INCNTR) GOTO 100
3STRTP=LLINK4(PATRNS)
8TOPGP=DOWN4(STRTP)
RTOP=TOPGP
CONT=1
IF(CONT.EQ.INCNTR) GOTO 500
9IF(DOWN2(RTOP).EQ.0)GOTO 500
RTOP=DOWN2(RTOP)
CONT=CONT+1
GOTO 9
500CONTR=CONTR+1
CALL GET3(L)
TYPE(CONTR)=L
TOPSN=L
RTOP1=RTOP
6IF(DOWN2(RTOP1).EQ.0) GOTO 7
RTOP1=DOWN2(RTOP1)
GOTO 6
7IF(DOWN2(RTOP).EQ.0)GOTO 4
DOWN2(RTOP1)=AVAIL2
AVAIL2=DOWN2(RTOP)
4DOWN2(RTOP)=0
DOWN3(TOPSN)=TOPGP
DOWN4(STRTP)=0
CHECK TO SEE IF BOTH NORTH AND SOUTH TYPE LIBRARY MEMORY IS SETUP.
IF(STP TP.EQ.LLINK4(PATRNN))GOTO 50
TOPS=L
2IF(DONOR)2,2,102
STRTP=LLINK4(PATRNN)
GOTO 8
50TOPN=L
51IF(WON)11,12,13

```

```

11  NAME3(TOPS)=0
    RLINK3(TOPS)=TOPN
    NAME3(TOPN)=TOPS
    RLINK3(TOPN)=0
    GOTO 40
12  NAME3(TOPS)=0
    RLINK3(TOPS)=0
    TOPN=TOPS
    DOWN2(RTOP)=AVAIL2
    AVAIL2=DOWN3(TOPSN)
    DOWN3(TOPSN)=AVAIL3
    AVAIL3=TOPSN
    INITN=RLHIST
    DOWN4(INITN)=AVAIL4
    AVAIL4=INITN
    INCNT=INCNT-1
    CONTR=CONTR-1
    GOTO 40
13  NAME3(TOPN)=0
    RLINK3(TOPN)=TOPS
    NAME3(TOPS)=TOPN
    RLINK3(TOPS)=0
    INITN=NAME4(HIST)
    DOWN4(INITN)=TOPS
    INITN=RLINK4(HIST)
    DOWN4(INITN)=TOPN
    RETURN
100 CALL DTYPLR
    IF(CKCN+1)333,331,331
333 DO,CCU=-1
    GOTO 330
331 TOPS=DOWN4(NAME4(HIST))
330 IF(CKCN=-1)
334 DONOR=-1
    GOTO 337
335 DONOR=1
337 IF(DONOR)3,200,200
200 IF(DONOR)2,102,102
335 WRITE(6,401)
401 FORMAT(5X,'THERE IS A MISTAKE IN DELETION OF A TYPE LIBRARY.')
102 TOPN=DOWN4(RLINK4(HIST))
    GOTO 51
101 RETURN
    END

```

```

C
C
C
SUBROUTINE DTYPLB
SUBROUTINE TO DELET OR COMBINE STRUCTURES IN THE
TYPE LIBRARY.
IMPLICIT INTEGER (A-W)
COMMON /TP4/SEQ(20),SCONT2
COMMON /TP3/TYPE(20),INIT(20)
COMMON /JEC/VALUE(7,3),RTARN(3,3),PTARS(3,3),ATARN(3),ATARS(2),
1  ISN,SS,WN,WS,RS,RSRW,NC,INRAND,YRAND,FELAG,CPN,CPS,STRATN,STRATS,
2  PNOLD,ISN,ISS,ECNXN,ECNXS,PN,PS,FAC TXN,FAC TXS
COMMON /CL$/DEFES,RDEFES,OFFEN,ROFFN,PMOVEN,MCNT,CONTR
COMMON /DP7/GRROUPS,GRROUPN,TMOVES,TMGVEN,NAME2(1000),NAME3(1000)
COMMON /T/AVAIL4,AVAIL3,AVAIL2,NAME4(1000),NAME2(1000),NAME3(1000)
COMMON /TD/TOP,ROTM,CATLG,CHAR,HIST,PLAYER,CATLGN,PATRNN
COMMON /TDP/MOVES,MOVEN
COMMON /TPJEC/DONCR,DOSOU,CKCNT1,CKCNT2
COMMON /RLINK4/RLINK4(1000),LLINK4(1000),DOWN4(1000),DOWN3
1  (1000),DOWN2(1000)
COMMON /EQU/VALENCE (NAME4(4),RLINK4(3),LLINK4(2),DOWN3(1))
2  (1), (NAME3(3),RLINK3(2),DOWN3(1))
DATA A1/A1/A2/A2/A2/
PATRN=PATRNS
TOPI=NAME4(HIST)
1  FLAG=1
2  CONT=1
CONT2=1
STRTP=LINK4(PATRN)
MOVE=DOWN3(TYPE(CONT))
TOPM=DOWN2(TOPM).EQ.0) GOTO 10
101 IF(DOWN2(MOVE).EQ.0) GOTO 10
IF(DOWN2(MOVE).EQ.0) GOTO 10
CKGRP1=NAME2(TOPM)
CKGRP2=NAME2(MOVE)
CALL CKTYP(CKGRP1,CKGRP2,NOGO,FLAG)
IF(NOGO.EQ.0) GOTO 20
CONT2=CONT2+1
FLAG=0
20 IF(CONT2.EQ.4) GOTO 30
TOPM=DOWN2(TOPM)
MOVE=DOWN2(MOVE)
FLAG=1
10 GOTO 101
DOWN4(TOP1)=TYPE(CONT)
IF(PATRN.EQ.PATRNN) GOTO 880
PATRN=PATRNN
TOPI=RLINK4(HIST)
GOTO 2

```

```

30 IF(CONT.EQ.CONTR) GOTO 40
   CONT=CONT+1
   GOTO 2
40 CONT=1
   MCNT=1
   TCPT=TYPE(CONT)
   TOPI=INIT(MCNT)
41 IF(DOWN4(TOPI).EQ.TOPT) GOTO 60
   IF(MCNT.EQ.INCNT) GOTO 61
   MCNT=MCNT+1
   GOTO 41
60 IF(PATR.N.EQ.PATR.NN) GOTO 45
   RTOP=CATLG
   RTOP1=RLINK3(RTOP)
   TOPD=NAME4(DOWN4(RTOP1))
44 IF(TOPI.EQ.TOPD) GOTO 46
42 IF(RTOP.EQ.RTM) GOTO 47
   RTOP=DOWN3(RTOP)
   RTOP1=RLINK3(RTOP)
   GOTO 44
46 IF(PATR.N.EQ.PATR.NN) GOTO 43
   NAME4(DOWN4(RTOP1))=0
43 GOTO 42
   RLINK4(DOWN4(RTOP1))=0
47 DOWN4(TOPI)=AVAIL4
   AVAIL4=TOPI
   CNTI1=MCNT
   CNTI2=CNTI1+1
38 IF(CNTI1.EQ.INCNT) GOTO 37
   INIT(CNTI1)=INIT(CNTI2)
   CNTI1=CNTI1+1
   CNTI2=CNTI2+1
   GOTO 38
37 INCNT=INCNT-1
   IF(PATR.N.EQ.PATR.NN) GOTO 61
45 RTOP=CATLG
   RTOP1=RLINK3(RTOP)
   TOPD=RLINK4(DOWN4(RTOP1))
   GOTO 44
61 RTOP=DOWN3(TOPT)
   RTM=RTOP
63 IF(DOWN2(RTM).EQ.0) GOTO 62
   RTM=DOWN2(RTM)
   GOTO 63
62 DOWN2(RTM)=AVAIL2
   AVAIL2=RTOP
   DOWN3(TOPT)=AVAIL3

```

```

        AVAIL3=TOPT
        MCT1=CONT
        MCT2=MCT1+1
        IF(MCT1.EQ.CONTR) GOTO 57
        IF(MCT2.EQ.CONTR) GOTO 57
        MCT1=MCT1+1
        MCT2=MCT2+1
        GOTO 620
57      CONTR=CONTR-1
        IF(PATRN.EQ.PATRNN) GOTO 570
        CKCNT1=-1
        GOTO 572
570     CKCNT2=-1
572     SCNT=1
58      TOPS=EQ(SCNT)
        DTOPS=TOPS
        DTOP1=DTOPS
        DTOP2=DOWN3(DTOPS)
        TOPT2=NAME3(DTOPS)
        IF(TOPT.EQ.TOPT2) GOTO 64
        DTOP1=DTOPS
        IF(DOWN3(DTOPS).EQ.0) GOTO 65
        DTOPS=DOWN3(DTOPS)
        GOTO 59
65      IF(SCNT.EQ.SCONTR) GOTO 66
        SCNT=SCNT+1
        GOTO 58
64      IF(DOWN3(DTOPS).EQ.0) GOTO 67
        IF(DTOPS.EQ.TOPS) GOTO 68
        DOWN3(DTOP1)=DTOP2
        DOWN3(DTOPS)=RLINK3(DTOPS)
        DOWN3(RLINK3(DTOPS))=AVAIL3
        AVAIL3=DTOPS
        DTOPS=DTOP2
        GOTO 59
68      RTP=DTOP2
69      IF(DOWN3(DTOP2).EQ.0) GOTO 70
        RTP=DTOP2
        DTOP2=DOWN3(DTOP2)
        GOTO 69
70      NAME3(DTOPS)=NAME3(DTOP2)
        NAME3(RLINK3(DTOPS))=NAME3(RLINK3(DTOP2))
        RLINK3(RLINK3(DTOPS))=RLINK3(RLINK3(DTOP2))
        DOWN3(RLINK3(DTOPS))=DOWN3(RLINK3(DTOP2))
        DOWN3(RLINK3(DTOP2))=AVAIL3
        DOWN3(DTOP2)=RLINK3(DTOP2)
        AVAIL3=DOWN3(RTP)

```

```

        DOWN3(RTP)=0
        GO TO 59
        IF(DTOPS.EQ.TOP5) GO TO 71
        DOWN3(DTOPS)=RLINK3(DTOPS)
        DOWN3(RLINK3(DTOPS))=AVAIL3
        AVAIL3=DOWN3(DTOP1)
        DOWN3(DTOP1)=0
        GO TO 65
        RUNT=CATLG
        CHARR=RLINK3(RUNT)
        HISTR=DOWN4(CHARR)
        IF(TOPS.EQ.LLINK4(HISTR)) GO TO 710
        IF(TOPS.EQ.DOWN4(HISTR)) GO TO 720
        IF(RUNT.EQ.BOTM) GO TO 730
        RUNT=DOWN3(RUNT)
        GO TO 72
        LLINK4(HISTR)=0
        GO TO 7200
        DOWN4(HISTR)=0
        GO TO 73
        DOWN3(DTOPS)=RLINK3(DTOPS)
        DOWN3(RLINK3(DTOPS))=AVAIL3
        AVAIL3=DTOPS
        CT1=SCNT+1
        CT2=SCNT+1
        IF(CT1.EQ.SCONTR) GO TO 80
        SEQ(CT1)=SEQ(CT2)
        IF(CT2.EQ.SCONTR) GO TO 80
        CT1=CT1+1
        CT2=CT2+1
        GO TO 731
        SCONTR=SCONTR-1
        GO TO 65
        CKCNT2=1
        RETURN
        IF(PATRN.EQ.PATRNN) GO TO 660
        PATRN=PA:RNN
        GO TO 2
        RETURN
        END

```

SUBROUTINE SEQLB(INCNTR,WON)

SUBROUTINE TO BUILD THE SEQUENCE LIBRARY.

IMPLICIT INTEGER (A-W)

COMMON/T/AVAIL4,AVAIL3,AVAIL2,NAME4(1000),NAME2(1000),NAME3(1000)

C
C
C

```

COMMON/TD/TOP, BOTM, CATLG, CHAR, HIST, PLAYER, CATLGN, PATRNS, PATRNN
COMMON/TDP2/STK(10), STKM(10), STKN(10), STKNM(10)
COMMON/TDP3/TYPE(20), INIT(20)
COMMON/TDP4/SEQ(20), SCNTR, TMOVES, TMOVEN, MCNT, CCNTP
COMMON/TDP7/GROUPS, GOUPT, TMOVES, TMOVEN, PMOVEN, PMOVES
COMMON/CLS/DEFS, BDEFS, OFFEN, ROFFEN, PMOVEN, PMOVES
COMMON/JEC/VALUE(7,3), PTABN(3,3), PTABS(3,3), ATARN(3), ATARS(3),
1SN, SS, WN, WS, RS, RW, NC, INRND, YRND, EFLAG, CPN, CPS, STRATN, STRATS,
2PNOLD, ISN, ISS, ECNXX, ECNXS, PN, PC, FACTXN, EACTXS
DIMENSION RLINK4(1000), LLINK4(1000), DCWN4(1000), RLINK3(1000), DOWN2
1(1000), DOWN2(1000)
1EQUIVALENCE (NAME4(4), PLINK4(3), LLINK4(2), DCWN4(1)), (NAME2(2), DOWN
12(1)), (NAME3(3), RLINK3(2), DOWN3(1))
DATA A1/A1/, A2/A2/,
PASN=-1
IF(NC.LT.10)GOTO 401
CNT=1
PSN1=0
SCNTP=INCNTR+10
IF(SCNTR.GT.SCNTR) GOTO 100
IF(LLINK4(HIST).EQ.0)GOTO 399
TOPSQ=LLINK4(HIST)
INIT=NAME4(HIST)
RTOP=TOPSQ
TYPE=DOWN4(INITS)
11 IF(NAME3(RTOP).EQ.TYPENO) GOTO 400
IF(DOWN2(RTOP).EQ.0) GOTO 10
RTOP=DOWN3(RTOP)
CNT=CNT+1
GOTO 11
IF(CNT.GE.INCNTP) GOTO 1010
CALL GET3(L)
DOWN3(RTOP)=L
RTOP=L
DOWN2(RTOP)=0
NAME3(RTOP)=TYPENO
CALL GET3(L)
RLINK3(RTOP)=L
IMOVES=L
NAME3(IMOVES)=NAME4(INITS)
RLINK3(IMOVES)=PLINK4(INITS)
DOWN2(IMOVES)=LLINK4(INITS)
GOTO 400
395 INIT=NAME4(HIST)
TYPENO=DOWN4(INITS)
GOTO 104
400 INIT=PLINK4(HIST)

```

```

103      TYPE NO=DOWN4(INITS)
          IF(DOWN4(HIST).EQ.0)GOTO 104
          IF(PASN1)IC1,103,101
          TOPSO=DOWN4(HIST)
          INITS=RLINK4(HIST)
          CNT=1
          PASN=9999
          PASN1=1
          GOTO 105
104      IF(LLINK4(HIST).EQ.0) GOTO 402
          TOPSO=LLINK4(HIST)
          CALL GET3(L)
          SCNTR=SCNTR+1
          SEQ(SCNTR)=L
          DOWN4(HIST)=L
          CNT=INCNTR
          PASN=9999
          GOTO 106
402      CALL GET3(L)
          SCNTR=SCNTR+1
          SEQ(SCNTR)=L
          LLINK4(HIST)=L
          CNT=INCNTR
          PASN=0
          GOTO 106
1010     PASN=-1
1011     IF(PASN)1,2,3
1012     GOTO IC4
1013     RETURN DSEQLB
100      CALL DSEQLB
401     GOTO 14
        RETURN
        END

```

SUBROUTINE DSEQLB

SUBROUTINE TO DELETE THE SEQUENCE STRUCTURES OF A PLAYER
AND THE SEQUENCE STRUCTURES FOR THE ASSOCIATED COMPUTER
PATTERNS OF PLAY.

CCCCC

```

IMPLICIT INTEGER (A-W)
COMMON/T/AVAIL4,AVAIL3,AVAIL2,NAME4(1000),NAME2(1000),NAME3(1000)
COMMON/TD/TOP,RTM,CATLG,CHAR,HIST,PLAYER,CATLGN,PATRNN
COMMON/TDP2/STK(10),STKM(10),STKN(10),STKNM(10)
COMMON/TDP4/SEQ(20),SCNTR

```



```

DIMENSION RLINK4(1000),LLINK4(1000),DOWN4(1000),RLINK3(1000),DOWN3
1(1000),DOWN2(1000)
EQUIVALENCE (NAME4(4),RLINK4(3),LLINK4(2),DOWN4(1)),(NAME2(2),DOWN
12(1)),(NAME3(3),RLINK3(2),DOWN3(1))
DATA A1/A1,A2/A2,
PATRNS=1
PATRNN=1
CALL GET4(J)
GET HEADER CELL FOR PATERNN MEMORY OF SOUTH.
PATRNS=J
TOPS=J
NAME4(TOPS)=0
RLINK4(TOPS)=0
GET HEADER CELL FOR GROUP MOVES THIS GAME FOR SOUTH
CALL GET4(J)
LLINK4(TOPS)=J
STRTP=J
NAME4(STRTP)=0
RLINK4(STRTP)=0
LLINK4(STRTP)=0
DOWN4(TOPS)=0
CALL GET4(J)
GET HEADER CELL FOR PATTERN MEMORY OF NORTH.
PATRNN=J
TOPN=J
NAME4(TOPN)=0
RLINK4(TOPN)=0
GET HEADER CELL FOR GROUP MOVES THIS GAME FOR NORTH
CALL GET4(J)
LLINK4(TOPN)=J
STRTPN=J
NAME4(STRTPN)=0
RLINK4(STRTPN)=0
LLINK4(STRTPN)=0
DOWN4(TOPN)=0
RETURN
END

```

CCC

CCC

CCC

CCC

```

CCCC
SUBROUTINE DPATNS
SUBROUTINE TO DELETE THE REMAINING NODES IN TEMPORARY
MEMORY AT THE END OF THE GAME.
IMPLICIT INTEGER (A-W)
COMMON/T/AVAIL2,AVAIL3,AVAIL4,NAME2(1000),NAME3(1000)
COMMON/ID/TOP,BOTM,CATLG,CHAR,HIST,PLAYER,CATLGN,PATRNS,PATRNN
COMMON/TDP2/STK(10),STKM(10),STKNM(10)
DIMENSION RLINK4(1000),LLINK4(1000),DOWN4(1000),DOWN3
1(1000),DOWN2(1000)
EQUIVALENCE (NAME4(4),RLINK4(3),LLINK4(2),DOWN3(1))
12(1)),(NAME3(3),RLINK3(2),DOWN3(1))
WRITE(6,5)
FORMAT(1,DPATNS '/')
PATRNS=PATRNS
STRTP=LLINK4(PATRN)
IF(DOWN4(STRTP).EQ.0)GOTO 10
TOPGP=DOWN4(STRTP)
IF(DOWN2(TOPGP).EQ.0)GOTO 11
TOPGP=DOWN2(TOPGP)
GOTO 12
DOWN2(TOPGP)=AVAIL2
AVAIL2=DOWN4(STRTP)
DOWN4(STRTP)=0
IF(DOWN4(PATRN).EQ.0)GOTO 13
MOVE=DOWN4(PATRN)
IF(DOWN2(MOVE).EQ.0)GOTO 14
MOVE=DOWN2(MOVE)
GOTO 15
DOWN2(MOVE)=AVAIL2
AVAIL2=DOWN4(PATRN)
DOWN4(PATRN)=STRTP
DOWN4(STRTP)=AVAIL4
AVAIL4=PATRN
13 IF(PATRN.EQ.PATRNN)GOTO 55
PATRN=PATRNN
GOTO 20
RETURN
55 END

```

```

SUBROUTINE PATUPD
SUBROUTINE TO UPDATE TEMPORARY MEMORY OF BOTH NORTH (PATRNN)
AND SOUTH (PATRNS)
CCCCC

```

```

CCCC
SUBROUTINE DPATNS
SUBROUTINE TO DELETE THE REMAINING NODES IN TEMPORARY
MEMORY AT THE END OF THE GAME.
IMPLICIT INTEGER (A-W)
COMMON/T/AVAIL4,AVAIL3,AVAIL2,NAME4(1000),NAME2(1000),NAME3(1000)
COMMON/TP/TP,BOBTH,CATLG,CHAR,HIST,PLAYER,CATLGN,PATRNS,PATRNN
COMMON/TP2/STK(10),STKM(10),STKN(10),STKNM(10)
DIMENSION RLINK4(1000),LLINK4(1000),DOWN4(1000),RLINK3(1000),DOWN3
1(1000),DOWN2(1000)
EQUIVALENCE (NAME4(4),RLINK4(3),LLINK4(2),DOWN4(1)),(NAME2(2),DOWN
12(1)),(NAME3(3),RLINK3(2),DOWN3(1))
WRITE(6,5)
FORMAT(1,DPATNS '/')
PATRNN=PATRNS
STRTP=LLINK4(PATRN)
IF(DOWN4(STRTP).EQ.0)GOTO 10
TOPGP=DOWN4(STRTP)
IF(DOWN2(TOPGP).EQ.0)GOTO 11
TOPGP=DOWN2(TOPGP)
GOTO 12
DOWN2(TOPGP)=AVAIL2
AVAIL2=DOWN4(STRTP)
DOWN4(STRTP)=0
IF(DOWN4(PATRN).EQ.0)GOTO 13
MOVE=DOWN4(PATRN)
IF(DOWN2(MOVE).EQ.0)GOTO 14
MOVE=DOWN2(MOVE)
GOTO 15
DOWN2(MOVE)=AVAIL2
AVAIL2=DOWN4(PATRN)
DOWN4(PATRN)=STRTP
DOWN4(STRTP)=AVAIL4
AVAIL4=PATRN
IF(PATRN.EQ.PATRNN)GOTO 55
PATRNN=PATRN
GOTO 20
RETURN
END
55

```

```

SUBROUTINE PATUPD
SUBROUTINE TO UPDATE TEMPORARY MEMORY OF BOTH NORTH (PATRNN)
AND SOUTH (PATRNS)
CCCCC

```



```

CALL TOTMV(TOTAL, GROUP)
STRTP=LLINK4(TOP$)
TOPGPS=STRTP
IF(DOWN4(TOPGPS).EQ.0)GOTO 25
TOPGPS=DOWN4(TOPGPS)
IF(DOWN2(TOPGPS).EQ.0)GOTO 26
TOPGPS=DOWN2(TOPGPS)
GOTO 27
25 CALL GET2(K)
DOWN4(TOPGPS)=K
TOPGPS=K
DOWN2(TOPGPS)=0
NAME2(TOPGPS)=GROUP
GOTO 26
26 CALL GET2(K)
DOWN2(TOPGPS)=K
TOPGPS=K
NAME2(TOPGPS)=GROUP
DOWN2(TOPGPS)=0
GOTO 20
C 1 DELETE OLDEST OF THREE MOVES ADD LAST MOVE TO BOTH NORTH AND SOUTH
TOPMS=MOVES
TOPMN=MOVEN
31 PT=DOWN2(TOPMS)
NAME2(TOPMS)=NAME2(PT)
IF(DOWN2(PT).EQ.0)GOTO 30
TOPMS=DCWN2(TOPMS)
GOTO 31
30 NAME2(PT)=STRATN
STRAT=1
CALL STRTOT(STRAT)
C SHIFT NEW MOVE INTO BOTTOM OF NORTH'S MOVE PATTERN AND UPDATE TOTAL
C HEADER.
C
32 PT=DOWN2(TOPMN)
NAME2(TOPMN)=NAME2(PT)
IF(DOWN2(PT).EQ.0)GOTO 35
TOPMN=DOWN2(TOPMN)
GOTO 32
35 NAME2(PT)=STRATN
STRAT=0
CALL STRTOT(STRAT)
GOTO 4
20 IF(NC.GT.3)GOTO 104
IF(DOWN4(TOPN).EQ.0)GOTO 110
TOPMN=DCWN4(TOPN)
108 IF(DOWN2(TOPMN).EQ.0)GOTO 105

```

```

TOPMN=DGWN2(TOPMN)
GOTO 108
105 CALL GET2(K)
DOWN2(TOPMN)=K
TOPMN=DOWN2(TOPMN)
DOWN2(TOPMN)=0
NAME2(TOPMN)=STRATN
STRAT=C
CALL STRTOT(STRAT)
IF(NC.EQ.3) GOTO 104
RETURN
110 CALL GET2(K)
DOWN4(TOPN)=K
MOVE=N=K
TOPMN=MOVE
DOWN2(TOPMN)=0
NAME2(TOPMN)=STRATN
STRAT=0
CALL STRTOT(STRAT)
RETURN
104 MOVE=DOWN4(PATRNN)
MOVE=MOVE
BYPASS=0
CALL TOTAL(MOVE,TOTRUE,BYPASS,INCNT,TOTAL,TOTAL2)
TMOVE=TOTAL2
CALL TOTMV(TOTAL,GROUP)
STRTP=LLINK4(TOPN)
TOPGPN=STRTP
IF(DOWN4(TOPGPN).EQ.0) GOTO 125
TOPGPN=DOWN4(TOPGPN)
127 IF(DOWN2(TOPGPN).EQ.0) GOTO 126
TOPGPN=DOWN2(TOPGPN)
GOTO 127
125 CALL GET2(K)
DOWN4(TOPGPN)=K
TOPGPN=K
DOWN2(TOPGPN)=0
NAME2(TOPGPN)=GROUP
RETURN
126 CALL GET2(K)
DOWN2(TOPGPN)=K
TOPGPN=K
NAME2(TOPGPN)=GROUP
DOWN2(TOPGPN)=0
RETURN
END

```

```

C
C
C
SUBROUTINE REMEMR(INCNT, INCNTR)
SURROUTINE TO RECONSTRUCT PERMANENT MEMORY PRIOR TO
A NEW GAME OR SERIES OF GAMES.
IMPLICIT INTEGER (A-W)
COMMON/TDJE/APN,APS,STACK(40)
COMMON/PAR/P(4),S(4),DRRLS(5)
COMMON/EEE/EN,ES,TOPEN,TOPES
COMMON/ENP/ENARRY(50),ESARRY(50)
COMMON/CLS/DEFS,BDEFS,OFFEN,POFFEN,PMOVEN,PMOVES
COMMON/JEC/VALUE(7,3),RTARN(3,3),RTARS(3,3),ATARN(3),ATARS(3),
1  ISN,SS,WN,WS,RS,RW,NC,INRAND,YRAND,EFLAG,CBN,CPS,STRATN,STRATS,
2  PNOLD,ISN,ISS,ECNXXN,ECNXS,PN,PS,FACTXN,FACTXS
COMMON/ST/STRUE,LASTGO,NFLAG,TUPN,ANAL,RECON
COMMON/T/AVAIL4,AVAIL3,AVAIL2,NAME4(1000),NAME2(1000),NAME3(1000)
COMMON/TD/TOP,BOTM,CATLG,CHAR,HIST,PLAYER,CATLGN,PATRNS,PATPNN
COMMON/TDP2/STK(10),STKM(10),STKN(10),STKNM(10)
COMMON/TDP8/RLHIST,NAHIST
COMMON/TDP3/TTYPE(20),INIT(20)
COMMON/TDP4/SEQ(20),SCONTR
COMMON/TDP7/GRNUPS,GRUPN,TMOVES,TMOVEN,MCNT,CONTR
DIMENSION RLINK4(1000),LLINK4(1000),DOWN4(1000),RLINK3(1000),DOWN3
1  (1000),DOWN2(1000)
EQUIVALENCE (NAME4(4),RLINK4(3),LLINK4(2),DOWN4(1)),(NAME2(2),DOWN
12  (1)),(NAME3(3),RLINK3(2),DOWN3(1))
DATA HEADS/HEAD/,TAILS/TAI/,YES/YES/,NO/NO/
DATA A1/A1/,A2/A2/,
WRITE(6,137) DO YOU WISH TO READ MEMORY AND REBUILD THE HISTORY OF
137 FORMAT(1)
1 PLAYERS, YES OR NO "A3" FORMAT JUSTIFIED)
695 READ(5,695)DU
FORMAT(A3)
IF(DU.EQ.NO) GOTO 27
READ(2,459)TSTK,(STACK(I),I=1,TSTK)
459 FORMAT(13,20I3)
READ(2,101)BUILD,INCNT,INCNTR,CNTR,SCONTR,APN,APS
101 FORMAT(7I3)
CALL REBUILD(INCNT)
CATLGN=0
10 IF(CATLGN.EQ.BUILD) GOTO 22
READ(2,102,END=50)PLAYER,NEM,DEM,NAG,DAG,NAP,RPA,LPA,DPA,NHIS,
1 PHIS,LHIS,DHIS
102 FORMAT(A4,12I3)
CALL CATLOG
ENPTR=NAME4(CHAR)
AGGRS=RLINK4(CHAR)

```

```

PARAM=LLINK4(CHAR)
NAME2(ENPTR)=NEM
DOWN2(ENPTR)=DEM
NAME2(AGGRS)=NAG
DOWN2(AGGRS)=DAG
NAME4(PARAM)=NAP
RLINK4(PARAM)=RPA
LLINK4(PARAM)=LPA
NAME4(HIST)=NHIS
RLINK4(HIST)=RHIS
LLINK4(HIST)=LHIS
DOWN4(HIST)=DHIS
GOTO 10
CONTINUE
50 MCNT=0
22 IF(MCNT.EQ.INCNT)GOTO 23
104 READ(2,104)MOVE1,MOVE2,MOVE3,TYPER
FORMAT(4I4)
MCNT=MCNT+1
TOP1=INIT(MCNT)
NAME4(TOP1)=MOVE1
RLINK4(TOP1)=MOVE2
LLINK4(TOP1)=MOVE3
DOWN4(TOP1)=TYPER
GOTO 21
23 TCNT=0
26 IF(TCNT.EQ.CONTR) GOTO 230
106 READ(2,106)DEF,OFF
FORMAT(2I4)
TCNT=TCNT+1
TOPT=TYPE(TCNT)
NAME3(TOPT)=DEF
RLINK3(TOPT)=OFF
CALL GET2(J)
DOWN3(TOPT)=J
TOPG=J
24 READ(2,109)NAMES,DOWNR
109 FORMAT(2I4)
IF(NAMES.EQ.15.OR.NAMES.EQ.25) GOTO 580
NAME2(TOPG)=NAMES
582 IF(DOWNR.EQ.0) GOTO 25
CALL GET2(J)
DOWN2(TOPG)=J
TOPG=DOWN2(TOPG)
GOTO 24
580 IF(NAMES.EQ.25)GOTO 581
NAME2(TOPG)=A1

```

```

GOTO 582
581 NAME2(TOPG)=A2
    GOTO 582
25   DOWN2(TOPG)=0
    GOTO 26
23C SCNT=0
231 IF(SCNT.EQ.SCONTR) GOTO 27
    SCNT=SCNT+1
    TOPQ=SEQ(SCNT)
    RTOPQ=TOPQ
    READ(2,121) TYPES,MOV11,MOV12,MOV13,DOWNS
    NAME3(RTOPQ)=TYPES
    CALL GET3(L)
    RLINK3(RTOPQ)=L
    NAME3(L)=MOV11
    RLINK3(L)=MOV12
    DOWN3(L)=MOV13
    IF(DOWNS.EQ.0) GOTO 31
    CALL GET3(L)
    DOWN3(RTOPQ)=L
    RTOPQ=DOWN3(RTOPQ)
    GOTO 200
31   DOWN3(RTOPQ)=0
    GOTO 231
121 FORMAT(5I4)
227 RETUPN
    END

SURROUTINE REBILD(INCNT)
SURROUTINE TO REBUILD THE HEADER CELLS FOR THE INITIAL,
TYPE, AND SEQUENCE LIBRARIES.

IMPLICIT INTEGER (A-W)
COMMON/TDJE/APN,APS,STACK(40)
COMMON/PAR/P(4),S(4),PRALS(5)
COMMON/EEE/EN,ES,TOPEN,TOPE
COMMON/ENP/ENARRY(50),OFFEN,PARRY(50)
COMMON/CLS/DEFS,BDEFS,CFFEN,RDEFN,PMOVEN,PMOVES
COMMON/JEC/VALUE(7,3),RTABN(3,3),RTARS(3,3),ATABN(3),ATARS(3),
1SN,SS,WN,WS,RS,ECHXN,ECN,XS,PN,PS,FAC TXN,FAC TXS
2PNOLD,I SN,I SS,ECHXN,ECN,XS,PN,PS,FAC TXN,FAC TXS
COMMON/ST/STRUE,LASTGO,NFLAG,TURN,ANAL,REC CON
COMMON/T/AVAIL4,AVAIL3,AVAIL2,NAME4(1000),NAME2(1000),NAME3(1000)
COMMON/TD/TOP,BOTM,CATLG,CHAR,HIST,PLAYER,CATLGN,PATRNS,PATRN
COMMON/TDP2/STK(10),STKM(10),STKN(10),STKNM(10)
COMMON/TDP9/TYPENU,TYPE1

```

```

COMMON/TDP3/TYPE(20),INIT(20)
COMMON/TDP4/SEQ(20),SCONTR
COMMON/TDP7/GROUPS,GRUPN,TMOVES,TMOVEN,MCNT,CONTR
DIMENSION COMPN(3,3)
DIMENSION RLINK4(1000),LLINK4(1000),DOWN4(1000),RLINK3(1000),DOWN3
1(1000),DOWN2(1000)
EQUIVALENCE (NAME4(4),RLINK4(3),LLINK4(2),DOWN4(1)),(NAME2(2),DOWN
12(1)),(NAME3(3),RLINK3(2),DOWN3(1))
DATA HEADS/'HEAD',TAILS/'TAIL',YES/'YES',NO/'NO'/
DATA A1/'A1',A2/'A2'/
RSTK=0
SSTK=0
TSTK=0
11 IF(TSTK.EQ.INCNT) GOTO 40
41 TSTK=TSTK+1
42 TJ=AVAIL4
JJ=TJ
CALL GET4(JJ)
IF(STACK(TSTK).EQ.J) GOTO 15
RJ=JJ
DOWN4(JJ)=AVAIL4
JJ=DOWN4(JJ)
GOTO 42
15 INIT(TSTK)=J
DOWN4(RJ)=AVAIL4
AVAIL4=TJ
DOWN4(JJ)=0
GOTO 11
40 IF(RSTK.EQ.CONTR) GOTO 400
RSTK=RSTK+1
TSTK=TSTK+1
TL=AVAIL3
LL=TL
410 CALL GET3(LL)
IF(STACK(TSTK).EQ.L) GOTO 150
RL=LL
DOWN3(LL)=AVAIL3
LL=DOWN3(LL)
GOTO 410
150 TYPE(RSTK)=L
DOWN3(RL)=AVAIL3
AVAIL3=TL
GOTO 40
400 IF(SSTK.EQ.SCONTR) GOTO 450
SSTK=SSTK+1
TSTK=TSTK+1
TL=AVAIL3
LL=TL

```

```

451 CALL GET3(L)
   IF(STACK(TSTK).EQ.L) GOTO 449
   RL=LL
   DOWN3(LL)=AVAIL3
   LL=DOWN3(LL)
   GOTO 451
449 SEQ(SSTK)=L
   DOWN3(RL)=AVAIL3
   AVAIL3=TL
   GOTO 400
450 RETURN
   END

SUBROUTINE SAVE(INCNT,INCNT,DU)
SUBROUTINE TO SAVE DATA IN THE PERMANENT MEMORY AT THE END OF
A GAME OR A SERIES OF GAMES.

IMPLICIT INTEGER (A-W)
COMMON/TDJE/APN,APS,STACK(40)
COMMON/T/AVAIL4,AVAIL3,AVAIL2,NAME4(1000),NAME2(1000),NAME3(1000)
COMMON/TD/TOP,BOTM,CATLG,CHAR,HIST,PLAYER,CATLGN,PATRNS,PATRNN
COMMON/TDP3/TYPE(20),INIT(20)
COMMON/TDP4/SEQ(20),SCONTR
COMMON/TDP7/GRUUPS,GRUUPN,TMOVFS,TMCVEN,MCNT,CONTR
DATA HEADS/HEAD//,TAILS/TAI//,YES/YES//,NO//NO//
DATA AI/A1//,A2//A2//
DIMENSION RLINK4(1000),DOWN4(1000),RLINK3(1000),DOWN3
(1000),DOWN2(1000)
EQUIVALENCE (NAME4(4),RLINK4(3),LLINK4(1)),(NAME2(2),DOWN
12(1)),(NAME3(3),RLINK3(2),DOWN3(1))
101 FORMAT(7I3)
   TSTK=0
   SSTK=0
   RSTK=0
401 IF(TSTK.EQ.INCNT)GOTO 400
   TSTK=TSTK+1
   STACK(TSTK)=INIT(TSTK)
   GOTO 401
400 IF(RSTK.EQ.CONTR) GOTO 500
   RSTK=RSTK+1
   TSTK=TSTK+1
   STACK(TSTK)=TYPE(RSTK)
   GOTO 400
500 IF(SSTK.EQ.SCONTR) GOTO 600
   SSTK=SSTK+1
   TSTK=TSTK+1

```

CC

```

STACK(TSTK)=SEO(SSTK)
GOTO 500
TSTK=TSTK+1
STACK(TSTK)=0
WRITE(3,560)TSTK,(STACK(I),I=1,TSTK)
WRITE(3,101)CATLGN,INCNTR,CONTR,SCONTR,APN,ADS
56C FORMAT(I3,20I3)
TOP=ROTM
ROTM=ROTM
DO 20 I=1,CATLGN
CHAR=RLINK2(TOP)
ENPTR=NAME4(CHAR)
AGGR=RLINK4(CHAR)
PARAM=LLINK4(CHAR)
HIST=DOWN4(CHAR)
WRITE(3,102)NAME3(TOP),NAME2(ENPTR),DOWN2(ENPTR),PARAM,
1 NAME2(AGGRS),DOWN2(AGGRS),NAME4(PARAM),RLINK4(HIST),
2 LLINK4(PARAM),DOWN4(PARAM),NAME4(HIST),RLINK4(HIST),
3 LLINK4(HIST),DOWN4(HIST)
102 FORMAT(A4,12I3)
IF(DU.EQ.NO)GOTO 55
WRITE(6,103)NAME3(TOP),NAME2(ENPTR),DOWN2(ENPTR),
1 NAME2(AGGRS),DOWN2(AGGRS),NAME4(PARAM),RLINK4(HIST),
2 DOWN4(HIST),NAME4(PARAM),LLINK4(PARAM),DOWN4(PARAM)
55 IF(TOP.EQ.CATLG)GOTO 99
TOP=CATLG
551 IF(DOWN3(TOP).EQ.ROTM)GOTO 550
GOTO 551
TOP=DOWN3(TOP)
ROTM=TOP
CONTINUE
DO 22 I=1,INCNTR
TOPI=INIT(I)
MOVE1=NAME4(TOPI)
MOVE2=RLINK4(TOPI)
MOVE3=LLINK4(TOPI)
TYPE=DOWN4(TOPI)
WRITE(3,104)MOVE1,MOVE2,MOVE3,TYPE
104 FORMAT(4I4)
IF(DU.EQ.NO)GOTO 56
WRITE(6,105)INIT(I),MOVE1,MOVE2,MOVE3,TYPE
56 RUN=1
CONTINUE
DO 23 I=1,CONTR
TOPT=TYPE(I)
DEF=NAME3(TOPT)
G=0
OFF=RLINK2(TOPT)

```

```

106 WRITE(3,106)DEF,OFF
    FORMAT(2I4)
    IF(DU.EQ.NO)GOTO 59
112 WRITE(6,112)I,TOP,DEF,OFF
159 FORMAT(5X,'TYPE(',I2,')=',I3,/2X,'DEFF=',I3,/2X,'OFF=',I3/)
57 RTOP=DOWN3(TOP)
    G=0+1
17 IF(NAME2(RTOP).EQ.A1.OR.NAME2(RTOP).EQ.A2)GOTO 60
    WRITE(3,17)NAME2(RTOP),DOWN2(RTOP)
    FORMAT(2I4)
    IF(DU.EQ.NO)GOTO 58
111 WRITE(6,111)G,NAME2(RTOP)
58 FORMAT(5X,'GROUP(',I2,')=',I4/)
    IF(DOWN2(RTOP).EQ.0)GOTO 23
    RTOP=DOWN2(RTOP)
    GOTO 57
60 IF(NAME2(RTOP).EQ.A1)GOTO 633
    IF(DU.EQ.NO)GOTO 590
    WRITE(6,110)G,NAME2(RTOP)
580 NAME2(RTOP)=25
581 WRITE(3,109)NAME2(RTOP),DOWN2(RTOP)
109 GOTO 58
633 FORMAT(2I4)
    IF(DU.EQ.NO)GOTO 590
    WRITE(6,110)G,NAME2(RTOP)
590 NAME2(RTOP)=15
110 FORMAT(5X,'GROUP(',I2,')=',I4/)
    GOTO 581
23 CONTINUE
    DO 24 I=1,SCNTR
        FLAG=0
        TOPS=SEQ(I)
117 RTOP=TOPS
118 TYPES=NAME3(RTOPS)
        MOV1=RLINK3(RTOPS)
        MOV11=NAME3(MOV1)
        MOV12=RLINK3(MOV1)
        MOV13=DOWN3(MOV1)
        IF(FLAG.GT.0)GOTO 125
        DOWNS=DOWN3(RTOPS)
        WRITE(3,121)TYPES,MOV11,MOV12,MOV13,DOWNS
121 FORMAT(5I4)
116 IF(DOWN3(RTOPS).EQ.0)GOTO 119
        RTOP=DOWN3(RTOPS)
        GOTO 118
119 IF(DU.EQ.NO)GOTO 24
        IF(FLAG.GT.0)GOTO 24
        WRITE(6,123)TOPS

```



```

20 IF(CKGRP2.EQ.2.OR.CKGRP2.EQ.3) GOTO 66
700 GOTO 76
RETURN
END

C
C
C
SUBROUTINE TOTMV(TOTAL, GROUP)
SUBROUTINE TO CLASSIFY COMBINATIONS OF ANY THREE MOVES TO
PRODUCE CATEGORIES OF GROUP 1, A1, 2, A2, 3.
IMPLICIT INTEGER (A-W)
COMMON/T/AVAIL4, AVAIL3, AVAIL2, NAME4(1000), NAME2(1000), NAME3(1000)
COMMON/TD/TOP, BOTM, CATLG, CHAR, HIST, PLAYER, CATLGN, PATRNS, PATRNN
DIMENSION RLINK4(1000), LLINK4(1000), DOWN4(1000), PLINK3(1000), DOWN3
1(1000), DOWN2(1000)
EQUIVALENCE (NAME4(4), RLINK4(3), LLINK4(2), DOWN4(1)), (NAME2(2), DOWN
12(1)), (NAME3(3), RLINK3(2), DOWN3(1))
DATA A1/A1, A2/A2, /
IF(TOTAL.EQ.3) GOTO 4
IF(TOTAL.EQ.5) GOTO 5
IF(TOTAL.EQ.6) GOTO 6
IF(TOTAL.EQ.7) GOTO 7
IF(TOTAL.EQ.8) GOTO 9
GROUP=1
RETURN
GROUP=A1
RETURN
GROUP=2
RETURN
GROUP=A2
RETURN
GROUP=3
RETURN
END

C
C
C
SUBROUTINE GETMV(NXTGRU, TMOVE2, PMOVE)
SUBROUTINE TO DETERMINE NEXT MOVE FROM PREDICTED GROUP.
IMPLICIT INTEGER (A-W)
COMMON/T/AVAIL4, AVAIL3, AVAIL2, NAME4(1000), NAME2(1000), NAME3(1000)
COMMON/TD/TOP, BOTM, CATLG, CHAR, HIST, PLAYER, CATLGN, PATRNS, PATRNN
DIMENSION RLINK4(1000), LLINK4(1000), DOWN4(1000), RLINK3(1000), DOWN3
1(1000), DOWN2(1000)
EQUIVALENCE (NAME4(4), RLINK4(3), LLINK4(2), DOWN4(1)), (NAME2(2), DOWN
12(1)), (NAME3(3), RLINK3(2), DOWN3(1))

```

10	DATA A1/A1//A2//A2//A2//	
2	IF(NXTGRU.EQ.1) GOTO 10	
3	IF(NXTGRU.EQ.A1) GOTO 11	
11	IF(NXTGRU.EQ.2) GOTO 12	
22	IF(NXTGRU.EQ.A2) GOTO 13	
23	IF(NXTGRU.EQ.3) GOTO 14	
24	IF(TMOVE2.EQ.2) GOTO 2	
12	IF(TMOVE2.EQ.3) GOTO 2	
33	IF(TMOVE2.EQ.4) GOTO 2	
34	IF(TMOVE2.EQ.5) GOTO 2	
35	IF(TMOVE2.EQ.6) GOTO 2	
13	IF(TMOVE2.EQ.7) GOTO 2	
44	IF(TMOVE2.EQ.8) GOTO 2	
45	IF(TMOVE2.EQ.9) GOTO 2	
46	IF(TMOVE2.EQ.10) GOTO 2	
14	IF(TMOVE2.EQ.11) GOTO 2	
56	IF(TMOVE2.EQ.12) GOTO 2	
55	IF(TMOVE2.EQ.13) GOTO 2	
	END	

```

CCCCC
SUBROUTINE STRTOT(STRAT)
SUBROUTINE TO DETERMINE WHERE THE LAST MOVE FITS IN THE TOTAL
OF TYPES OF MOVES IN THE TEMPORARY MEMORY HEADER AND UPDATE THE
APPROPRIATE CELLS IN THE HEADER NODES.

IMPLICIT INTEGER (A-W)
COMMON/JEC/VALUE(7,3),RTABN(3,3),RTABS(3,3),ATABN(3),ATARS(3),
1SN,SS,WN,WS,RS,RW,NC,INRAND,YRAND,EFLAG,CPN,CPS,STRATN,STRATS,
2PNOLD,ISS,ECNXXN,ECNXP,PN,PS,EACTXN,FACTXS
COMMON/T/AVAIL3,AVAIL2,NAME4(1000),NAME2(1000),NAME3(1000)
COMMON/TD/TOP,BOTM,CATLG,CHAR,HIST,PLAYER,CATLGN,PATRNS,PATRNN
DIMENSION RLINK4(1000),LLINK4(1000),DOWN4(1000),RLINK3(1000),DOWN3
1(1000),DOWN2(1000)
EQUIVALENCE (NAME4(4),RLINK4(3),LLINK4(2),DOWN4(1)),(NAME2(2),DOWN
12(1)),(NAME3(3),RLINK3(2),DOWN3(1))
IF(STRAT.EQ.1) GO TO 100
STRTP=LLINK4(PATRNN)
VALU1=STRATN-2
IF(VALU1)1C,20,30
10 NAME4(STRTP)=NAME4(STRTP)+1
RETURN
20 RLINK4(STRTP)=RLINK4(STRTP)+1
RETURN
30 LLINK4(STRTP)=LLINK4(STRTP)+1
RETURN
100 STRTP=LLINK4(PATRNS)
VALU1=STRATS-2
IF(VALU1)11,21,31
11 NAME4(STRTP)=NAME4(STRTP)+1
RETURN
21 RLINK4(STRTP)=RLINK4(STRTP)+1
RETURN
31 LLINK4(STRTP)=LLINK4(STRTP)+1
RETURN
END

SUBROUTINE TOTALM(MOVE,TOTRUE,BYPASS,INCNT,MCNT,TOTAL,TOTAL2)
THIS SUBROUTINE TOTALS THE THREE MOVES IN TEMPORARY MEMORY
FOR BOTH PATRNN AND PATRNS.

IMPLICIT INTEGER (A-W)
COMMON/T/AVAIL4,AVAIL3,AVAIL2,NAME4(1000),NAME2(1000),NAME3(1000)
COMMON/TD/TOP,BOTM,CATLG,CHAR,HIST,PLAYER,CATLGN,PATRNS,PATRNN
COMMON/TDP/MOVES,MOVEN
COMMON/TDP3/TYPE(20),INIT(20)

```

```

DIMENSION RLINK4(1000), LLINK4(1000), DOWN4(1000), RLINK3(1000), DOWN3
1(1000), DOWN2(1000)
EQUIVALENCE (NAME4(4), RLINK4(3), LLINK4(2), DOWN4(1)), (NAME2(2), DOWN
12(1)), (NAME3(3), RLINK3(2), DOWN3(1))
DATA A1//A1//, A2//A2//
MCNT=1
TOTAL2=0
TOTAL=C
IF(MOVE.EQ.MOVES)GOTO 100
TOPMN=MOVE
106 TOTAL=TOTAL+NAME2(TOPMN)
IF(DOWN2(TOPMN).EQ.0) GOTO 107
TOPMN=DOWN2(TOPMN)
TOTAL2=TOTAL2+NAME2(TOPMN)
GOTO 106
100 TOPMS=MOVE
108 TOTAL=TOTAL+NAME2(TOPMS)
IF(DOWN2(TOPMS).EQ.0) GOTO 107
TOPMS=DOWN2(TOPMS)
TOTAL2=TOTAL2+NAME2(TOPMS)
GOTO 108
107 IF(BYPASS.EQ.0) GOTO 110
C
C
C BYPASS IF YOU DO NOT WANT TO TOTAL MOVES IN THE INITIAL PATTERN.
62 IF(MCNT.GT.INCNT)GOTO 64
TOPI=INIT(MCNT)
ITOTAL=0
ITOTAL=ITOTAL+NAME4(TOPI)
ITOTAL=ITOTAL+RLINK4(TOPI)
ITOTAL=ITOTAL+LLINK4(TOPI)
IF(TOTAL.EQ.ITOTAL)GOTO 65
MCNT=MCNT+1
GOTO 62
65 TOTRUE=1
RETURN
64 TOTRUE=0
C
C
C "0"=> NOT TRUE NO MATCH IN THE INITIAL LIBRARY.
"1"=> TRUE THERE IS A MATCH IN INITIAL LIBRARY.
110 RETURN
END

```

```

CCCC
SUBROUTINE MOVEPL
SUBROUTINE TO PUT OLD PLAYER TO THE TOP OF THE CATALOGUE.
IMPLICIT INTEGER (A-W)
COMMON/T/AVAIL4,AVAIL3,AVAIL2,NAME4(1000),NAME2(1000),NAME3(1000)
COMMON/ID/TOP,ROTM,CATLG,CHAR,HIST,PLAYER,CATLGN,PATRNS,PATPNN
DIMENSION RLINK4(1000),LLINK4(1000),DOWN4(1000),RLINK3(1000),DOWN3
1(1000),DOWN2(1000)
EQUIVALENCE (NAME4(4),RLINK4(3),LLINK4(2),DOWN4(1)),(NAME2(2),DOWN
12(1)),(NAME3(3),RLINK3(2),DOWN3(1))
MV=CATLG
IF(MV.EQ.TOP) GOTO 4
IF(DOWN3(MV).EQ.TOP) GOTO 2
MV=DOWN3(MV)
GOTO 1
IF(DOWN3(MV).NE.ROTM) GOTO 3
ROTM=MV
DOWN3(ROTM)=0
DOWN3(MV)=DOWN3(TOP)
DOWN3(TOP)=CATLG
CATLG=TOP
RETURN
END
1
2
3
4

```

```

CCCC
SUBROUTINE SEARCH(PLY)
SUBROUTINE TO FIND OUT IF A PLAYER IS IN THE CATALOGUE
OF PLAYERS
IMPLICIT INTEGER (A-W)
COMMON/T/AVAIL4,AVAIL3,AVAIL2,NAME4(1000),NAME2(1000),NAME3(1000)
COMMON/ID/TOP,ROTM,CATLG,CHAR,HIST,PLAYER,CATLGN,PATRNS,PATPNN
DIMENSION RLINK4(1000),LLINK4(1000),DOWN4(1000),RLINK3(1000),DOWN3
1(1000),DOWN2(1000)
EQUIVALENCE (NAME4(4),RLINK4(3),LLINK4(2),DOWN4(1)),(NAME2(2),DOWN
12(1)),(NAME3(3),RLINK3(2),DOWN3(1))
TOP=CATLG
IF(NAME3(TOP).EQ.PLAYER) GOTO 1
IF(DOWN3(TOP).EQ.0) GOTO 3
TOP=DOWN3(TOP)
GOTO 2
PLY=0
RETURN
PLY=1
RETURN
END
2
3
1

```

```

C
C
SUBROUTINE CLSMVS(MOV, INCNT, PCNT, PCNTM, INCNTR)
SUBROUTINE TO CLASSIFY INITIAL MOVES FROM SOUTHS PATTERN.

IMPLICIT INTEGER (A-W)
COMMON/T/AVAIL2, AVAIL3, AVAIL4, NAME2(1000), NAME3(1000)
COMMON/TDPS/TYPENU, TYPE1, TYSCU, TYNOR
COMMON/TD/TOP, BOTM, CATLG, CHAR, HIST, PLAYER, CATLGN, PATRNS, PATRNN
COMMON/TDP4/SEQ(20), SCCNTR
COMMON/TDP3/TYPE(20), INIT(20)
COMMON/TDP2/STK(10), STKM(10), STKN(10), STKNM(10)
COMMON/CLS/DEFFS, BDEFFS, OFFEN, ROFFEN, PMOVEN, PMOVES
COMMON/JEC/VALUE(7,3), RTABN(3,3), RTABS(3,3), ATABN(3), ATABS(3),
1  ISN, SS, WN, WS, RS, ECN, RN, NC, INRAND, YR, AND, EFLAG, CPN, CPS, STRATN, STRATS,
2  PNOLD, ISN, ISS, ECN, RN, NC, INXN, PN, PS, FACTXN, FACTXS
COMMON/TDP7/GRUUPS, GROUPN, TMOVES, TMOVEN, MCNT, CONTR
COMMON/TDP8/RLHIST, NAHIST
DIMENSION RLINK4(1000), DOWN4(1000), RLINK3(1000), DOWN3
1  (1000), DOWN2(1000)
EQUIVALENCE (NAME4(4), RLINK4(3), LLINK4(2), DOWN4(1)), (NAME2(2), DOWN
12(1)), (NAME3(3), RLINK3(2), DOWN3(1))
MOV2=NC
MCNT=1
DOCTM=1
DOCTM=1
MCNT=1
MOVES=DOWN4(PATRNS)
IF(INCNT.EQ.0)GOTO 82
TCPMS=MOVES
IF(MOV2.GT.3) GOTO 35
IF(MCNT.GT.INCNT) GOTO 20
TOP1=INIT(MCNT)
IF(NAME4(TOP1).EQ.NAME2(TOPMS)) GOTO 30
MCNT=MCNT+1
GOTO 10
IF(PCNT.GT.0) GOTO 21
DEFES=0
"O"=> NO DEFENSIVE STRATEGY IS AVAILABLE. FIND BEST DEFENSE
FOR MIXED MOVE MATCH STRATEGY.
MCNT=1
TOPMS=MOVES
43 IF(MCNT.GT.INCNT)GOTO 40
41 TOP1=INIT(MCNT)
IF(RLINK4(TOP1).EQ.NAME2(TOPMS))GOTO 45
MCNT=MCNT+1
GOTO 41

```

```

40 IF(PCNTM.GT.0) GOTO 42
   RDEFES=0
   PMOVES=0
   "O"=X BEST DEFENSE FROM MIXED MOVE STRATEGY.
   RETURN
45 DCTM=DCTM+1
   TOPMS=DCWN2(TOPMS)
   IF(NAME4(TOPI).EQ.NAME2(TOPMS)) GOTO 46
   MCNT=MCNT+1
   DCTM=1
   GOTO 43
49 POINTR=MCNT
   PCNTM=PCNTM+1
   STKM(PCNTM)=POINTR
   PTRM=PCINTR
   MCNT=MCNT+1
   DCTM=1
   GOTO 43
42 IF(PCNTM.GT.1) GOTO 47
   TOPI=INIT(PTRM)
   MCNT=PTRM
   PMOVES=LLINK4(TOPI)
   NORSE=C
   CALL CLSTYP(NORSE)
   RDEFES=DEFES
   RETURN
47 PTRM=STKM(PCNTM)
   PCNTM=PCNTM-1
   TOPI=INIT(PTRM)
   MCNT=PTRM
   GOTO 46
30 DCT=DCT+1
   TOPMS=DCWN2(TOPMS)
   IF(LLINK4(TOPI).EQ.NAME2(TOPMS)) GOTO 31
   DCT=1
   MCNT=MCNT+1
   GOTO 9
31 POINTR=MCNT
   PCNT=PCNT+1
   STK(PCNT)=POINTR
   PTR=PCINTR
   DCT=1
   MCNT=MCNT+1
   GOTO 9
21 IF(PCNT.GT.1)GOTO 25
   TOPI=INIT(PTR)
   MCNT=PTR

```

```

24 PMOVES=LLINK4(TOPI)
   NORSO=0
   CALL CLSTYP(NORSO)
   RETURN
25 PTR=STK(PCNT)
   TOPI=INIT(PTR)
   MCNT=PTR
   GOTO 24
   GET STRATEGY FOR THE FORTH MOVE AND SET UP
   ADDITIONAL INITIAL MOVES CELLS.
35 IF(PCNT.EQ.0) GOTO 36
   PTR=STK(PCNT)
   PCNT=PCNT-1
   TOPMS=MOVES
   DCNT=1
39 IF(DCNT.EQ.3) GOTO 38
   TOPMS=DCWN2(TOPMS)
   DCNT=DCNT+1
   GOTO 39
38 TOPI=INIT(PTR)
   MCNT=PTR
   IF(LLINK4(TOPI).EQ.NAME2(TOPMS)) GOTO 55
   GOTO 35
60 MOVE=1
   MOVE=MOVES
   CKINCT=INCENT
   CALL INTLIB(INCENT,MOVE,TOPI,INCENTR,MOV)
   IF(CKINCT.EQ.INCENT) GOTO 81
   NOW CELL HAS BEEN ADDED
   MOVES=DCWN4(PATRS)
   TOPMS=MOVES
   NAME4(TOPI)=NAME2(TOPMS)
   TOPMS=DCWN2(TOPMS)
   RLINK4(TOPI)=NAME2(TOPMS)
   TOPMS=DCWN2(TOPMS)
   LLINK4(TOPI)=NAME2(TOPMS)
   DOWN4(TOPI)=0
   GOTO 80
36 DEFES=0
   MOVE=MOVES
   BYPASS=1
   CALL TOTALM(MOVE,TOTRUE,BYPASS,INCENT,MCNT,TOTAL,TOTAL2)
   IF(TOTRUE) 64,64,65
65 NORSO=C
   IF(DOWN4(MCNT).EQ.0) GOTO 650
   TOPI=INIT(MCNT)
   IF(DOWN4(TOPI).EQ.0) GO TO 650

```

```

65C  TYPE NU=DOWN4(TOPT)
      CALL CLSTYP(NORSO)
      PDEFS=DEFS
      GOTO 6C
64    PDEFS=0
      GOTO 6C
65    NORSO=C
      CALL CLSTYP(NORSO)
      GOTO 6C
80    NAHIST=INIT(INCNT)
      RETURN
81    NAHIST=TOPI
      RETURN
82    IE(MOV2.GT.3)GOTO 6C
      PDEFS=0
      PMOVES=0
      RETURN
      END

```

SUBROUTINE CLSTYP(NORSC)

SUBROUTINE TO DETERMINE IF A PLAYER FITS A PARTICULAR PATTERN IN THE TYPE LIBRARY AND IF SO OBTAIN THE PRECICTED MOVE FOR THE BEST DEFENSIVE OR OFFENSIVE MOVE FOR THE MATCHED PATTERN.

```

NORSO=> NORTH OR SOUTH 1=> NORTH (COMPUTER)
0=> SOUTH (PLAYER)

```

```

IMPLICIT INTEGER (A-W)
COMMON/T/AVAIL4,AVAIL3,AVAIL2,NAME4(1000),NAME2(1000),NAME3(1000)
COMMON/TD/TOP,BOTM,CATLG,CHAR,HIST,PLAYER,CATLGN,PATRN,PATRN
COMMON/TDP9/TYPNU,TYPE1,TYSQU,TYNDR
COMMON/TDP7/GRUUPS,GRUUPN,TMOVES,TMOVEN,MCNT,CONTR
COMMON/TDP4/SEQ(20),SCONTR
COMMON/TDP3/TYPE(20),INIT(20)
COMMON/CLS/DEFS,BDEFS,OFFEN,ROFFEN,PMOVEN,PMOVES
COMMON/JEC/VALUE(7,3),RTABN(3,3),ATARN(3),ATARS(3),
1SN,SS,WN,WS,RS,PW,NC,INRND,YRAND,EFLAG,CN,CPS,STRATN,STRATS,
2PNOLD,ISN,ISS,ECNXN,FCNXS,PN,PS,FACITXN,FACITXS
DIMENSION RLINK4(1000),LLINK4(1000),DOWN4(1000),RLINK3(1000),DOWN3
1(1000),DOWN2(1000)
EQUIVALENCE (NAME4(4),RLINK4(3),LLINK4(2),DOWN4(1)),(NAME2(2),DOWN
12(1)),(NAME3(3),RLINK3(2),DOWN3(1))

```

CCCCCCCC

```

TOT=0
IFLAG=0
IF(NC.GT.3) GOTO 11
CALL FINDTY(MCNT,TYPENU,NORSC)
I=(TYPENU.EQ.0) GOTO 100
TYPE1=TYPENU
IF(NC.EQ.3) GOTO 10
IF(NC.EQ.2) GOTO 1
IF(NORSC.EQ.1) GOTO 2
TYSOU=TYPE1
PMOVES=NAME3(TYPE1)
RETURN
PMOVEN=NAME3(TYPE1)
TYNOR=TYPE1
RETURN
IF(NORSC.EQ.1) GOTO 3
PMOVES=RLINK3(TYPE1)
TYSOU=TYPE1
RETURN
PMOVEN=RLINK3(TYPE1)
TYNOR=TYPE1
RETURN
MOVE=DOWN4(PATRNS)
IF(NORSC.EQ.1) MOVE=DOWN4(PATRNN)
TOT=TOT+NAME2(MOVE)
IF(DOWN2(MOVE).EQ.0) GOTO 5
MOVE=DOWN2(MOVE)
GOTO 6
IF(NORSC) 555,554,555
TYSOU=TYPE1
GOTO 556
TYNOR=TYPE1
FINDGU=DOWN3(TYPE1)
NEXTGRU=NAME2(DOWN2(FINDGU))
TMOVE2=TOT
CALL GETMV(NXTGRU,TMOVE2,PMOVE)
IF(NORSC.EQ.1) GOTO 50
PMOVES=PMOVE
DEFES=NAME3(TYPE1)
OFFES=RLINK3(TYPE1)
TYPE1=0
RETURN
PMOVFN=PMOVE
DEFEN=NAME3(TYPE1)
OFFEN=RLINK3(TYPE1)
IF (DEFES.EQ.OFFEN) GO TO 90
IF (DEFEN.EQ.0) GO TO 90
IF (OFFES.EQ.DEFEN) GOTO 51

```

2

1

3

10

6

5

554

555

556

50

```

IF(NC.LE.3) RETURN
GOTO 11
IF (NC.GT.4.AND.TFLAG.EQ.0) RETURN
TFLAG=0
FINDGU=DOWN3(DEFEN)
TYNOR=DEFEN
NXTGRU=NAME2(DOWN2(FINDGU))
CALL GETMV(NXTGRP, TMOVE2, PMOVE)
PMOVE=PMOVE
TYPE1=0
RETURN
11 IF (TYPENU.EQ.0) GOTO 100
IF(NOR SO) 777, 770, 777
770 IF (TYSOU.NE.0) GOTO 110
TYSOU=TYPENU
110 TYPE1=TYSOU
GOTO 112
777 IF (TYNOR.NE.0) GOTO 111
TYNOR=TYPENU
111 TYPE1=TYNOR
112 DOWNTY=DOWN3(TYPE1)
FINDGU=DOWN2(DOWNTY)
IF(NOR SO.EQ.1) GOTO 41
PATRN=PATRNS
TMOVE2=TMOVES
40 STRTP=LLINK4(PATRN)
TOPGP=DOWN4(STRTP)
DOWNPT=TOPGP
GOTO 42
41 PATRN=PATRNN
TMOVE2=TMOVEN
GOTO 40
42 IF(DOWN2(DOWNPT).EQ.0) GOTO 45
IF(DOWN2(FINDGU).EQ.0) GOTO 45
TOPGU=DOWNPT
TOPTY=DOWNTY
DOWNPT=DOWN2(DOWNPT)
DOWNTY=DOWN2(DOWNTY)
FINDGU=DOWN2(DOWNTY)
GOTO 42
45 IF(NAME2(DOWNPT).EQ.NAME2(DOWNTY)) GO TO 47
FLAG=0
IF(NAME2(TOPGU).EQ.NAME2(TOPTY)) GOTO 48
CKGRP1=NAME2(TOPGU)
CKGRP2=NAME2(TOPTY)
80 CALL CKTYP(CKGRP1, CKGRP2, NOGO, FLAG)
IF(NOGO.EQ.1) GOTO 49
47 NXTGRU=NAME2(FINDGU)

```

```

471 CALL GETMV(NXTGRU,TMOVE2,PMOVE)
   IF(NORSO.EQ.1) GOTO 43
   PMOVES=PMOVE
   DEFFS=NAME2(TYPE1)
   OFFES=RLINK3(TYPE1)
   TYPE1=0
   RETURN
43 PMOVEN=PMOVE
   DEFFEN=NAME2(TYPE1)
   OFFEN=RLINK3(TYPE1) GO TO 90
   IF (DEFFEN.EQ.OFFEN) GO TO 90
   IF (DEFFEN.EQ.O) GO TO 90
   IF (OFFES.EQ.DEFFEN) GO TO 51
   RETURN
48 FLAG=1
   CKGRP1=NAME2(DOWNPT)
   CKGRP2=NAME2(DOWNTY)
   GOTO 80
49 CONT=1
53 IF(CONT.GT.CONTR) GOTO 56
   TOPGP=DOWN3(TYPE(CONT))
   NXTGP=DOWN2(TOPGP)
   IF(NAME2(TOPGU).EQ.NAME2(TOPGP)) GOTO 57
   CKGRP1=NAME2(TOPGU)
   CKGRP2=NAME2(TOPGP)
   CALL CKTYP(CKGRP1,CKGRP2,NOGO,FLAG)
   IF(NOGO.EQ.O) GOTO 57
   CONT=CONT+1
   GOTO 53
57 FLAG=1
   IE2(DOWNPT).EQ.NAME2(NXTGP)) GOTO 58
   IF(NAME2(DOWNPT).EQ.NAME2(NXTGP))
   CKGRP1=NAME2(NXTGP)
   CKGRP2=NAME2(NXTGP)
   CALL CKTYP(CKGRP1,CKGRP2,NOGO,FLAG)
   IF(NOGO.EQ.O) GOTO 58
   CONT=CONT+1
   GOTO 53
56 IF(NORSO.EQ.1) GOTO 52
   PMOVES=0
   RETURN
52 PMOVEN=0
   RETURN
58 IF(NORSO.EQ.1) GOTO 55
   TYSCU=TYPE(CONT)
   NXTGRU=NAME2(NXTGP)
   TYPE1=TYPE(CONT)
   GO TO 471
55 TYNOR=TYPE(CONT)

```

```

TYPE1=TYPE(CONT)
NXTGRU=NAME2(NXTGP)
TFLAG=1
GO TO 471
100 PMOVES=0
    PMOVEN=0
    RETURN
    TYPE1=0
    RETURN
    END

SUBROUTINE CLSMVN(MOV, INCNT, PCNTN, PCNTNM, INCNTR)
SUBROUTINE TO CLASSIFY INITIAL MOVES FROM NORTH'S PATTERN.

IMPLICIT INTEGER (A-W)
COMMON/T/AVAIL4, AVAIL3, AVAIL2, NAME4(1000), NAME2(1000), NAME3(1000)
COMMON/TD/TOP, RDTM, CATLG, CHAR, HIST, PLAYER, CATLGN, PATRNS, PATRNN
COMMON/TDP3/TYPE(20), INIT(20)
COMMON/TDP4/SEQ(20), SCNTR
COMMON/TDP2/STK(10), STKM(10), STKN(10), STKNM(10)
COMMON/TDP9/TYPENU, TYPE1, TYSDU, TYNOR
COMMON/TDP8/PLHIST, NAHIST
COMMON/CLS/DEFS, RDEFS, OFFEN, BOFFN, PMOVEN, PMOVES
COMMON/JEC/VALUE(7,3), RTABN(3,3), RTABS(3,3), ATARN(3), ATABS(3),
1SN, SS, WN, WS, RS, ECNPN, NC, INRND, YRAND, EFLAG, CPN, CPS, STRATN, STRATS,
2PNOLD, ISN, ISS, ECNPN, ECUPN, TMOVES, TMOVEN, MCNT, CONTR
COMMON/TDP7/GRUPS, GROUPN, LLINK4(1000), DOWN4(1000), PLINK3(1000), DOWN2
1(1000), DOWN2(1000)
EQUivalence (NAME4(4), RLINK4(3), LLINK4(2), DOWN4(1)), (NAME2(2), DOWN
12(1)), (NAME3(3), RLINK3(2), DOWN3(1))
MOV2=NC
DCNT=1
DCNT=1
DCNT=1
MCNT=1
MOVEN=DOWN4(PATRNN)
IF(INCNT.EQ.0)GOTO 82
TOPMN=MOVEN
IF(MOV2.GT.3) GOTO 35
IF(MCNT.GT.INCNT) GOTO 20
TOP1=INIT(MCNT)
IF(NAME4(TOP1).EQ.NAME2(TOPMN)) GOTO 30
MCNT=MCNT+1
GOTO 10
20 IF(PCNTN.GT.0) GOTO 21

```

CC


```

31  POINTR=MCNT
    PCNTN=PCNTN+1
    STKN(PCNTN)=POINTR
    PTR=POINTR
    DCNT=1
    MCNT=MCNT+1
    GOTO 0
21  IF(PCNTN-GT-1)GOTO 25
    TOPI=INIT(PTR)
    MCNT=PTR
24  PMOVEN=LLINK4(TOPI)
    NORSO=1
    CALL CLSTYP(NORSO)
    RETURN
25  PTR=STKN(PCNTN)
    TOPI=INIT(PTR)
    MCNT=PTR
    GOTO 24
C   GET STRATEGY FOR THE FORTH MOVE AND SET UP
C   ADDITIONAL INITIAL MOVEN CELLS.
35  IF(PCNTN-EQ-0) GOTO 36
    PTR=STKN(PCNTN)
    PCNTN=PCNTN-1
    TOPMN=MOVEN
    DCNT=1
39  IF(DCNT-EQ-2) GOTO 38
    TOPMN=DCWN2(TOPMN)
    DCNT=DCNT+1
    GOTO 39
38  TOPI=INIT(PTR)
    MCNT=PTR
    IF(LLINK4(TOPI).EQ.NAME2(TOPMN)) GOTO 55
    GOTO 35
60  MOVE=1
    MOVE=MOVEN
    CKINCT=INCENT
    CALL INTLIB(INCENT,MOVE,TOPI,INCENT,MOV)
    IF(CKINCT-EQ-INCENT)GOTO 81
    NOW CELL HAS BEEN ADDED
    MOVEN=DCWN4(PATRNN)
    TOPMN=MOVEN
    NAME4(TOPI)=NAME2(TOPMN)
    TOPMN=DCWN2(TOPMN)
    RLINK4(TOPI)=NAME2(TOPMN)
    TOPMN=DCWN2(TOPMN)
    LLINK4(TOPI)=NAME2(TOPMN)
    DOWN4(TOPI)=0
C

```

```

35      GOTO 80
      OFFEN=0
      BYPASS=1
      MOVE=MOVEN
      CALL TOTALM(MOVE,TOTRUE,BYPASS,INCNT,MCNT,TOTAL,TOTAL2)
      IF (TOTRUE) 64,64,65
65      NORSO=1
      IF (MCNT.GT.INCNT) GO TO 650
      IF (MCNT.EQ.0) GO TO 650
      IF (DOWN4(TOPI).EQ.0) GO TO 650
      TYPENU=DOWN4(TOPI)
      CALL CLSTYP(NORSO)
      ROFFN=OFFEN
      OFFEN=0
      GOTO 60
64      OFFEN=0
      ROFFN=0
      GOTO 60
55      NORSO=1
      CALL CLSTYP(NORSO)
      GOTO 60
80      RLHIST=INIT(INCNT)
      RETURN
31      RLHIST=TOPI
      RETURN
82      IF (MOV2.GT.3) GOTO 60
      OFFEN=0
      ROFFN=0
      PMOVEN=0
      RETURN
      END

```

SUBROUTINE FINDTY(MCNT,TYPENU,NCRSO)

SUBROUTINE TO FIND A HEADER NODE IN THE TYPE LIBRARY IF A
PLAYER HAS PLAYED BEFORE AND IS IN THE CATALOGUE OF PLAYERS.

```

      IMPLICIT INTEGER (A-W)
      COMMON/T/AVAIL4,AVAIL3,AVAIL2,NAME4(1000),NAME2(1000),NAME3(1000)
      COMMON/TD/TOPI,BOTM,CATLG,CHAR,HIST,PLAYER,CATLEN,PATRNS,PATRNN
      COMMON/TDP4/SEQ(20),SCNTR
      COMMON/TDP3/TYPE(20),INIT(20)
      COMMON/JEC/VALUE(7,3),RTABN(3,3),RTARS(3,3),ATABN(3),ATABS(3),
1      ISN,SS,WN,WS,RS,RW,NC,INRAND,YRAND,EFLAG,CN,CPS,STRAIN,STRATS,
2      PNOLD,ISN,ISS,ECNXXN,ECNXX,PN,PS,FACTXN,FACTXS
      DIMENSION RLINK4(1000),LLINK4(1000),DOWN4(1000),RLINK3(1000),DOWN3
1      (1000),DOWN2(1000)

```

C
C
C

```

EQUIVALENCE (NAME4(4),RLINK4(3),LLINK4(2),DOWN4(1)),(NAME2(2),DOWN
12(1)),(NAME3(3),RLINK3(2),DOWN3(1))
IF(NC.LT.2)GOTO 20
IF(NC.GT.3) GOTO 20
TOP1=INIT(MCNT)
IF(DOWN4(TOP1).EQ.0) GOTO 20
TYPENU=DOWN4(TOP1)
RETURN
IF(NOP SO.EQ.1) GOTO 21
IF(NAME4(HIST).EQ.0) GOTO 22
INITN=NAME4(HIST)
IF(DOWN4(INITN).EQ.0) GOTO 22
TYPENU=DOWN4(INITN)
RETURN
IF(RLINK4(HIST).EQ.0) GOTO 32
INITN=RLINK4(HIST)
IF(DOWN4(INITN).EQ.0) GOTO 32
TYPENU=DOWN4(INITN)
RETURN
IF(LLINK4(HIST).EQ.0) GOTO 42
SEQN=LLINK4(HIST)
RTOP=SEQN
IF(DOWN3(RTOP).EQ.0) GOTO 43
GOTO 40
RTOP=DOWN3(RTOP)
TYPENU=NAME3(RTOP)
RETURN
IF(DOWN4(HIST).EQ.0) GOTO 42
SEQN=RLINK4(HIST)
RTOP=SEQN
GOTO 40
TYPENU=0
RETURN
END
20
21
22
40
43
32
42

```

BIBLIOGRAPHY

1. Baker, F. B., "An IPL-V Program for Concept Attainment," Educational and Psychological Measurement, v. 24, p. 119-127, Spring 1964.
2. Baker, F. B., "The Internal Organization of Computer Models of Cognitive Behavior," Behavioral Science, v. 12, p. 156-161, March 1967.
3. Beurle, R. L., "Storage and Manipulation of Information in Random Networks," Aspects of the Theory of Artificial Intelligence, p. 19-42, Plenum Press, 1962.
4. Friedell, M. F., "On the Structure of Shared Awareness," Behavioral Science, v. 14, p. 28-39, January 1969.
5. Guetzkow, H., "A Use of Simulation in the Study of Inter-Nation Relations," Behavioral Science, v. 4, p. 183-191, July 1959.
6. Lave, L. B., "Factors Affecting Co-operation in the Prisoner's Dilemma," Behavioral Science, v. 10, p. 26-38, January 1965.
7. Luce, R. D., Raiffa, H., Games and Decisions, p. 95, Wiley, 1957.
8. Messick, D. M., "Interdependent Decision Strategies in Zero-Sum Games: A Computer Controlled Study," Behavioral Science, v. 12, p. 33-48, January 1967.
9. Piaget, J., Play, Dreams, and Imitation in Childhood, Norton, 1962.
10. Piaget, J. The Growth of Logical Thinking from Childhood to Adolescence, Basic Books, 1958.
11. Piaget, J. The Origins of Intelligence in Children, Norton, 1963.
12. Rapoport, A. and Orwant, C., "Experimental Games: A Review," Behavioral Science, v. 7, p. 1-37, January 1962.
13. Strub, M. H., "Experienced and Prior Probability in a Complex Decision Task," Journal of Applied Psychology, v. 53, p. 112-117, April 1969.
14. Swensson, R., "Cooperation in the Prisoner's Dilemma Game I: The Effects of Asymmetric Payoff Information and Explicit Communication," Behavioral Science, v. 12, p. 314-322, July 1967.
15. Taber, C. W., Taber's Cyclopedic Medical Dictionary, Davis, 1965.

16. Tuttle, W. W. and Schottelius, B. A., Textbook of Psysiology, Mosby, 1965.
17. Walton, R. E. and McKersie, R. B., "Bargaining Dilemmas in Mixed-Motive Decision Making," Behavioral Science, v. 10, p. 370-384, September 1966.
18. Weil, R. L., "The N-Person Prisoner's Dilemma: Some Theory and a Computer-Oriented Approach," Behavioral Science, v. 11, p. 227-233, May 1966.

Unclassified

Security Classification

DOCUMENT CONTROL DATA - R & D		
<i>(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)</i>		
1. ORIGINATING ACTIVITY (Corporate author) Naval Postgraduate School Monterey, California 93940		2a. REPORT SECURITY CLASSIFICATION Unclassified
		2b. GROUP
3. REPORT TITLE The Modeling of Human Intelligence in the Computer As Demonstrated in the Game of DIPLOMAT		
4. DESCRIPTIVE NOTES (Type of report and inclusive dates) Master's Thesis; June 1970		
5. AUTHOR(S) (First name, middle initial, last name) James Edward Collins and Thomas Dean Paulsen		
6. REPORT DATE June 1970	7a. TOTAL NO. OF PAGES 161	7b. NO. OF REFS 18
8a. CONTRACT OR GRANT NO.		8b. ORIGINATOR'S REPORT NUMBER(S)
b. PROJECT NO.		
c.		9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)
d.		
10. DISTRIBUTION STATEMENT This document has been approved for public release and sale; its distribution is unlimited.		
11. SUPPLEMENTARY NOTES		12. SPONSORING MILITARY ACTIVITY Naval Postgraduate School Monterey, California 93940
13. ABSTRACT The purpose of this thesis is a discussion of developing human-like behavior in the computer. A theory of the human learning processes is first described. This leads to the presentation of a computer game which simulates the human capabilities of reasoning and learning. The program is required to make intelligent decisions based on past experiences and critical analysis of the present situation.		

Unclassified

Security Classification

14 KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Artificial Intelligence						
Brain Model						
Computer Game Playing						
Information Structures						
Management Decision Simulation						
Thinking Machine						