Third Misconceptions Seminar Proceedings (1993)

Paper Title: A study on the Analysis of Error Patterns and Misconceptions for BASIC Programmings of Novice College Students in Taiwan
Author: Chang, Bor-nian

Abstract: To use the theories of artificial intelligence and cognitive science in designing the intelligent computer-assisted instruction (ICAI) systems is a challenging job with much research value. This research collected different types of error patterns that novice learners had during computer programming. These error patterns were analyzed and categorized. The misconception of semantic errors during the programming were also analyzed. The outcome will serve as the base knowlege for computer programming ICAI system design.

Keywords: Concept Formation,Testing,,Misconceptions,Error Patterns,,,,
General School Subject: Information Science
Specific School Subject: Programming
Students: College Freshmen

Macintosh File Name: Chang - Basic Programming
Release Date: 12-15-1993 C, 11-4-1994 I

supporting our ability to support you by purchasing a subscription to the collection or joining the Meaningful Learning Research Group.

----- -----

A study on the Analysis of Error Patterns and Misconceptions for BASIC Programmings of Novice College Students in Taiwan

Bor-nian Chang, Associate Professor
Department of Mathematics and Science Education
National Taichung Teachers College, Taiwan, R.O.C.

INTRODUCTION

To use the theories of artificial intelligence and cognitive science in designing the intelligent computer-assisted instruction (ICAI) systems is a challenging job with much research value. This research collected different types of error patterns that novice learners had during computer programming. These error patterns were analyzed and categorized. The misconception of semantic errors during the programming were also analyzed. The outcome will serve as the base knowlege for computer programming ICAI system design.

Errors in a computer program are called bugs; removing these errors is called debugging. Common to BASIC and all programming languages is a set of syntax rules. If in using BASIC, each statement is not written according to a specific set of rules, a syntax error occurs. Many errors in programming language are due to misconceptions of syntax and semantic errors of the programming language (Sleeman & Gong, 1985). Trying to find an error in a seemingly perfect program can be frustrating (Bitter, 1984). When an error occurs, the computer is not "thinking" correctly, because its logic is faulty. The time spent in debugging all too frequently exceeds the time need to write program code.

The using of traditional computer assisted instruction (CAI) method to design teaching systems that provide instructional feedback is acknowledged to be a very difficult task, especially for adapting to the needs of individual students. Applying the knowledge of artificial intelligence (AI) and cognitive science to the design of CAI system hold much promise for the improvement of individualized teaching systems on which computer programming instruction relies (Sleeman & Brown. 1982). An important component of any programming course is a method for representing, diagnosing, and correcting errors in computer programs, commonly called bugs (Stevens et al, 1981).

Designing good debugging software using the knowledge of AI and cognitive science to help students to correct their syntax errors represents an important step toward designing future programming course using CAI (Miller, 1982).

A few automatic or intelligent debugging programs are currently available to teach programming. Several of the intelligent tutoring systems with considerable debugging ability are still in developmental stages. Program Understander for student (PROUST) is a set of program debugging software constructed using the List Processing (LISP) programming language. It find semantic bugs in Pascal programs written by novice programmers (Johnson & Soloway, 1985). After students correct every syntactic error from complied programs, bugs resulting from conceptual misunderstanding may still remain. These bugs are the most difficult to correct. When PROUST find nonsyntactic errors, it not only identifies the line of code that are faulty, but also determines how the bugs can be corrected (which is most beneficial to students). The aim of PROUST is to build an instructional system that assigns programming problems to students, analyzes their work, gives then helpful suggestion.

Goal-Restricted Environment for Turtoring and educational Research on Programming (GREATERP) is an computer-based LISP tutoring system that combines AI technology and a psychological theory of skill acquisition into a program that serves as an effect human tutor (Anderson & Reiser, 1985). The assumption underlying the design of the LISP tutor is that a student should be able to work on a programming problem in a friendly environment. Whenever the student makes a programming errors or asking for help, the tutor provides useful information to bring student back on to a path toward a solution. Problem-solving becomes a more effective learning experience.

The majority of cognitive psychologist in America today follow a theoretical framework known as information processing (Anderson, 1982; Gagne, 1985; Williams & Hollan, 1981). Human cognition is developed in the process of transferring information from senses to the brain and vice versa. In this model information is received in the form of some physical energy by receptors that are sensitive to that particular form of energy. From the receptors the nerve impulse goes to a sensory register in the central nervous system. There appear to be different sensory registers for each sense. The working memory (WM) is the place where conscious mental work is done. It is also call "short-term" because this memory retains information for about ten seconds (Murdock,1961). Information in working memory may be coded; it is then stored in long-term memory (LTM). Coding is a transformation process in which new information is integrated in various ways with known information. Once information is stored in LTM, it must be retrieved in order to be used.

Information is said to flow from LTM to WM and then to the response generator, which is a place in human memory to store retrieved information. The response generator organizes the response sequence and guide the effectors (i,e. arms and hands). The expectancy and executive control (both are located inside the brain and arrange the flow of information in the human system to be purposeful and organized) govern the stratrgies to reach a goal.

Information processing systems that can transform data from input to output. Most computers contain at least the follow following parts (a) input device, (b) output device, (c) central processing unit, and (d) memory (Bitter, 1984); so the computer is example of an information processing system (Swanson, 1982).

In the information processing model, knowledge is stored by means of propositions and productions. Propositions are similar to facts and productions are similar to rules. Propositions are linked to builld up the Propositional network that form one's declarative knowledge to represent factual information. Productions are procedural knowledge that represent the rule to process or link the data (Anderson, 1982). In other word, declarative knowledge is knowledge that something is the case (fact), and procedural knowledge is represented by productions and enable a person to act (Gagne, 1985). Anderson (1986) assumes that all of an individual's declarative knowledge is represented in a propositional network. The WM is the place to code new knowledge and add it to old knowledge; the LTM includes knowledge of interrelated propositions and productions and store this information for later use. Knowledge is first learned as declarative knowledge. The newly formed propositions are linked to the current propositional network to form a new propositional network through WM. The limitations of the working memory cause many learning errors. This is particularly true when learning to program a computer. Novice programmers first learn syntax rules in order to write correct programming statements and run the program. An overloaded WM may produce inappropriate linkages among newly learned syntax rules and form incorrect productions, which may later result in faulty procedural knowledge (Mayer, 1985). Productions are similar to IF-THEN statements that are found in computer languages. Procedural knowledge represents rules to the process the declarative knowledge or information. Productions can be divided into two categories--pattern recognition procedure and action-sequence procedures. Pattern-recognition procedures are the conditional part of a production, and action-sequence

procedures are the actions to be performed if conditions in the pattern-recognition part are met (Gagne,1985).   Practice with feedback is more appropriate for learning procedural knowledge than for learning declarative knowledge, for which elaborations and analogies might be more effective. Syntax rules for programming in BASIC are first learning as declarative knowledge then later are stored as procedural knowledge.

Novice learner must form many interrelated propositions while learning to program in BASIC. Although some BASIC statements appear to be simple, they actually require learners to build many procedures. One incorrectly formed procedure may mislead subsequently build procedures and generate misconceptions which are stored in LTM. Learners must know a series of transactions to understand BASIC statements.   After one group of learners completed a six hour mastery course in BASIC, they failed to correctly form a conceptual model of function of BASIC statements (Mayer, 1985). Soloway and Ehrich (1984) listed several rules of convention which experienced programmers expected to find in programming.   Their study showed experienced programmers to be more affected by violations of programming conventions than novice programmers.   This discrepancy may indicate that novice programmers need more correct programming statements to follow in order to finish the program.   If syntax error occurred while writing programming statement, novice programmers might not be able to complete the program. Spock (1986) conducted research in learning BASIC statements and summarized reasons that learners may fail to learn BASIC: (a) novice programmers have to store simultaneously many new prerequisite propositions and productions in memory, which often overload the working memory; (b) novice programmers miss one of the many prerequisite procedures which a single BASIC statement involves; and (c) novice programmers are misled by previously learned propositions and procedures that occur in ones daily communication.

RESEARCH PROCEDURE

Seventy-two students from two classes of freshmen who took "Information Science" course in the Department of Math and Science Education were invited as samples for collecting error patterns during BASIC programming exercises for the first year. The inquirying and recording methods that are frequently used for collecting error patterns in cognitive science were used. The handout for collecting error patterns and the programming exercises

were made and implemented under proper schedule. Those BASIC instructions such as READ, DATA INPUT, PRINT, LET, GOTO, IF..THEN were used as test instructions for programming. After students first learned above BASIC instructions, an error pattern collecting multiple choice test was given (more than one answer were acceptable). Students might have correct or incorrect answers. Results showed what they thought those instructions were.

I.		number of students	percentage
	INPUT A		
1.	print "?" on the screen	17	23.6%
2.	waiting to input a number	36	50.0%
3.	input a number to A	29	40.3%
4.	write A into memory	21	29.2%
5.	waiting to input a number or a character	27	37.5%
6.	print A on the screen	6	8.3%
7.	others:	3	4.2%
<1>	input A		
<2>	A=inputing number		
<3>	must input a number or character		
8.	no answer	2	2.8%

II.
30 READ A

		number of students	percentage
1.	get a number from DATA to memory	41	56.9%
2.	print A on the screen	10	13.9%
3.	write character "A" into memory	15	20.8%
4.	waiting number input from keyboard	4	5.6%
5.	others:	14	19.4%

<1> get A sequentially from DATA
<2> read A from DATA area
<3> read A
<4> read data A
<5> get A from memory
<6> read data from DATA A
<7> read data from memory A
<8> get data from A
<9> same as <2>
<10>read data that stored in DATA
<11>same as <7>
<12>same as <4>
<13>read data from file A
<14>get data from file A

III.
IF A<B GOTO 99

		number of students	percentage
1.	if A less than B then goto line 99	68	94.4%
2.	if A greater or equal to B then go to next line	37	51.4%
3.	goto line 99 (without comparison)	0	0%
4.	print 99 on the screen	1	1.4%
5.	write number A or B into memory	5	6.9%

6.	others	0	0%

IV.

LET A=B+1

1.	write result of B+1 into memory	11	15.3%
2.	write A=B+1 this expression into memory	49	68.1%
3.	write B+1 into memory	22	30.5%
4.	print character A or result of A on the screen	11	15.3%
5.	others:	2	2.8%

<1>   let A equal to B+1
<2>   if give a value to B then will
     get the value of A

6.	no answer	1	1.4%

V.

20 DATA 80,90,99

1.	put above numbers into input queue	50	69.4%
2.	input numbers into memory	39	54.2%
3.	print numbers on the screen	10	13.9%
4.	input number into memory A	8	11.1%
5.	others:	2	2.8%

<1>   input data

VI.

60 GO TO 30

1.	jump to line 30	60	83.3%
2.	execute from line 30	40	55.6%
3.	fine the number 30	3	4.2%
4.	if not equal to A then goto line 30	10	13.9%
5.	print line 30 on the screen	0	0%
6.	others:	5	6.9%

<1>   no space between GO TO
<2>   goto line 30 then run
<3>   jump to line 30 and quit
<4>   sequentially execute the program to line 30
<5>   execute to the end then jump back to line 30

VII.

LET D=0

1.	write 0 into memory D	22	30.6%
2.	wrote D=0 this expression into memory	47	65.3%
3.	write D or 0 into memory	6	8.3%
4.	print D=0 on the screen	6	8.3%
5.	others:	1	1.4%

<1>   let D equal to zero

6.	no answer	1	1.4%

VIII.

PRINT "C"

1.	print character C on the screen	57	79.2%
2.	write character C into memory	7	9.7%
3.	print C on the screen	14	19.4%
4.	get a value from memory C	5	6.9%
5.	others:	0	0%

IX.

PRINT C

1.	print number or 0 on the screen	22	30.6%
2.	print character C on the screen	12	16.7%
3.	write character into memory	11	15.3%
4.	print "ERROR" on the screen or print nothing	24	33.3%
5.	others:	18	25.0%

<1>   depending on the definition of C,
       if C is a expression print its
       value on the screen
<2>   if C is an expression then
       calculate its result
<3>   print the value of variable C
<4>   print C's value on the screen
<5>   no answer
<6>   print variable C on the screen
<7>   print out the value inside varible
<8>   C should have no quotation onto it
<9>   print a number or a string
       standing for C
<10>   show data of memory C on the screen
<11>   execute C then display its result
<12>print out the data inside variable C
<13>   print out the number of C on the screen
<14>   getnt data of C
<15>   print out the result

Twenty students who didn't perform well had been invited to take following programming exercises. Each student took three paper and pencil proramming exercises and one on-line programming exercise. These three paper and pencil programming exercises had been randomly selected from twelve test exercises. Students were allowed to check with the book if needed. Assistants asked students to explain their designing process and traced and recorded their programs and output while they did the exercises. Exercises and students' results were listed below.

Paper and Pencil Test:
    01.
    CLS
     NAME$="Tom"
     CHINESE=90
     ENGLISH=85
     TOTAL=CHINESE+ENGLISH
      PRINT NAME$,CHINESE,ENGLISH,TOTAL
    END

    1.   NAME$,CHINESE,ENGLISH,TOTAL
    2.   Tom,90,85,175
    3.   "Tom",90,85,175
    4.   Tom 90 85 175
    5.   Tom
          90
          85
          175

6. can't print string and number at the same time
7. other_____

Degree of difficulty:
   Is this test too difficult? <No 1 2 3 4 5 Yes>

Check out the following questions if you have.
_ What data type NAME$ is? (string or number)
 _ What datatype CHINESE and ENGLISH are?
_ What is the function of PRINT?
 _ What is the effect of "," on PRINT?
_ What is the output data type for string and number?

Write down other questions that you have: _____

Students' average of degree of difficulty: 1.67 (between very simple
and simple)
Student asked about the meaning of "$" and its effect to variable "NAME". Would "TOTAL=CHINESE+ENGLISH" do sum up scores? What the effect of "," and ";" to "PRINT"? Some errors occurred such as: (1) The quotation marks would be printed out simultaneously when the string was printed out. (2) "," or ";" would be printed out with string or number. (3) ";" would create intervals while "," add a new seperate line.
02.
  CLS
   INPUT "PLEASE KEY IN 2 NUMBERS:",A,B
   PRINT "A+B","A-B"
  PRINT A+B,A-B
  END
 [8,4]

Degree of difficulty:
   Is this test too difficult? <No 1 2 3 4 5 Yes>

Check out the following questions if you have.
_ What is the function of INPUT ?
_ What are data types of A, B ?
_ What is the function of PRINT ?
 _ What is the effect of "," on PRINT?
_ What are the output data types of string and number?

Write down other questions that you have: _____

Students' average of degree of difficulty: 3.8 (between medium and
difficult)
Some errors occurred such as: (1) "INPUT A,B" input string "A,B" instead of two numbers "8,4". (2) Some students didn't know that why put quotation mark on A+B and A-B; they thought "A+B" and "A-B" can be calculated. (3) "," would seperate results into different lines.

03.
  CLS
  Q=9
   INPUT "PLEASE KEY IN A NUMBER:",P
  Q=Q+1
  END

[3]

Degree of difficulty:
   Is this test too difficult? <No 1 2 3 4 5 Yes>

Check out the following questions if you have.
 _ What is the function of INPUT?
 _ P=?
 _ Q+1=?

   Write down other questions that you have: _____

Students' average of degree of difficulty: 2   (simple)
Two errors : (1) No results were made because not PRINT instruction. (2) "P" and "Q" had no relationship, so the program could not run.

04.
   CLS
    INPUT "PLEASE KEY IN 2 NUMBERS:",BIG,SMALL
   PRINT BIG,SMALL,BIG-SMALL
 END
 [3,10]
   Degree of difficulty:
      Is this test too difficult? <No 1 2 3 4 5 Yes>

Check out the following questions if you have.

 _ What are the functions of INPUT and PRINT ?
   _ BIG=? SMALL=?
 _ What are the sequence for INPUT to read data?

   Write down other questions that you have: _____

Students' average of degree of difficulty: 2.67 (between simple
                                              and medium)
Two errors : (1) Some thought variable BIG should have a bigger input number. (2) The output of BIG-SMALL is a string - "BIG-SMALL".

05.
 CLS
    INPUT "PLEASE KEY IN 3 NUMBERS:",C,B,A
   PRINT A,B,C
 END
 [10,15,20]

   Degree of difficulty:
      Is this test too difficult? <No 1 2 3 4 5 Yes>

Check out the following questions if you have.
 _ What are the functions of INPUT and PRINT ?
   _ A=? B=? C=?
 _ What is the sequence for INPUT to read data?

   Write down other questions that you have: _____

Students' average of degree of difficulty: 1.67 (between very simple
and simple)
Two errors : (1) Program read input data by alphabetical order, so A=10, B=15, C=20 (2)
print out were seperated into different lines by ",".

06.
 CLS
   INPUT "PLEASE KEY IN 2 NUMBERS:",EVEN,ODD
   PRINT EVEN,ODD
 END
 [3,8]

   Degree of difficulty:
       Is this test too difficult? <No 1 2 3 4 5 Yes>

   Check out the following questions if you have.
   _ What are the functions of INPUT and PRINT ?
     _ EVEN=? ODD=?
   _ What is the sequence for INPUT to read data?

   Write down other questions that you have: _____

Students' average of degree of difficulty: 1.67 (between very simple
and simple)
One error : Some thought Even=8, ODD=3 by variable's verbal meaning.

07.
 CLS
  X=3
   PRINT "THE VALUE OF X IS 5!"
   PRINT X
 END

   Degree of difficulty:
       Is this test too difficult? <No 1 2 3 4 5 Yes>

   Check out the following questions if you have.
   _ What does the statement "X=3" mean?
   _ What is the function of PRINT ?
   _ Will the print out of the string change the value of X?

   Write down other questions that you have: _____

Students' average of degree of difficulty: 2.5 (between simple
and medium)
Two errors : (1) Print "THE VALUE OF X IS 5!" changed X to 5. (2) This program could not run
because "X=3" conflicted with "X is 5".

08.
  CLS
  A=2
  B=3
  A=B

```
     PRINT "A=";A;",";"B=";B
END
```

Degree of difficulty:
     Is this test too difficult? <No 1 2 3 4 5 Yes>

Check out the following questions if you have.
  _ What are meanings of "A=2", "B=3", and "A=B"?
  _ What are the final values of A, B?
  _ What is the function of PRINT ?
  _ What are the meaning and function of quotation mark?

Write down other questions that you have: _____

Students' average of degree of difficulty: 3 (medium)

Some errors occurred such as : (1) "A=B" was ambiguious because mathematically 2 was not equal to 3. (2) Misunderstanding about memory assignment happened. (3) ";" and "," had no effect on the print out.

```
09.
  CLS
   A=1
   B=2
   C=A+B
   A$="1"
   B$="2"
   C$=A$+B$
   PRINT C
   PRINT C$
  END
```

Degree of difficulty:
     Is this test too difficult? <No 1 2 3 4 5 Yes>

Check out the following questions if you have.
  _ How to represent the value of a string or a number?
  _ What is the meaning of "$"?
  _ What is the meaning of C=A+B ?
  _ What is the meaning of C$=A$+B$?
  _ What is the function of PRINT ?
  _ What are data types while print out a string or a number?

Write down other questions that you have: _____

Students' average of degree of difficulty: 2.5 (between simple
                                         and medium)
Some students thought the answer of C$ was 3. Mentally they misunderstood the meaning of "+" for "plus" number mathematically and for "concatenate" string in computer programming.

```
10.
  CLS
   READ A$,B$,C$
```

```
RESTORE
READ D$
 PRINT A$;B$;C$;D$
   DATA "SPRING","SUMMER","FALL","WINTER"
END
```

Degree of difficulty:
    Is this test too difficult? <No 1 2 3 4 5 Yes>

Check out the following questions if you have.
 _ What does the statement READ ... DATA mean?
 _ What does RESTORE mean?
  _ What do A$, B$, C$, and D$ stand for?
 _ What is the function of PRINT ?
 _ What is the data type for PRINT to output a string?

 Write down other questions that you have: _____

Students' average of degree of difficulty: 2.67 (between simple
                                    and medium)
    Some students thought even without the RESTORE instruction, D$ still got the string -
"SPRING" because the second "READ" was in a different line.

11.
```
CLS
A=5
 B=A+2
 C=A+B
 PRINT C
END
```

Degree of difficulty:
    Is this test too difficult? <No 1 2 3 4 5 Yes>

Check out the following questions if you have.
 _ What is the meaning of "="?
 _ What are values of A, B, C?
  _ What dose the "B=A+2" stand for? What's the value of B?
  _ What dose the "C=A+B" stand for? What's the value of C?

 Write down other questions that you have: _____

Students' average of degree of difficulty: 1.8 (between very simple
                                    and simple)
    Some students didn't know that the assignment of data was from left to right. They might
mistake the flow as from right to left.

12.
```
FOR I=1 TO 3
FOR J=2 TO 3
PRINT J
NEXT J
NEXT I
END
```

Degree of difficulty:
Is this test too difficult? <No 1 2 3 4 5 Yes>

Check out the following questions if you have.
_ What is the function of "FOR..NEXT"?
_ What is the function of Double "FOR..NEXT" loop?
_ What is the function of PRINT ?

Write down other questions that you have: _____

Students' average of degree of difficulty: 2 (simple)
Some students thought the value of J in "PRINT J" had no relationship with the value of J in "NEXT J"; also had no concept about the control function of J in "NEXT J".

Programming exercise :

1. Write a program to read odd numbers from 5 to 91 then output
its average.
Degree of difficulty:
Is this test too difficult? <No 1 2 3 4 5 Yes>

Describe briefly how will you write the program

Draw the flow of your program

Explain source codes of your program

Estimate the result of your program

If you use some of following instructions and have questions on them Please point them out.

_ How to set up the initial value?
_ How to input number from the keyboard?

Looping construct:
_ How to write FOR..NEXT loop instruction?
_ How to write WHILE..WEND loop instruction?
_ How to write DO..LOOP loop instruction?

_ How to use STEP instruction?
_ How to use IF..THEN instruction?
_ How to use GOTO instruction?
_ How to terminate a looping function?
_ What is the meaning of X=X+2?

_ How to use a counter to count?
_ How to write program statements to get the summation?
_ How to write program statements to get the average?
_ How to print out results on the screen?
_ How to use PRINT instruction?
_ How to use PRINT to print number?
_ How to use PRINT to print string?

Write down other questions that you have : _____

Students' average of degree of difficulty: 2.5 (between simple
                                                                and medium)
Some errors or misconeption that students had :
1.   Some did not know how to start it.
2.   Some wrote CLS as first instruction, then stopped.
3.   Some knew looping construct should be used, but couldn't do it.
4.   Some did not know how to sum up the total.
5.   Some confused about the function of WHILE with UNTIL in DO..LOOP;
     for example: DO WHILE X > 91 and DO UNTIL X < 91.
6.   Some did not utilize the repeative function of loop construct,
      still tried to sum up the total from the first number to
      the last number.
7.   Some put the assignment of initial value inside the loop function
     then the variable had been reset every time.
8.   Some even used paper and pencil to calculate the result.

2.   Write a program to read integers repeatly till the total is over 100,
        then print out the average

  Hint : 1. You decide your own methods to read input data
            2. Average means the summation of input integers
              by its number

     Degree of difficulty:
         Is this test too difficult? <No 1 2 3 4 5 Yes>

     Describe briefly how will you write the program

     Draw the flow of your program

     Explain source codes of your program

     Estimate the result of your program

     If you use some of following instructions and have questions on them Please point
them out.

_ How to set up the initial value?
_ How to input number from the keyboard?

Looping construct:
    _ How to write FOR..NEXT loop instruction?
    _ How to write WHILE..WEND loop instruction?
    _ How to write DO..LOOP loop instruction?

_ How to use STEP instruction?
_ How to use IF..THEN instruction?
_ How to use GOTO instruction?
_ How to terminate a looping function?
_ What is the meaning of X=X+2?
_ How to use a counter to count?
_ How to write program statements to get the summation?
_ How to write program statements to get the average?
_ How to print out results on the screen?
_ How to use PRINT instruction?
_ How to use PRINT to print number?
_ How to use PRINT to print string?
_ How to judge or decide whether the total is over 100 or not?
_ How to count the number of input integers?
Write down other questions that you have : _____

Students' average of degree of difficulty: 2.6 (between simple
                                               and medium)
Some errors or misconeption that students had :
1. Some did not know how to start it.
2. Some wrote CLS as first instruction, then stopped.
3. Some knew looping construct should be used, but couldn't do it.
4. Some did not know how to sum up the total. mathematically speaking
   some students do had great difficulty on the meaning of
   expression such as A=A+1 or A=A+B.
5. Some confused about the function of WHILE with UNTIL in DO..LOOP;
   for example: DO WHILE sum > 100 or DO UNTIL sum < 100.
6. Some did not utilize the repeative function of loop construct,
   still tried to sum up the total from the first number to
   the last number.
7. Some put the assignment of initial value inside the loop function
   then the variable had been reset every time.
8. Some even used paper and pencil to calculate the result.
9. Some could not add remarks in the quotations to help to
   understand or explain the INPUT or PRINT function.
10. Some neglected the condition of "sum=100".

ANALYSIS OF COLLECTED ERRORS

The errors that we collected were categorized and analyzed by the following five types.

I. The sequence of program: Line numbers were indispensable for program to execute.

II. The statement of READ..DATA:

1. Some students didn't know the meaning and use of READ..DATA.

2. If put READ on different lines, then it had the function of RESTORE.

3. Some students knew the statement of READ is to read numeric value or strings, but didn't know　　　　they were read from the statement of DATA.

4. Some students didn't know how to use RESTORE.

III. INPUT/OUTPUT Statements:

1. The statement of INPUT:

(1) Didn't know the input format of numeric and string variables.

(2) Didn't know the meaning and function of ";" and ","

(3) Some mistook the value of numeric variable by variable name.

(4)　　　Read the value of variables according to alphbatic orders.

(5)　　　Program would go wrong when verbal meaning of variables name and its value were not　　　　　matched

(6)　　　A and B in INPUT A,B were input variables, and it's not necessary to input from keyboard again.

(7)　　　After execution, the "?" would not appear on the screen

(8) Programs couldn't run, if input variable inconsistent with old variable.

(9)　　　Didn't know how to use strings as remark or hint.

2. The statement of PRINT:

(1)　　　The string would be output with quotation marks(" ") and even print out the "PRINT"　　　　　　statement.

(2)　　　Some students were confused with the functions of ";" and ",".

(3)　　　";" and "," have no meaning or function, and they would appear again in output.

(4)　　　Didn't know the meanings and functions of ";", ",", and quotation mark.

(5)　　　Quotation mark had no meaning for "A+B" or "A-B", these strings could still do mathematical operation.

(6) Variables' value could be changed even inside a string.

(7)　　　Some thought statements inside the quotation mark were still executable.

(8)　　　Numeric variable couldn't be calculated directly from PRINT. For example, the output　　　　　of PRINT X-Y was string "X-Y".

(9)　　　"PRINT C" wouldn't output the value of C, and only print out strings and quotation　　　　　　marks.

(10)　　A=5

PRINT A

The output would be: A 5 or A=5 , that was, output included the name of variable. String variable would be operated in the same way.

(11)　　Some didn't know the instructions inside the quotation mark were only a string.

IV. Assign statement:

1. A=B:

(1)　　The meaning of A=B and B=A were the same.

(2)　　Some students thought that " A=2; B=3; A=B" was illogic.

(3)　　"=" here was meant equal on two side of the equal sign.

(4)　　It just meant A=B.

2. C=A+B, C$=A$+B$:

(1)　　Some were confused with or even didn't know how to define string and numberic variables.

(2)　　Didn't know the meaning of "$" in A$.

(3)　　The meaning of "+" was the summation of number in the quotaion marks. For example, if   A$="1", B$="2" then C$="3".

(4) Didn't know how to do it; If A$="GOOD" B$="MORNING" then C$=A$+B$

(5)　　Didn't know if there was any space between A$ and B$ in C$=A$+B$.

3. Concept of memory address for numeric variables:

(1) Didn't know the meaning and function of A=A+1 and A=A+B were to count numbers and   get the total.

(2)　　Some thought Q+1=10 and then Q's value was 9 mathematically.

(3)　　Didn't know to use the repeating function of looping to count numbers, instead calculated   with pencil or brain.

(4)　　Couldn't calculate arithmatic average.

(5) Some used natural language to write the program, such as " T = THE TOTAL OF X", then   would get the total.

4. Setting the initial value:

(1)　　Couldn't do the setting, or had no concept on it.

(2) Didn't know where to set.

(3)　　Set the wrong number, so got the wrong result.

(4)　　Put it in the loop construct, and kept to reset the variable.

(5)　　The starting value of any variable was 1.

(6)　　N=1 and N=N+1 had to write out at the same time.

V. The looping construct:

1. The misconception of FOR..NEXT instruction:

(1)　　omit NEXT

(2)　　Didn't know the operating mechanism (sequence).

(3)　　If put the PRINT J inside the loop construct, then one space line was added.

(4)   Thought that the number of looping would get same number of output data.

(5)   FOR I=1 TO 3

              FOR J=2 TO 3

              PRINT J

              NEXT J

              NEXT I

              In the loop construct above "I" and "J" hadn't mutually operate, so control variable "I" had no meaning or function here.

(6)   Some didn't know the increment function of "STEP".

2. DO .... LOOP:

(1)   In the construct of DO ... LOOP, the function of "WHILE" and "UNTIL" were confused with each other.

(2)   Some didn't know the sphere of loop function.

(3)   Didn't know where to put the repeat or judge statement?

      The followings are the frequency of each type of error. Of them, the calculation of "Total statements" is according to the frequency of instructions or structures occurred in each student' test or program. The calculation of "Number of errors" is the frequency of errors. Because subjects might have more than one question or error about same instruction, the sum of "Number of errors" could be more than the sum of "Total statements".

	Total statements	no. of errors	percentage
I. The sequence of program: Line numbers were indispensable for program to execute.	16	3	18.75%
II. The statement of READ..DATA	8	6	75.00%
1. didn't know the meaning and use of READ..DATA		1	12.50%
2. If put READ on different lines, then it had the function of RESTORE.		1	12.50%
3. didn't know numeric value or strings were read from DATA statement		1	12.50%
4. didn't know how to use RESTORE		3	37.50%
III. INPUT/OUTPUT Statements			
1. The statement of INPUT:	20	16	80.00%
(1) Didn't know the input format of numeric and string variables		2	10.00%
(2) didn't know the meaning and function		2	10.00%

		of ";" and ","		
	(3)	mistook the value of numeric	4	20.00%
		variable by variable name		
	(4)	read the value of variables according	1	5.00%
		to alphabetic orders		
	(5)	Program went wrong when variables and	1	5.00%
		values were not matched		
	(6)	A and B of INPUT A,B are input	1	5.00%
		variables		
	(7)	After execution, hint strings and	2	10.00%
		input variables would not appear on the screen.		
	(8)	Program couldn't run, if input	1	5.00%
		variable inconsistent with old variable.		
	(9)	didn't know how to use hint strings	2	10.00%

2. The statement of PRINT: 56   2   44.64%

	(1)	The string would be output with quotation marks(" ")	4	7.14%
		and even print out the "PRINT" statement.		
	(2)	Some students were confused with functions	2	3.57%
		of ";" and ",".		
	(3)	";" and "," have no meaning or function,	7	12.50%
		they would appear again while output results.		
	(4)	Didn't know meaning and function of	1	1.79%
		";", ",", and quotation mark		
	(5)	Quotation mark had no meaning for "A+B" or "A-B",	1	1.79%
		these strings could still do mathematical operation.		
	(6)	Variables' value could be changed even	1	1.79%
		inside a string.		
	(7)	Some thought statements inside the quotation	1	1.79%
		mark were still executable.		
	(8)	Numeric variable couldn't be calculated	1	1.79%
		directly from PRINT statement. For example, the output of		
		PRINT X-Y was string "X-Y"		
	(9)	"PRINT C" wouldn't output the value of C,	1	1.79%
		only print out strings and quotation marks.		
	(10)	A=5	5	8.92%

PRINT A

The output would be: A 5 or A=5 , that is, output includes
the name of variable. String variable would be operated
in the same way.

(11)	Some didn't know the instructions inside the quotation mark were only a string.		1	1.79%

IV. Assign statement :

1.	A=B:	4		61	50.00%
	(1)	The meaning of A=B and B=A were the same.		1	25.00%
	(2)	Some students thought that " A=2; B=3; A=B" was illogic.		2	50.00%
	(3)	"=" here was meant equal on two side of the equal sign.		2	50.00%
	(4)	It just meant A=B.		1	25.00%
2.	C=A+B, C$=A$+B$:		8	8	100.00%
	(1)	Some were confused with or even didn't know how to define string and numberic variables.		1	12.50%
	(2)	Didn't know the meaning of "$" in A$.		3	37.50%
	(3)	The meaning of "+" was the summation of number in the quotation marks. For example, if A$="1", B$="2" then C$="3".		2	25.00%
	(4)	Didn't know how to do it; If A$="GOOD" B$="MORNING" then C$=A$+B$		1	12.50%
	(5)	Didn't know if there was any space between A$ and B$ in C$=A$+B$.		1	12.50%
3.	Concept of memory address for numeric variables:		16	16	100.00%
	(1)	Didn't know the meaning and function of A=A+1 and A=A+B were to count numbers and get the total.		9	56.25%
	(2)	Some thought Q+1=10 and then Q's value was 9 mathematically.		1	6.25%
	(3)	Didn't know to use the repeating function of looping to count numbers, instead calculated with pencil or brain.		4	25.00%
	(4)	Couldn't calculate arithmatic average.		1	6.25%
	(5)	Some used natural language to write the program, such as " T = THE TOTAL OF X", then would get the total.		1	6.25%
4.	Setting the initial value:		16	9	56.25%
	(1)	Couldn't do the setting, or had no concept on it.		2	12.50%
	(2)	Didn't know where to set.		2	12.50%
	(3)	Set the wrong number, so got the wrong result.		2	12.50%

(4)	Put it in the loop construct, and kept to reset the variable.		1	6.25%
(5)	The starting value of any variable was 1.		1	6.25%
(6)	N=1 and N=N+1 had to write out at the same time.		1	6.25%

V. The looping construct:

1.	The misconception of FOR..NEXT instruction:	9	9	100.00%
(1)	omit NEXT		1	11.11%
(2)	Didn't know the operating mechanism (sequence).		2	22.22%
(3)	If put the PRINT J inside the loop construct, then one space line was added.		3	33.33%
(4)	Thought that the number of looping would get same number of output data.		1	11.11%
(5)	FOR I=1 TO 3		1	11.11%

FOR J=2 TO 3

PRINT J

NEXT J

NEXT I

In the loop construct above "I" and "J" hadn't mutually operate,

so control variable "I" had no meaning or function here.

(6)	Some didn't know the increment function of "STEP".		1	11.11%
2.	DO .... LOOP: 6		6	100.00%
(1)	In the construct of DO ... LOOP, the function of "WHILE" and "UNTIL" were confused with each other.		1	16.67%
(2)	Some didn't know the sphere of loop function.		2	33.33%
(3)	Didn't know where to put the repeat or judge statement?		3	45.00%

While planning the research direction for the second year, results from the first year showed that many students had difficulty in doing the looping programming exercises. From the observation and interview some problems such as: (1) What is the function of looping program? (2) How to arrange the control variables? (3) How to count the number of looping? (4) When to stop the loop? and (5) What is the control flow of looping function? often caused frustration and errors when students did their programs. Based on these findings, the second year of this research systematically collected and analyzed the error patterns and misconceptions about looping in programming exercise. The goal-planning process (Miller, 1982; Soloway, et al, 1985) and the problem-solving and general reasoning procedures of the information processing model in cognitive psychology (Gagne, 1985) will be used to design an intelligent tutoring system from these knowledge bases for error patterns and misconception analysis later. Samples were invited from freshmen and elementary school teachers who took "Information Science" course (100 persons totally).

MISCONCEPTION OF LOOP PROGRAMS
Type I: Programs to calculate the summation from 1 to N
    Ex 1 :

```
        10 S=0
        20 K=1
        30 K=K+1
        40 S=S+K
        50 IF K=100 THEN 70
        60 GOTO 40    ------->  Change to  "GOTO 30", otherwise
        70 PRINT K              the function of K=K+1 is missed.
        80 END
Ex 2 :

        10 S=0
        20 K=1
        30K=K+1
        40 S=S+K
        50 IF K=100 THEN 70
        60 GOTO 30   ------->  After changing to "GOTO 30" and
        70 PRINT K             rerunning the program, the answer
        80 END                 "5049" is still wrong.  The k=1
                               in line 20 should be replaced by
                               k=0, if not, the k=k+1 in line 30
                               will start the value of k from 2
                               and make the result of the summation
                               less 1 than it should be.
Ex 3 :
        10 INPUT N        ----> Some students neglect this
        20 S=0                  instruction.
        30 K=1
        40 S=S+K
        50 PRINT S
        60 K=K+1 GOTO 40   ----> Cause infinite loop, should
        70 IF K=N+1 THEN 80      swap with line 70.
        80 END
Ex 4 :
        10 S=0
        20 INPUT N
        30 FOR K=1 TO N
        40 IF K=N THEN 80  ----> Some students forget that the
        50 S=S+K                 "FOR..NEXT" can make decision
        60 NEXT K                by itself.
        70 PRINT "S"       ----> Output the string "S" in stead of
        80 END                   the numeric value of the summation.
Ex 5 :
        10 FOR I=1 TO N
        20 S=0          ----> The summation has repeatly
        30 S=S+I              got the value of zero.
        40 NEXT I
Ex 6 :
        10 FOR A=0 TO N-1
        20 S=0         ----> S is repeatly got zero.
        30 A=A+1       ----> Ignore the "FOR..NEXT" loop
        40 S=S+A             has a counter embedded in it.
        50 NEXT A
        60 PRINT S
        70 END
```

Type II: Programs to print out the "9 * 9" multiplication table

Ex 1 :

```
10 FOR I=1 TO 9
20 FOR J=1 TO 9
30 PRINT I "*" J "=" I*J ----> Many students were confused
40 NEXT J                        with the function of ";"
50 NEXT I                        and of ",".
60 END
```

Ex 2 :

```
10 FOR I=1 TO 9
20 FOR J=1 TO 9
30 PRINT I*J=K          ----> I*J=K is mathematically
40 NEXT J                     right but it is a wrong
50 NEXT I                     concept in computer program.
60 END                        The concept of variable and
                              its memory allocation is
                              misunderstood.
                              No quotations had shown on
                              the program.
```

EX 3 :

```
10 FOR I=1 TO 9
20 FOR J=1 TO 9
30 NEXT J
40 NEXT I
50 PRINT I:"*";J;"=";I*J --> Line 50 should be put
60 END                        into the double
                              "FOR..NEXT" loop, otherwise
                              the wrong output of
                              "10*10=100" will be printed out.
```

Ex 4 :

```
10 FOR I=1 TO 9
20 FOR J=1 TO 9
30 PRINT I;"*";J; "=";"I*J" ---> incorrect quotation marks
40 NEXT J
50 NEXT I
60 END
```

Ex 5 :

```
10 FOR I=1 TO 9
20 FOR J=1 TO 9
30 PRINT "I*J";=I*J -----------> incorrect quotation marks
40 NEXT J
50 NEXT I
60 END
```

Ex 6 :

```
10 FOR I=1 TO 9
20 FOR J=1 TO 9
30 PRINT "I";*;"J"="K" ----> "K" should be replaced
40 NEXT J                         by I*J
50 NEXT I
60 END
```

Ex 7 :

```
10 FOR I=1 TO 9 ; J=1 TO 9 --> Using the ";" and put
20 PRINT I"*"J"="I*J            the two "TO" structure
40 NEXT J                       on the same line is similar
```

```
        50 NEXT I                              to natural language.
        60 END
Ex 8 :

        10 FOR I=1 TO 9
        20 FOR J=1 TO 9
        30  PRINT  I;"*";J"=";K
        40 IF J=9 THEN GOTO 70   ----> "IF..THEN" instructions have
        50 IF I=9 THEN GOTO 80        been redundantly used,
        60 NEXT J                         neglecting that the "FOR..NEXT"
        70 NEXT I                        loop can stop the program.
        80 END
```

Type III: Programs to display various "*" outputs

Ex 1 :

```
        FOR I=1 TO 9
        READ N
        NEXT I
        FOR J=1 TO N
        PRINT  "*"
        NEXT J
        DATA 5,4,3,2,1,2,3,4,5
                        RUN:
                              *****
```

incorrect concept about memory

Ex 2 :   RUN:

```
        FOR I=1 TO 9                 * * * * *
        READ N                       * * * *
        FOR J=1 TO N                   * * *
        PRINT "*"                        **
        NEXT J                            *
        NEXT I                          * *
        DATA 5,4,3,2,1,2,3,4,5         * * *
                                      * * * *
                                      * * * * *
```

Ex 3 :

```
        FOR I=1 TO 3
        FOR J=1 TO 3
        PRINT  "*";
        NEXT J
        PRINT
        NEXT I
                    RUN:
                                   ***
                                   ***
                                  * * *
```

";" wrongly used

Ex 4 :

```
        FOR I=1 TO 3
        FOR J=1 TO 3             RUN:
        PRINT  "*"                       *
        NEXT J                           *
        PRINT                            *
        NEXT I                           *
                                         *
```

```
*
*
*
*
```

Control variables were incorrectly used all the time. For example:

1. matching mistake
   ```
   FOR I=1 TO 9
   FOR J=1 TO 9
   PRINT  I;"*";J;"=";I*J
   NEXT I
   NEXT J
   ```
2. Control variables I after the "NEXT I" can be omitted.
3. can use real number
4. The function of "STEP" is misunderstood.

The collected misconceptions will be constructed as the based knowledge for the student model and planning in an ICAI programming tutoring system. The ideal of individualized instruction is a long term goal of any ICAI system to reach and this research is the initial step toward the goal and much effort is needed.

REFERENCES

Anderson, J. R. (1982). Acquisition of cognitive skill. Psychological Review, 89, 369-406.

Anderson, J. R. & Reiser, B. J. (1985). The LISP tutor: it approaches the effectiveness of the human   tutor. Byte, 10(4), 159-175.

Bitter, G. G. (1984). Computer in today's world. New York: John Wiley & Sons.

Gagne, R. M. , & Dick, W. (1983).   Instructional Psychology. Annual Review of Psychology, 34, 261-259.

Gagne, E. D. (1985).   The cognitive psychology of school learning. Boston, MA: Little, Brown and   Company.

Mayer, R. E. (1985).   Learning in complex domains : A cognitive analysis of computer programming. The   Psychology of Learning and Motivation, 19, 89-130.

Miller, M. L. (1982). A structured planning and debugging environment for elementary programming. In   D, Sleeman and J. S. Brown (Ed.), Intelligent tutoring systems. New York : Academic Press.

Murdock, B. B., Jr. (1961).   The retention of individual items. Journal of Experimental Psychology, 62,   618-25.

Sleeman, D. & Brown, J. S. (Eds). (1982). Intelligent tutoring systems. New York : Academic Press.

Soloway, E. (1986).   Learning to program = learning to construct mechanisms and explanation.   Communications of the ACM, 29(9), 850-858.

Soloway, E. et al., (1981).   Cognition and programming why your students write those crazy program.   National Educational Computing Conference, 206-219.

Stevents, A., Collins, A., & Goldin, S. E. (1982). Misconceptions in students' understanding. In D.   Sleeman and J. S. Brown Intelligent tutoring system. New York: Academic press.

Williams, M. D. & Hollan, J. D. (1981). The process of retrieval from very long-term memory. Cognitive   Science, 5, 87-119.

Wood P. H. & Holt P. P. (1990).   Bibiliographies, SIGART Bulletin. 1(1), 21-42.