COMPUTER TECHNOLOGY

A 4-K computer language for experimentation with human subjects*

THEODORE J. DOLL

State University of New York at Stony Brook, Stony Brook, New York 11790

This paper describes a conversational language for programming psychological experiments on small computers. The language can handle experiments of considerable complexity or with a large number of stimuli in less than 4-K core storage. The language is convenient for human experimentation because it can evaluate verbal responses and store and selectively display verbal stimuli.

Small computers are increasingly being used for on-line control of psychological experiments. Among the most widely used computers, and the one to which this paper is particularly relevant, is the PDP-8 series by Digital Equipment Corporation (DEC).

Although small computers have a great potential for streamlining research, their use has been retarded, until recently, by the lack of a convenient, high-level programming language. The past few years have seen the introduction of a number of languages intended to fill this gap. The new languages include ACT (Lehigh Valley Electronics, 1968), SCAT (Grason-Stadler, 1970), FOCAL (DEC, 1970), and PSYCHOL (McLean, 1968, 1969). The ACT and SCAT languages are too limited in their capability to present verbal stimuli and in their power to interpret responses to be of very general use in experimentation with human subjects. As a result, ACT and SCAT have found their principal use in animal conditioning studies. The FOCAL language is excellent for computational problems and can be adapted to control experiments. However, depending on how it is adapted, FOCAL may suffer some of the same difficulties as ACT and SCAT. In addition, FOCAL will require more than 4 K of core storage for all but the very simplest of experiments. The last possibility mentioned above, PSYCHOL, is the most attractive language for controlling experiments but requires a relatively large computing system (e.g., 16 K words of core storage and a disk).

This paper describes a language called HECL (Human Experimentation Control Language) which is capable of controlling many kinds of human

experimentation and requires less than 4-K core storage. HECL has been used to implement studies in reaction time, verbal learning, and information processing and memory. It could also be applied in such other areas as verbal conditioning, problem solving, and psycholinguistics. HECL can handle a wide variety of experimental paradigms because its terminology is similar to the natural language used to describe experimental procedures. Some of the most important features of HECL include: (1) the ability to evaluate responses and to provide feedback according to the customary criteria (e.g., free, serial, and probed recall, "yes-no" recognition, paired-associates learning, choice reaction-time, etc.); (2) a chronologically organized memory for past stimuli which can be altered selectively; (3) automatic random selection of stimuli from the chronological memory or from other specifiable pools; (4) the ability to branch and form loops in any desired configuration; (5) the ability to branch probabilistically; and (6) the ability to program real-time modifications of the object (source) program. The current version of HECL is designed for single-terminal use.

EQUIPMENT REQUIREMENTS

In its present form, HECL uses a PDP-8/L computer with 4-K core storage, a magnetic tape drive for data collection, an ASR-33 Teletype, and a subject station. The subject station includes a device capable of simultaneously displaying up to 16 alphanumeric characters, and a keyboard on which the subject can enter alphanumeric responses. The display unit consists of two horizontal rows of eight "NIXIE" tubes each (Burroughs Corporation). The response keyboard is a Teletype Corporation Model 33.

The minimal set of hardware required to implement HECL consists of (1) a 4-K computer, (2) a peripheral

device for data collection (e.g., magnetic tape drive, paper-tape punch), (3) a stimulus display, and (4) a response apparatus. The stimulus and response devices should have alphanumeric capabilities in order to take advantage of HECL's ability to handle verbal materials. (HECL could, of course, be used to drive a CRT display with appropriate alterations of its display routine.) One simple possibility is to use a Teletype machine as both the display and response apparatus. If nonalphanumeric stimuli are required, a random-access slide projector could be used in conjunction with a Teletype. The HECL program is easily modified to handle a Teletype and slide projector.

PROGRAM EXAMPLE

Table 1 contains a HECL object (source) program for a short-term recognition memory experiment (e.g., Sternberg, 1969). On each trial of this experiment the subject views a series of 1, 2, or 4 different letters and is tested immediately for recognition. He is to respond "Y" (yes) if the test item was in the series and "N" (no) if it was not. The object program consists of six major segments which contain, respectively: (1) parameters, such as the number of characters per stimulus and the unit of measurement for response latencies, (2) all the time intervals to be used in the experiment, (3) a list of instructions, (4) all the stimuli to be used, (5) a list of all admissible responses, and (6) numbers which indicate how the stimuli are to be mapped onto responses. The lists of responses and S-R mappings are used to evaluate responses for correctness in certain types of experiments (see the later discussion of the F instruction). The responses and S-R mappings are generally not needed in those types of experiments where the number of admissible responses is large or the S-R mappings are very complex.

The most important part of the sample program in Table 1 is the list of instructions, Segment C. For the sake of discussion, the instructions will be broken down into four functionally distinct sections. The initial instructions from Line 1 through Line 7 warn the subject that the experiment is about to begin and allow him to initiate the procedure. They

*A more detailed description and listing of the program can be obtained by writing Theodore J. Doll, Department of Psychology, State University of New York at Stony Brook, Stony Brook, New York 11790.

also alert the subject at the beginning of each trial by presenting the word "ready" and perform certain initialization functions. The second group of instructions on Lines 8-20 displays a series of 1, 2, or 4 different stimuli, each series size occurring with probability 0.333. The third section of instructions runs from Line 25 through Line 37. These instructions determine whether to display a negative or positive test probe, then accept the response, and then provide immediate feedback on whether the response was correct or incorrect and "fast" or "slow." The fourth group of instructions, Lines 39-43, causes the machine to loop back to start a new trial or creates a rest period before the next trial.

The following paragraphs describe the function of each type of instruction in the context of the sample program (Table 1). It may be helpful to the reader to refer also to Table 2, which summarizes the function of each instruction in general terms.

The first instruction, S, stands for "randomly select a stimulus from the list of stimuli (Segment D) and display it." The numbers following the S refer to ordinal positions in the list of stimuli. The selection is random within the limits specified by these numbers. In this case, the stimulus in Position 11 (BEGIN) will be selected because the limits both equal 11. The stimulus remains on the display until a new stimulus is presented. The only instructions which affect the display are those that begin with the letters S, N, or O. In this program, "BEGIN" remains on the display until "READY" is displayed on Line 5.

The next instruction, R, causes the machine to wait until a response is emitted. The first number after the R ordinarily indicates the maximum time to wait for the response before going to the next instruction. However, when this number is zero there is no limit to how long the machine will wait. The second number indicates the number of response characters that will be accepted before going on.

The third instruction, D, creates a data record of 5 bytes and also initializes the temporary data storage area (see the "Data Recording" section for further details).

The fourth instruction, I, selects a time interval from Segment B in the same fashion that the S instruction selects stimuli, and then executes it.

The next instruction requiring explanation is the E on Line 7. This instruction erases a portion of the chronological memory of previous stimuli presented. Each instruction which displays a stimulus (S, O, and N) also automatically deposits a record

Table 1
An HECL Program for a Short-Term Recognition Memory Experiment

Segment	Line	Description	Object	Program
Parameters				
A	1	Characters/stimulus	0005	
	2	Max characters/response	0001	
	3	Unit of measure for intervals*	3891	
	4	Unit of measure for latencies*	2016	
Time Intervals				
B	1	2 Sec	0200	
	2	1 Sec	0100	
	3	.5 Sec	0050	
	4	2.5 Sec	0250	
Instructions				
C	1	Display "BEGIN"	S 0011	0011
	2	Wait for response	R 0000	0001
	3	Write 5 bytes of data**	D 0000	0005
	4	Execute time interval 4	I 0004	0004
	5	Display "READY"	S 0012	0012
	6		I 0002	0002
	7	Erase chronological memory	E 0001	0013
	8	Display one of first 10 stimuli	S 0001	0010
	9		I 0003	0003
	10	Substitute 1 for third item in line 28	M 0084†	0001
	11	Branch to line 21 with p = .333	B 0021	0333
	12	Display "new" stimulus	N 0001	0010
	13		I 0003	0003
	14	Add 1 to third item in line 28	M 1084†	0001
	15	Branch to 21 with p = .500	B 0021	0500
	16	Display "new" stimulus	N 0001	0010
	17		I 0003	0003
	18	Display "new" stimulus	N 0001	0010
	19		I 0003	0003
	20	Add 2 to third item in line 28	M 1084	0002
	21	Display blank	S 0013	0013
	22		I 0001	0001
	23	Display "PROBE"	S 0014	0014
	24		I 0002	0002
	25	Branch to 28 with p = .500	B 0028	0500
	26	Display "new" (negative) probe	N 0001	0010
	27	Jump to 29	J 0029	0000
	28	Display "old" (positive) probe	O 0001	0000
	29	Wait 2 sec for response	R 2000	0001
	30	Evaluate response and jump to:	F 1100	0600
	31	Here if it is correct and fast	S 0015	0015
	32		J 0038	0000
	33	Here if it is correct and slow	S 0016	0016
	34		J 0038	0000
	35	Here if it is incorrect and fast	S 0017	0017
	36		J 0038	0000
	37	Here if it is incorrect and slow	S 0017	0017
	38		I 0002	0002
	39	Display blank	S 0013	0013
	40	Write 21 bytes of data	D 0000	0021
	41	Do lines 4-40 fifty times, then go on	J 0004	0050
	42	Display "REST"	S 0018	0018
	43	Do lines 2-42 six times, then go on	J 0002	0006
	44	Display "END"	S 0019	0019
	45	Terminate experiment	J 0000	0000
Stimuli				
D	1	("\" stands for blank)	\\A\\	
	2		\\B\\	
	3		\\C\\	
	4		\\D\\	
	5		\\E\\	
	6		\\F\\	
	7		\\G\\	
	8		\\H\\	
	9		\\I\\	
	10		\\J\\	
	11		BEGIN	
	12		READY	
	13		\\\\\\\\	
	14		PROBE	
	15		\\FAST	
	16		\\SLOW	

Table 1 (Continued)

Segment	Line	Description	Object Program
<i>Stimuli (Continued)</i>			
	17		WRONG
	18		\REST
	19		\END\
<i>Responses</i>			
E	1	Y for "yes"	Y
	2	N for "no"	N
<i>S-R Mappings</i>			
F	1	(Not used in this experiment)	

*Timing is accomplished by cycle counting; 3,891 and 2,016 are the number of repetitions of a set of instructions which consume a hundredth of a second and a millisecond, respectively.

**In this instance the D instruction is used to clear and initialize the temporary data storage area.

†The item to be modified is addressed by "item number" rather than by line number. Since there are three items per line (i.e., a letter and two 4-digit numbers), the item number is 84 in this case.

of the stimulus in the chronological memory. Each trial in the present experiment generates no more than 13 entries in the chronological memory; hence only Positions 1-13 need be erased.¹

The instructions on Lines 8-20 generate the series of stimuli on which the subject is later tested. The S instruction on Line 8 randomly chooses one of the first 10 stimuli and displays it. The next instruction, I,

Table 2
Summary of HECL Instructions

Instruction			Function
B	1111	m m m m	Branch to the 1111th instruction with probability m m m m; otherwise go on to the next instruction.
D	1111	m m m m	Create a data record of size m m m m bytes.
E	1111	m m m m	Erase positions 1111 through m m m m of the chronological memory.
F	k l m m	n n n n	Provide feedback about the last response by jumping ahead to the next, third, fifth, or seventh instruction (see text for explanation).
I	1111	m m m m	Randomly select and execute a time interval from among positions 1111 through m m m m in the list of time intervals.
J	1111	m m m m	Jump to the 1111th instruction m m m m - 1 times; then go on to the next instruction. (m m m m = ϕ causes an unconditional jump to 1111; 1111 = ϕ terminates the experiment.)
M	k 111	m m m m	Modify the 111th item in the list of instructions; k specifies the operation (ϕ = substitute, 1 = add, 2 = subtract) and m m m m is the argument.
N	k*111	m m m m	Randomly select and display a "new" stimulus from among positions 111 through m m m m in the list of stimuli. "New" means that the stimulus does not appear in the chronological memory.
O	k*111	m m m m	Randomly sample and display an "old" stimulus from among positions 111 through m m m m in the chronological memory. (Note that the N and S instructions sample the list of stimuli.)
R	1111	m m m m	Wait up to 1111 milliseconds† for m m m m response characters (1111 = ϕ makes the waiting period unlimited).
S	k*111	m m m m	Randomly select and display a stimulus from among positions 111 through m m m m in the list of stimuli.
X	1111	m m m m	Execute user-supplied routine.

Note—The letters k, l, m, and n represent numbers in an actual program.

*The k in N, O, and S instructions specifies the position of the stimulus display (k = ϕ means the upper row of NIXIE tubes, k = 3 means the lower row).

†The unit of measure of 1111 can be changed to hundredths of a second by altering a parameter in the object program (Line 4, Segment A, Table 1).

creates a time interval of .5 sec. The M instruction on Line 10 records the fact that one stimulus was presented by appropriately modifying a later instruction (the reason for this will become clear later). The instruction on Line 11, B, decides whether to present another stimulus or branch directly to Line 21 to present the test probe. If the decision is not to branch, control is transferred to Line 12, where a second stimulus is presented.

The N instruction on Line 12 functions exactly like the S instruction, except that the selected stimulus is always "new," i.e., not a repetition of the stimuli already in the chronological memory. Of course, the composition of the set of "new" stimuli depends upon the manner in which the chronological memory was manipulated by the E instruction.

After the second stimulus is displayed, the record of the number of stimuli presented is updated by the M instruction on Line 14. Then, on Line 15, another B instruction decides whether to present two more stimuli or branch directly to Line 21 to present the test probe.

On Line 21 an S instruction is used to erase the display; on Line 23 the word "PROBE" is presented as a warning signal; and on Line 25 a decision is made whether to present a negative or positive test probe. If the probe is to be positive, control is transferred to the O instruction on Line 28. The O functions like the S instruction, except that the stimulus is sampled from the chronological memory rather than the list of stimuli. The numbers which follow the O refer to positions in the chronological memory (1 = oldest). In this instance, the second number is given a value of zero. The proper number is filled in by the M instructions on Lines 10, 14, and 20 as the program runs. This procedure is used because there is no a priori way of telling whether 1, 2, or 4 stimuli will be presented, and hence from how large a pool the positive probe can be selected. If the second number in the O instruction were always set to 4, an empty position in the chronological memory would sometimes be sampled, in which case a blank would appear on the display.

The F instruction on Line 30 is the most complex. It evaluates the correctness of the response according to one of three scoring criteria: (1) reproduction (recall tasks), (2) coding (any task which can be conveniently described in terms of S-R mappings), and (3) "yes-no" recognition. The F instruction also evaluates the response latency as "fast" or "slow." The criterion for judging the latency is the second number in the F instruction. The

machine branches to the first, third, fifth, or seventh line after the F when the response is, respectively, correct and fast, correct and slow, incorrect and fast, or incorrect and slow.

The criterion for judging correctness is specified by the first number of the F instruction. The present case (Line 30, Table 1) is an instance of the "yes-no" recognition criterion. The digits of the first number serve different functions. The leftmost digit specifies how many times the probe stimulus must have been repeated in order for the appropriate response to be "yes." In almost all cases this will be 1. The second digit specifies which of the stimuli in the chronological memory is the probe. A value of 1 means the newest item is the probe. The last two digits are used only in recall tasks and are therefore given a value of zero here. When using the "yes-no" recognition criterion, the correct "yes" and "no" responses must be listed in order in Segment E.

Another way of judging correctness is the reproduction criterion (not illustrated in Table 1). It is implemented as follows: The two rightmost digits of the first number in the F instruction are set to a value, X, between 1 and 62. The response is considered correct only if it reproduces the stimulus in Position X in the chronological memory. Serial recall scoring is accomplished by repeatedly calling the F instruction and incrementing X before each call. When X = 63, a free-recall criterion is in effect. The response is considered correct if it matches any of the stimuli in the chronological memory. Probed recall (e.g., Waugh & Norman, 1965) is represented as a sort of hybrid of "yes-no" recognition and reproduction. The details, however, are too lengthy to cover here. A manual containing this and other more technical information can be obtained from the author.

The third and last method for scoring responses is the coding criterion. In order to use this option, all admissible responses must appear in the response segment of the object program. The user must also specify the S-R mappings in the last segment of the object program. The S-R mappings are best explained by an example. Suppose the first response in the list is appropriate only after the first, second, third, or fourth stimulus has occurred, and the second response is appropriate only after the third, fourth, or fifth stimulus has occurred. Then the S-R mapping segment should read 0001 0004 0003 0005. Successive pairs of numbers indicate the range of stimuli for which each response is appropriate. A response may appear more than once in the list of responses

and may have a number of S-R mapping entries. A response is scored "correct" if it has been preceded by one or more of the stimuli specified by its S-R mappings. The program decides which stimuli have preceded a response by examining the chronological memory.

The last instruction which requires explanation is the J (e.g., Line 40). This instruction is used to perform simple "jump" operations and to form loops. The first number after the J is the line number to jump to or the first line of the loop. The second number is zero for simple jumps or, in the case of loops, specifies the total number of times the loop is to be executed. The J instruction itself is always the last instruction in a loop. Loops and jumps may be programmed in any desired configuration.

In addition to the instructions shown in Table 1, there is a dummy instruction, X, which allows the user to add his own special-purpose routines. Of course, the use of this option requires some knowledge of the machine language, PAL (Program Assembly Language).

DATA RECORDING

The data are temporarily stored and output in raw form. Data accumulate in a "data buffer" area of core until a data record is created by executing the D instruction. The data record consists of 8-bit bytes. The number of bytes is specified by the second number in the D instruction.

The first byte of each data record is automatically filled by a number which the experimenter assigns to the subject at the beginning of the experiment. The subsequent bytes contain a chronological record of experimental events, including: (1) certain stimuli and time intervals, (2) all probabilistic branching operations, and (3) all responses and response latencies. Stimuli and time intervals appear in the data record only if there is uncertainty about which stimulus or interval actually occurred. For example, the instruction S 0001 0010 displays any one of 10 stimuli; therefore, a record of the stimulus which actually occurred must be included in the data. On the other hand, an instruction such as S 0011 0011 leaves no uncertainty about which stimulus was displayed and no record of this stimulus is included in the data. It is hard to imagine a case in which redundant information of this sort would be important in the analysis of the data. Therefore, nothing is lost by excluding it from the data record, and the benefit is a more compact data record.

The capacity of the data buffer is 128 bytes. Each occurrence of a

stimulus, time interval, and probabilistic branching operation creates 2 bytes of data. The execution of an R instruction creates 4 bytes of data per each response character called for. Two of the 4 bytes contain the character itself (or zero if no response was made) and the other two bytes contain the latency for the response character. The data buffer is large enough to hold a complete record of a single trial for almost any experiment.

In the present system configuration, the data are output in two forms to guard against loss of data in the event of a hardware failure. The output appears in binary form on magnetic tape and in ASCII² code on a Teletype. The Teletype, of course, converts the ASCII code to alphabetic and numeric characters. Thus the data can be viewed directly by the experimenter. At the experimenter's option, the Teletype can also produce a punched paper tape in ASCII format. Data in this form are conveniently analyzed by using FOCAL.

PROGRAM ORGANIZATION

The bulk of the HECL program consists of nine interpretive subroutines. Each subroutine carries out the function prescribed by one or more of the instructions in the programming language. The HECL program also contains a preprocessor, several general utility routines and, of course, input/output routines.

The preprocessing routine translates the object program into binary code, which is then stored in the machine. Each of the parameters, time intervals, and other decimal numbers is stored as a 12-bit binary number. The letters which initiate instructions are stored as subroutine calls. Stimuli and responses are converted to ASCII trim code and packed into core at the rate of 2 characters per 12-bit word. When the experiment is running, the coded object program acts as an executive routine.

The total core space available for the coded object program (assuming a 4-K PDP-8 type of computer) is 2,160 locations. This figure does not include the highest page (128 locations) of core. The highest page is reserved for a program-loading routine, such as DEC's Binary Loader. The various segments of the object program all compete for the 2,160 locations. Each time interval requires one location; each instruction (including the letter and two numbers) requires three locations; each stimulus and response character requires one-half location; and each feedback number uses one location. The total size of the object program is, therefore, $T + 3I + \frac{1}{2}S + \frac{1}{2}R + M$ locations, where T = the number of time intervals, I = the

number of instructions, S = the number of stimulus characters (plus 1 if odd), R = number of response characters (plus 1 if odd), and M = number of S-R mapping numbers. The HECL object programs are quite compact in relation to the amount of space available for storage. For example, the object program in Table 1 requires only 188 locations; hence, it uses less than 9% of the available core space.

The general utility routines in HECL are called internally by the routines associated with the various instructions. The utility routines include a pseudorandom number generator, a routine that searches internal lists (such as the chronological memory) for a specified bit pattern, and another routine that retrieves information from certain lists (such as the list of stimuli) and loads the information into a software buffer area.

The pseudorandom number generator is especially important because it controls the selection of

stimuli and, in some cases, the sequencing of other experimental events. The generator is based on routine written and tested by Griffith (1969). The algorithm has also been tested by Brady (1965). Griffith found a statistically significant bias in a length of runs test. The bias, however, was relatively small when compared to those of other generators. Griffith also conducted a test for uniformity, and found "an extremely even distribution of the numbers generated." Although the number generator is demonstrably nonrandom, it performs satisfactorily for the present application.

REFERENCES

- BRADY, P. T. A pseudo-random number generator for the PDP-5 computer. Decus Program Library, No. 5-25, 1965.
- Digital Equipment Corporation. Focal-8: Programming manual (No. DEC-08-AJAD-D). Digital Equipment Corporation, Maynard, Massachusetts, 1970.
- Grason-Stadler. SCAT user's manual. Grason-Stadler Company, Inc., West Concord, Massachusetts, 1970.
- GRIFFITH, G. A. A pseudo-random number generator for the PDP-8 for use

- with FOCAL. Decus Program Library, No. 8-194, 1969.
- Lehigh Valley Electronics. The ACT primer: Computer language and hardware interface for controlling psychological experiments. Lehigh Valley Electronics, Inc., Fogelsville, Pa., 1969.
- McLEAN, R. S. PSYCHOL: A computer language for experimentation. Behavior Research Methods & Instrumentation, 1969, 1, 323-328.
- McLEAN, R. S. PSYCHOL: A formal language for the control of psychological experiments. Complex Information Processing Working Paper No. 120, Department of Psychology, Carnegie-Mellon University, 1968.
- STERNBERG, S. Memory-scanning: Mental processes revealed by reaction time experiments. American Scientist, 1969, 57, 421-457.
- WAUGH, N. C., & NORMAN, D. A. Primary memory. Psychological Review, 1965, 72, 89-104.

NOTES

1. Upon the erasure of items in the chronological memory, any remaining items "pop up" to fill any vacancies at the top (Position 1) of the list. Subsequent entries are made at the bottom end of the list. This property is useful in generating certain types of stimulus sequences.
2. ASCII is an abbreviation for American Standard Code for Information Interchange.