SAINT ELIZABETH HIGH SCHOOL

1530 THIRTY-FOURTH AVENUE
OAKLAND, CALIFORNIA 94601

OFFICE OF THE PRINCIPAL

March 9, 1971

TO WHOM IT MAY CONCERN:

This letter is to verify that the Teacher Evaluation Program enclosed was designed and written by Ralph Wondra. He worked completely on his own, and ran the program after the Evaluation was given to the students. The program worked exactly as it should have, and was of inestimable help in running our evaluation.

Sincerely,

John G Appel Jr

John G. Appel, Jr.
Principal

JGA:j

Title: TEACHER EVALUATION

Category: BUSINESS

Contestant: RALPH WONDRA
4449 ANDERSON AVENUE
OAKLAND, CALIFORNIA

Sponsor: MR. JOHN APPEL

School: SAINT ELIZABETH HIGH SCHOOL
1530 34th AVENUE
OAKLAND, CALIFORNIA

MARCH 8, 1971

Title:	TEACHER EVALUATION
Category:	BUSINESS
Contestant:	RALPH WONDRA
Computer used:	HEWLETT-PACKARD 9100B
Computer language:	HEWLETT-PACKARD LANGUAGE
Configuration:	H-P 9100B CALCULATOR-COMPUTER
	H-P PRINTER
	H-P CARD READER

Table of Contents

Abstract

The Teacher Evaluation is a program designed and written by Ralph Wondra, for tallying the results of a survey given by Saint Elizabeth High School. The survey compiled by students and faculty, numbered 48 questions, most of which having 5 possible responses. Each student evaluated his six teachers on how well they did their jobs.

The purpose of this program is to tally the number of students that chose each of the responses for the 48 questions in the survey. After all of the cards for a certain class were run through, then the computer would calculate and print out all the percentages in a neat way.

Limitations and restrictions to this program are present only because of the small size of the Hewlett-Packard 9100-B. Only the cards of one class of less than 100 students may be run through at a time. A computer card can only hold up to 24 responses to questions, but this program can only tally 12 at a time. This makes it necessary to run each card through twice. The only other restriction is to assume that the cards are filled out correctly.

STATEMENT OF THE PROBLEM

In order to understand the difficulty of this problem, some idea of the relatively small size of this computer, must be indicated. The Hewlett-Packard 9100B contains 32 storage registers, each capable of containing 14 instructions or 12 numeral digits and 2 exponents. This makes it necessary for the program, including its storage, not to exceed 448 instructions or digits. There are three other registers, the X-reg, the Y-reg, the Zreg, but these are the only ones used for calculating. To multiply the X & Y registers it takes one instruction, to store or recall a register it takes two instructions,'if statements' or computed 'go to' statements take three instructions. There are many other steps needed to maneuver the numbers to the proper calculating registers.

A very challenging section of this program was the excessive craming of the instructions, so that enough room for storing the running totals could be saved. Plenty of room is needed for storing the responses of just 12 survey questions. Each survey question has 5 possible responses creating 60 separate and distinct numbers. Adding the votes of each student takes a lot of sorting in this computer.

Program Description

The Teacher Evaluation consists of 229 instructions. It uses 15 registers for storage of numbers, or 210 characters, coming to a grand total of 439 locations in the computer or just 9 fewer than the capacity of the 9100B. Operational time of an average class of 35 is 6 minutes including print-out. There were 41 teachers, most having five classes, evaluated with this program. Approximately 10,000 cards were run through this program, two per student per class. Each one of these cards had to be run through twice, because the program is limited to tally only 12 questions at a time.

Many problems appeared while writing this program. It is best to expain most of these promblem and how they were solved before giving a total description of the program. The major problem occuring in this program was that having enough room for instructions and the data to be stored. One technique used to overcome this, was to pack a register. Why store only a three digit number in an entire register, when four three digit numbers can be stored there? To tally 12 questions in the survey, as this program did, 60 different numbers had to be stored. Since every class had less than a hundred students, these 60 numbers would each be two digits. Therefore only these 60 can be easily packed into 12 registers. Only five numbers are packed into a register, because if the votes from the first question are packed into the first register this will greatly reduce confusion.

To pack five numbers into a register, the decimal point is placed in front of the first number, the second number trails the first , the third trails the second, et cetera. These numbers are all kept at two digits each, with zeroes in between if need be. Five numbers such as 14,34,89, 54, and 67 would be packed such as 0.1434895467. Placing the decimal point before the first number saves steps when recalling for print-out. With out using this technique it would have been impossible to write this program.

Another problem encountered was to find an easy way of create the additioning number, or the number added to the register to mark a vote for that response. For example, if the number indicated on the card was a 1 then 0.01 would have been the additioning number. IF the number was 4 the additioner would have been 0.00000001 or 10^{-8}. A way of finding these numbers would have been to ask a series of if statements. this way would have taken about 50 steps, while the method used consisted of only 20. The method finally used was a looping technique, in which the current value of the additioner was increased by raising its power of ten by two while reducing the response number by one until this equaled zero. At this time the positive exponent was change to a negitive exponent making the additioner ready. This method is show in the program listing to be from -2.b through -4.2.

An extremely difficult problem encountered was that of adding the additioner of a question to the storage register that corresponds to that question. This might have been accomplished by calling the section of the program that deals with seperating the response number from the rest of the response numbers and creating the additioner for that question, a subroutine. This would have been a good way except that it consumed too much space. To go to a subroutine takes four steps, then after returning, the additioning number would have to be added to that register, taking another five steps. For 12 questions this procedure itself would have taken 108 steps. The subroutine itself would have taken more than 50 steps, and the print-out section of the program would still have to fit in somewhere. The print-out itself would have to recall each register, compute the percentage, and print the votes out. This solution to the problem would definitely not fit in this computer.

If the additioner could be added to only one number, the procedure would have been a lot simplier. This was the basis of the method used. TO accomplish this consider the 12 registers to be in a verticle column from A-L. The additioner from the first response would be added to A. Then the column of numbers would roll up, that is the number that was in B is now in A, the one that was in C is now in B, and A is now at the bottom in L. Now the additioner from the second response is added to the number in A which really is the one that was in B. If the column of numbers is rolled up 12 times all together, then its back at the original place. This procedure for rolling up can be used as a subroutine so that it may be used for the print-out section. By using a special instruction called Y interchange ( ) , the roll up method can be easily done. This method used only 29 instructions. This section is found in the program listing to be from -6.2 through -8.3. In this program the verticle column is A, D, C, B, 4, 5, 6, 7, 8, 9, -F, & -E, from top to bottom.

To understand the Teacher Evaluation program, it is necessary to know what happens when a card filled out by a student is run through the computer. Referring to the sample card pictured in appendix B on page 19 it is noted that 24 questions can be answered. Ten different responses can be filled in for each question. If box number 1 is filled in, then a 1 is entered into the machine. If boxes 1 & 4 are filled in then a 5 is entered. The codes from the first set of 12 numbers form a 12 digit packed number, with the decimal point placed before the first digit. This 12 digit number is first in the X reg, where all numbers are entered, then it is arrowed up to the Y reg. by the next instruction. Then the second 12 digit number is entered into the X reg. The next instruction previously filled in signifys the computer to continue.

Note, before any cards are to be run through, the 12 store registers are to be cleared, or set to zero. If they are not cleared, the numbers previously stored in those registers will remain. Also note that before starting it is necessary to indicate which set of 12 digits is to be tallied, either the first or the second. In this program it was accomplished by the use of a special instruction, called the flag. If the second 12 was to be tallied, the flag would be set. In the program after it had cleared the 12 registers, an if flag statement would be asked. Now if the flag had been set the computer would go to the location indicated by the next two instructions, which in this case is -5.8. If the flag was not set the computer would skip the two instruction previously mentioned then continue with the procedure. If the flag was set, the computer would store only the second 12 digits, those in the X-register. The first 12 digits would be stored only if the flag was not set.

Either if the flag was set or not, the computer would stop so that the first card could be run through. The correct set of 12 numbers would be stored. This concept of the flag might be clearer if the flow chart is studied. At this time the computer would add 10^{-2} to the F- reg. This is the running number of students needed to compute percentages. One thing to note, this F-register is cleared of its intiger value before each card is run through, while all of the F-reg. is cleared out before each set of cards is run through.

One is now added to the F-reg, to indicate how many times the loop has been run through. So the first time it goes through the loop F is at 1. The first digit of the 12 digit number is separated, by multiplying by 10 then taking the intiger of this. To have the second digit where the first was, the intiger value is subtracted from that of ten times the original. This new number recieved from the subtraction is now stored where the old one was.

```
For Example:          original # = 0.123456789012
                 10 x original # = 1.23456789012
            int. 10 x original # = 1.0
                           new # = 0.23456789012
```

'If statements' make sure that this interger value is not equal to zero, or greater than five. Then the additioning number is found as explained on page four. The additioner is added to the first of the 12 registers, which in this program is A. This first response is now tallied and the additioner or the interger values can be discarded. The program continues by going to the rollup subroutine, as explained on pages five and six. When it returns from the subroutine, it is asked if 12 is greater than F. This first time through it is not, because F = 1.__ . It then returns to were one is added to F. The program continues going through the loop until it goes through the entire loop 12 times, which is when F is greater than 12. Because the loop was run through 12 times, the 12 storage registers are at their original places, as was explained on pages five and six. The program now goes to the place where the intiger value of F was cleared, then it stops so that the next card can be fed through.

The printing out of the results also involves the use of the flag. After all of the cards are fed through, then the flag is set and an interger number is placed in the X register to signify the number of the question that is to be printed out. For example: if the numbers are to be 1 through 12 then a 1 is placed in the X register. If the numbers are to be 25-36 then a 3 is entered. The computer only knows that 60 votes are store in 12 registers, it doesn't know how to start numbering the print out. After these two instructions are complete, then continue is pressed manually. This restarts the program.

The number in the X register is multiplied by 12 then subtracted by 12. This new number is one less than the the number to start printing out. It is added to F, and this total of the starting number, an interger value, and the number of students is then stored in F. The response counting number E is set to zero. Now 1 is added to F, to give the correct print out number. The interger value of F is printed out and spaced. The register that stored the responses for the first question, A, is recalled to the Y register so that the five tallied responses can be separated. One is added to E to indicate what response it is calculating now. Y is multiplied by 100, the intiger value is taken and subtracted with the result becoming Y. The number of students is separated from the F register so that together with the interger value the percentage can be computed. The decimal place of the percentage is brought before the 100 in 100% so that it would read .100. This is done so that the percentage and the number of the response can be added for simplicity in print out. For example: the response number is 3; and the percentage is 45; then it would be printed out as 3.045. A sample set of print out would be:

```
9.

1.033
2.025
3.017
4.012
5.012
```

Other sample print outs may be found in appendix C on page 20. The 9 in the sample above is the question number. Note the percentages as indicated above do not have to add up to 100%. A student might have left that question blank or filled in the wrong boxes, or the difference might be from the computer's rounding off 33.3% to 33.

The print out loop must have some counter so that after printing the 5 response percentages for a question, it will go on to print the next question. This is accomplished by asking if the response number and the percentage is greater than 4.9. If this is not true than the response percentage number would print out than go back into the loop until the fifth percentage number was to be printed out, at which time 4.9 would be greater. Then the computer would print out this fifth number, spaces it by two and goes to the roll up subroutine. The response number would be set to zero by storing it in E. It would now see if it had gone the the outer loop 12 times by dividing the number of the question last printed out by 12, then asking if the interger value of this is equal. If this was equal it would end, so that the next set of cards could be run through. But if this was not equal, it would go back to the beginning of the outer loop where 1 is added to the response number and the question print out number. The numbers stored in the 12 register remain after print out so that the same list of numbers may be printed out more than once. The Teacher Evaluation as used for the survey at Saint Elizabeth tallied only 5 responses for each question, but by changing two instructions, -2.3 to code # 6, and -b.2 to code # 5, this program will tally and print out the responses for 6.

Input and Output

There are no input formats for this program. This is taken care of on the answer cards previously. The decimal place is one of the instructions on the card filled in by the manufacturer. All the data entered by the cards would be a 12 place digit with the decimal point in front of the first digit, such as 0.135214251153. The only other numbers entered are for the print out, which would have been an interger value entered manually, by just pressing the appropriate key.

Output formats are just the same. There are only two types of numbers, one is just the interger value to indicate the question number, this number may be as large as 12 significant digits. The other number is the packed response and percentage number, as explained on page 10. It is between 1.0 and 5.1. The percentage could have as many places as wanted within the limits of the machine. The program for use in this survey needed only an interger value for the percentage, so the decimal wheel was placed at 3. Sample input and output numbers are in appendices B and C.

Operating Instructions

1.) Enter the program at the locations indicated in the listing.

2.) End. This places the computer at +0.0.

3.)

a.) If the first 12 digits on every card are to be tallied press continue.

b.) If the second 12 digits on every card are to be tallied press set flag then continue.

4.) Run cards through the computer.

5.) When print out is desired press set flag.

6.) Turn printer on for the X register.

7.) Set decimal wheel at 3.

8.) Enter an interger in the X register to indicate how the numbers are to be numbered. For example: if 1-12 are to be printed place a 1 in the X register; if 37-48 are to be printed enter a 4. Then press continue. This interger indicates which set of 12 is to be printed out, the first, the second the third, et cetera.

Appendix A

Program Listing

location	code #	instruction
+0.0	37	clear X
+0.1	23	X into
+0.2	13	reg A
+0.3	23	X into
+0.4	14	reg B
+0.5	23	X into
+0.6	16	reg C
+0.7	23	X into
+0.8	17	reg D
+0.9	23	X into
+0.a	34	minus
+0.b	12	reg E
+0.c	23	X into
+0.d	34	minus
+1.0	15	reg F
+1.1	23	X into
+1.2	11	reg 9
+1.3	23	X into
+1.4	10	reg 8
+1.5	23	X into
+1.6	07	reg 7
+1.7	23	X into
+1.8	06	reg 6
+1.9	23	X into
+1,a	05	reg 5
+1.b	23	X into
+1.c	04	reg 4
+1.d	61	recall
+2.0	63	acc-
+2.1	44	go to
+2.2	34	minus
+2.3	00	zero
+2.4	00	zero
-0.0	61	recall
-0.1	64	int X
-0.2	63	acc-
-0.3	43	if flag
-0.4	05	{ 5
-0.5	10	8
-0.6	41	stop
-0.7	43	if flag
-0.8	10	{ 8
-0.9	03	3
-0.a	30	X intchng Y
-0.b	30	X intchng Y
-0.c	26	ent exp.
-0.d	02	2
-1.0	32	ch. sign

THIS FIRST SECTION CLEARS THE 12 STORAGE REGISTERS AS EXPLAINED ON PAGE 6.

THIS SECTION DEALS WITH WHETER THE FIRST 12 OR THE SECOND 12 WILL BE TALLIED, AS EXPLAINED ON 7.

Appendix A

Program Listing Cont'd.

location	code #	instruction
-1.1	60	acc+
-1.2	37	clear X
-1.3	30	X intchng Y
-1.4	01	1
-1.5	60	acc+
-1.6	61	recall
-1.7	01	1
-1.8	00	0
-1.9	36	mult
-1.a	25	arrow down
-1.b	27	arrow up
-1.c	64	int X
-1.d	34	subtract
-2.0	40	Y into
-2.1	12	reg E
-2.2	30	X intchng Y
-2.3	05	5
-2.4	52	if X lt. Y
-2.5	04	4
-2.6	07	7
-2.7	00	0
-2.8	50	if X = Y
-2.9	04	4
-2.a	07	7
-2.b	26	ent exp.
-2.c	02	2
-2.d	30	X intchng Y
-3.0	22	roll up
-3.1	01	1
-3.2	50	if X = Y
-3.3	04	4
-3.4	01	1
-3.5	34	subtract
-3.6	26	ent exp.
-3.7	02	2
-3.8	22	roll up
-3.9	36	mult.
-3.a	30	X intchng Y
-3.b	31	roll down
-3.c	44	go to
-3.d	03	´3
-4.0	01	1
-4.1	22	roll up
-4.2	35	divide
-4.3	13	recall A
-4.4	33	add
-4.5	40	Y into
-4.6	13	reg A

THIS STEP PLUS THE LAST 3 ON PAGE 14 KEEP A TOTAL # OF STUDENTS, AS EXPAINED ON 7.

THIS SECTION KEEPS A RUNNING TOTAL OF HOW MANY TIMES IT HAS GONE THROUGH THE LOOP, AS EXPLAINED ON 7 & 8.

THIS SECTIONS SEPERATES THE FIRST DIGIT, WHICH IS THE RESPONSE # MAKING SURE IT IS GREATER THAN ZERO BUT LESS SIX, AS EXPLAINED ON PAGE 8.

THIS IS THE ADDITIONER SECTION, HERE THE RESPONSE # IS CHANGED INTO A POWER OF TEN TO BE ADDED TO THE REGISTER AS EXPLAINED ON PAGE 4.

THIS SECTION ADDS THE ADDITIONER TO THE A REGISTER WHICH HOUSES THE TOTAL OF RESPONSES FOR THAT QUESTION, AS EXPLAINED ON PAGE 8.

Appendix A

Program Listing Cont'd

location	code #	instruction
-4.7	44	go to
-4.8	77	subroutine
-4.9	06	6
-4.a	02	2
-4.b	15	recall F
-4.c	64	int X
-4.d	22	roll up
-5.0	01	1
-5.1	02	2
-5.2	50	if X = Y
-5.3	00	0
-5.4	00	0
-5.5	44	go to
-5.6	01	1
-5.7	02	2
-5.8	41	stop
-5.9	43	if flag
-5.a	10	8
-5.b	03	3
-5.c	54	set flag
-5.d	44	go to
-6.0	00	0
-6.1	14	B
-6.2	24	Y intchng
-6.3	13	reg A
-6.4	24	Y intchng
-6.5	34	minus
-6.6	12	reg E
-6.7	24	Y intchng
-6.8	34	minus
-6.9	15	reg F
-6.a	24	Y intchng
-6.b	11	reg 9
-6.c	24	Y intchng
-6.d	10	reg 8
-7.0	24	Y intchng
-7.1	07	reg 7
-7.2	24	Y intchng
-7.3	06	reg 6
-7.4	24	Y intchng
-7.5	05	reg 5
-7.6	24	Y intchng
-7.7	04	reg 4
-7.8	24	Y intchng
-7.9	14	reg B
-7.a	24	Y inchng
-7.b	16	reg C
-7.c	24	Y intchng
-7.d	17	reg D
-8.0	24	Y intchng
-8.1	13	reg A
-8.2	77	RETURN SUB.

HERE IT GOES TO THE ROLL-UP SUBROUTINE.

THIS SECTION MAKES SURE IF IT WENT THROUGH THE LOOP 12 TIMES AS EXPLAINED ON PAGE 8.

THIS IS PART OF THE SECTION WHICH TALLIES ONLY THE SECOND 12 DIGITS ON THE CARD, AS EXPLAINED ON PAGE 7.

THIS IS THE ROLL-UP SUBROUTINE, WHICH ROLLS A COLUMN OF NUMBERS UP, SO THAT THE ADDITIONER CAN BE ADDED TO ONLY ONE REGISTER.

THE COLUMN IS:

1	A
2	D
3	C
4	B
5	4
6	5
7	6
8	7
9	8
10	9
11	-E
12	-F

THIS IS EXPLAINED ON PAGES 5 & 6.

Appendix A

Program Listing Cont'd

```
location code # instruction
-8.3     22      roll up
-8.4     01        1
-8.5     02        2
-8.6     36      mult
-8.7     34      subtract
-8.8     37      clear X
-8.9     30      X intchng Y
-8.a     60      acc+
-8.b     01        1
-8.c     27      arrow up
-8.d     60      acc+
-9.0     61      recall
-9.1     64      int X
-9.2     45      print
-9.3     45      space
-9.4     13      recall A
-9.5     22      roll up
-9.6     26      ent exp
-9.7     02        2
-9.8     36      mult
-9.9     25      arrow down
-9.a     27      arrow up
-9.b     64      int X
-9.c     34      subtract
-9.d     22      roll up
-a.0     24      Y intchng
-a.1     15      reg F
-a.2     25      arrow down
-a.3     27      arrow up
-a.4     64      int X
-a.5     34      subtract
-a.6     30      X intchng Y
-a.7     33      add
-a.8     24      Y intchng
-a.9     15      reg F
-a.a     35      divide
-a.b     26      ent exp
-a.c     03        3
-a.d     35      divide
-b.0     12      recall E
-b.1     33      add
-b.2     04        4
-b.3     21      point
-b.4     11        9
-b.5     52      if X LT. Y
-b.6     16      { c
-b.7     04      {  4
-b.8     25      arrow down
-b.9     45      print
-b.a     01        1
-b.b     30      X intchng Y
```

THIS IS THE PRINT OUT.

IN THIS SECTION THE NUMBER THAT IS TO BE PRINTED OUT FIRST IS CALCULATED AND SAVED IN THE F REGISTER AS EXPLAINED ON 9.

THIS INCREASES THE QUESTION NUMBER EACH TIME THROUGH THE LOOP. THEN PRINTS IT OUT.

HERE THE VOTES FOR EACH RESPONSE IS SEPERATED, ONE AT A TIME. SEE PAGE 9.

THE NUMBER OF STUDENTS STORED IN THE DECIMAL PART OF REGISTER F IS SEPARATED FOR CALCULATING THE PERCENTAGES.

THE PERCENTAGE IS CALCULATED, IN THE WAY SPECIFIED ON PAGE 10, IN ORDER TO BE ADDED TO THE RESPONSE NUMBER.

THIS SECTION ASKS IF FIVE PERCENTAGES FOR THIS QUESTION HAVE BEEN CALCULATED, SO THAT IT CAN RECALL THE NUMBERS FOR THE NEXT QUESTION, AS EXPLAINED ON PAGE 10.

Appendix A

Program Listing Cont'd

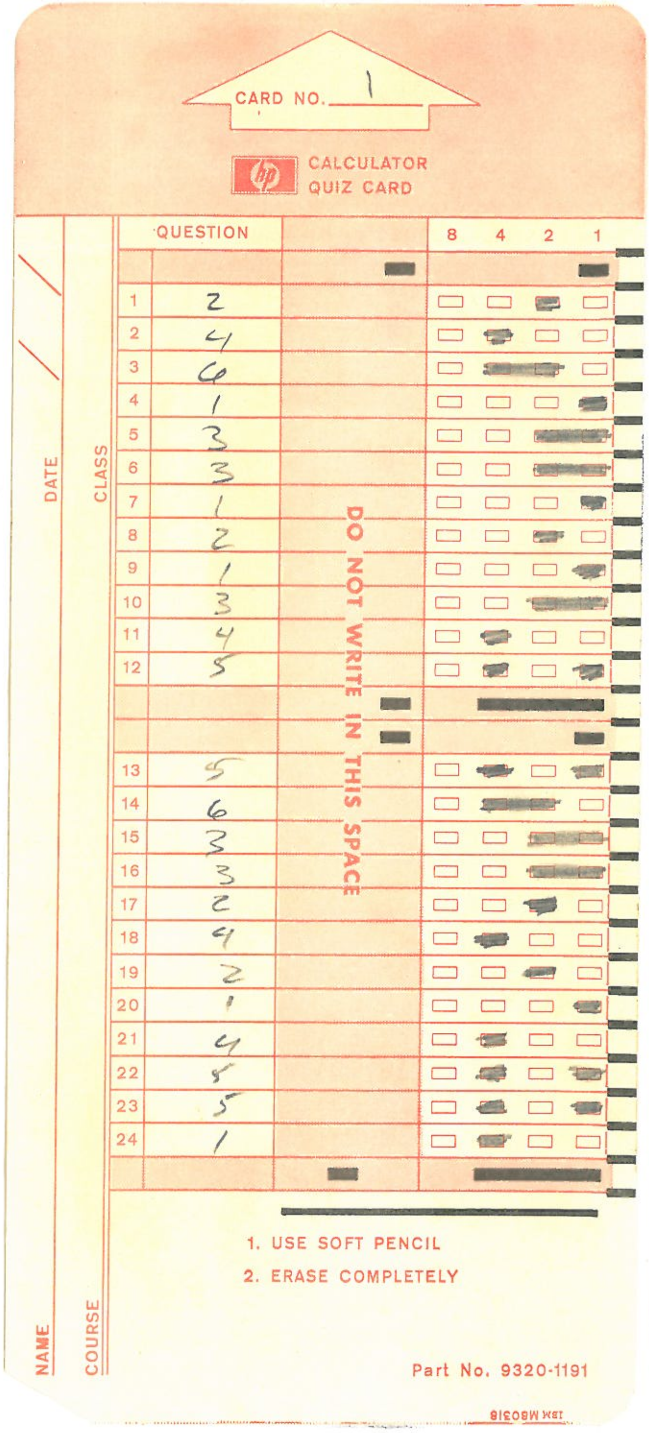
location	code #	instruction
-b.c	37	clear X
-b.d	60	acc+
-c.0	25	arrow down
-c.1	44	go to
-c.2	11	9
-c.3	06	6
-c.4	25	arrow down
-c.5	45	print
-c.6	45	space
-c.7	45	space
-c.8	44	go to
-c.9	77	subroutine
-c.a	06	6
-c.b	02	2
-c.c	37	clear X
-c.d	23	X into
-d.0	12	reg E
-d.1	15	recall F
-d.2	64	int X
-d.3	27	arrow up
-d.4	01	1
-d.5	02	2
-d.6	35	divide
-d.7	25	arrow down
-d.8	27	arrow up
-d.9	64	int X
-d.a	52	if X LT. Y
-d.b	10	{ 8
-d.c	14	B
-d.d	46	END

THIS SECTION IS CONTIUED FROM PAGE 17, IN WHICH THE RESPONSE # IS INCREASED BY 1, THEN RETURNS TO WHERE THE NEXT TWO DIGITS ARE SEPARATED.

THIS LAST SECTION PRINTS OUT THE FIFTH RESPONSE FOR EVERY QUESTION. THEN IT GOES TO THE ROLL-UP SUBROUTINE, RESETS THE RESPONSE COUNTER TO ONE, THEN IT CHECKS TO SEE IF THE RESPONSES FROM ALL 12 QUESTIONS HAVE BEEN PRINTED OUT. IF IT HAS , THEN IT ENDS, BUT IF IT HAS NOT PRINTED THEM ALL OUT, THEN IT WILL RETURN TO THE BEGINNING OF THE LOOP, WHERE THE QUESTION NUMBER IS INCREASED BY 1,, AS EXPLAINED ON PAGES 10 AND 11.

Appendix B

Sample Inputs



CARD NO. 1

CALCULATOR QUIZ CARD

DATE

CLASS

QUESTION		8	4	2	1
1	2	□	□	■	□
2	4	□	■	□	□
3	6	□	■	■	□
4	1	□	□	□	■
5	3	□	□	■	■
6	3	□	□	■	■
7	1	□	□	□	■
8	2	□	□	■	□
9	1	□	□	□	■
10	3	□	□	■	■
11	4	□	■	□	□
12	5	□	■	□	■
13	5	□	■	□	■
14	6	□	■	■	□
15	3	□	□	■	■
16	3	□	□	■	■
17	2	□	□	■	□
18	4	□	■	□	□
19	2	□	□	■	□
20	1	□	□	□	■
21	4	□	■	□	□
22	5	□	■	□	■
23	5	□	■	□	■
24	1	□	■	□	□

DO NOT WRITE IN THIS SPACE

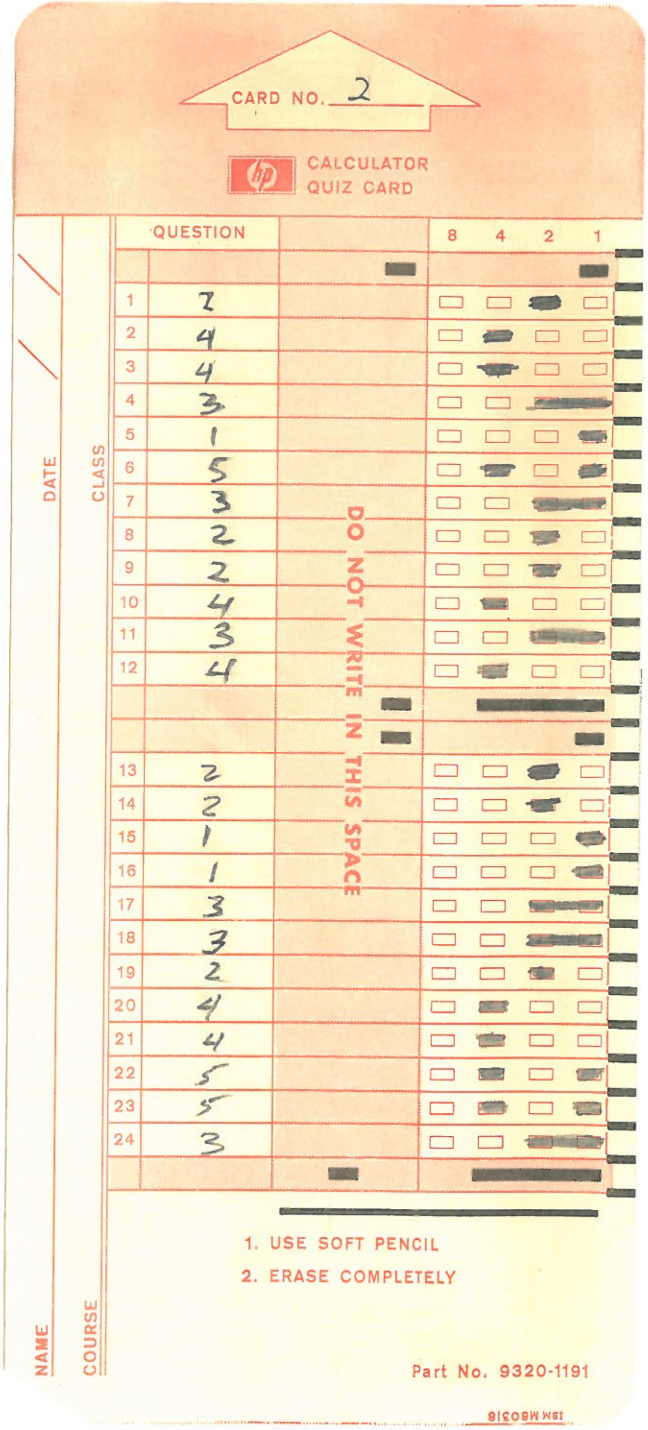
1. USE SOFT PENCIL
2. ERASE COMPLETELY

NAME

COURSE

Part No. 9320-1191

IBM M80318



CARD NO. 2

CALCULATOR QUIZ CARD

DATE

CLASS

QUESTION		8	4	2	1
1	2	□	□	■	□
2	4	□	■	□	□
3	4	□	■	□	□
4	3	□	□	■	■
5	1	□	□	□	■
6	5	□	■	□	■
7	3	□	□	■	■
8	2	□	□	■	□
9	2	□	□	■	□
10	4	□	■	□	□
11	3	□	□	■	■
12	4	□	■	□	□
13	2	□	□	■	□
14	2	□	□	■	□
15	1	□	□	□	■
16	1	□	□	□	■
17	3	□	□	■	■
18	3	□	□	■	■
19	2	□	□	■	□
20	4	□	■	□	□
21	4	□	■	□	□
22	5	□	■	□	■
23	5	□	■	□	■
24	3	□	□	■	■

DO NOT WRITE IN THIS SPACE

1. USE SOFT PENCIL
2. ERASE COMPLETELY

NAME

COURSE

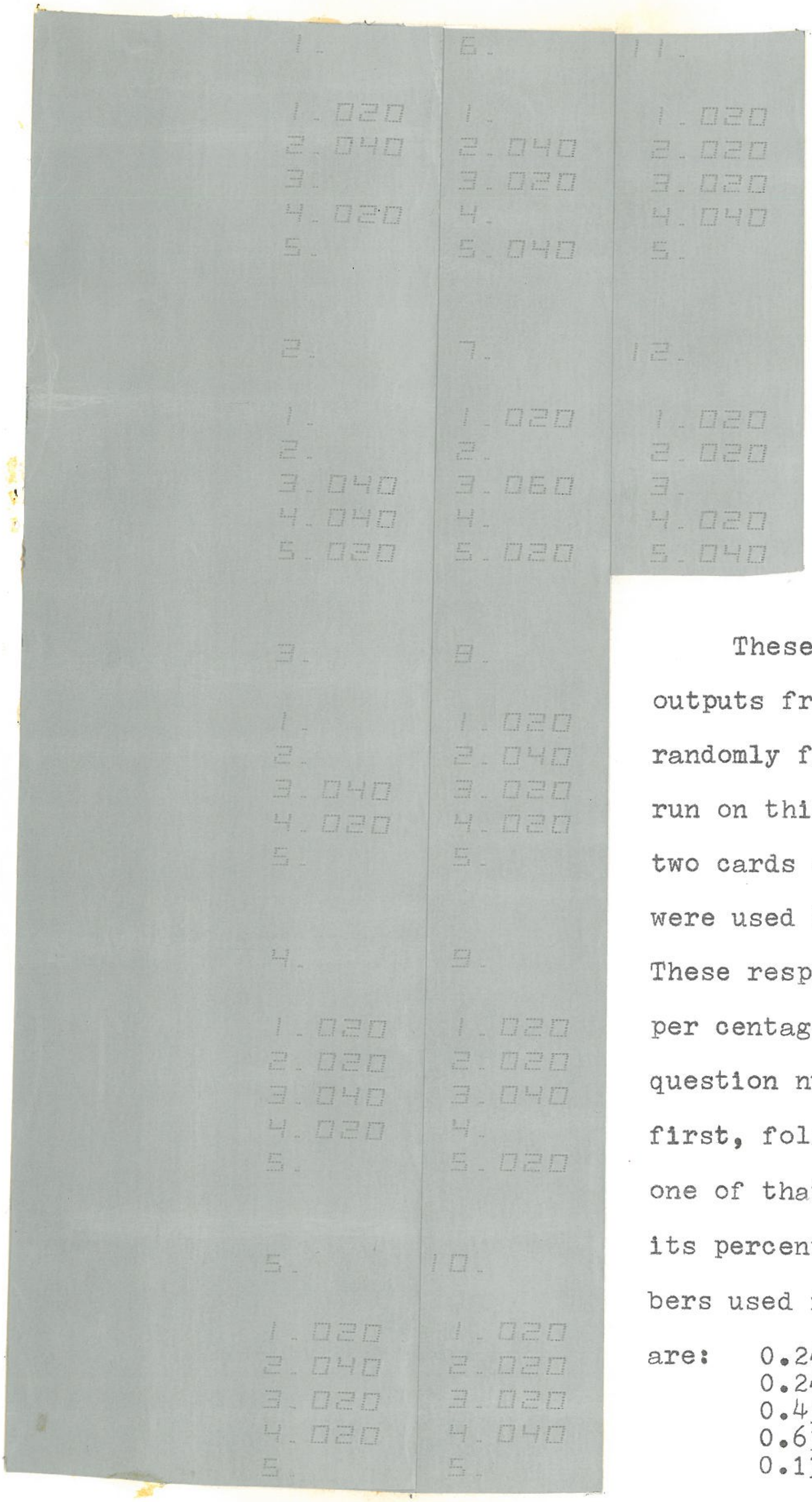
Part No. 9320-1191

IBM M80318

These are two sample cards that students could have filled out. Note that on card number one, questions 3 and 14 are filled out as 6's. The Teacher Evaluation program as it is now will not tally 6's. It also will not tally 0's,7's, 8's, or 9's.

Appendix C

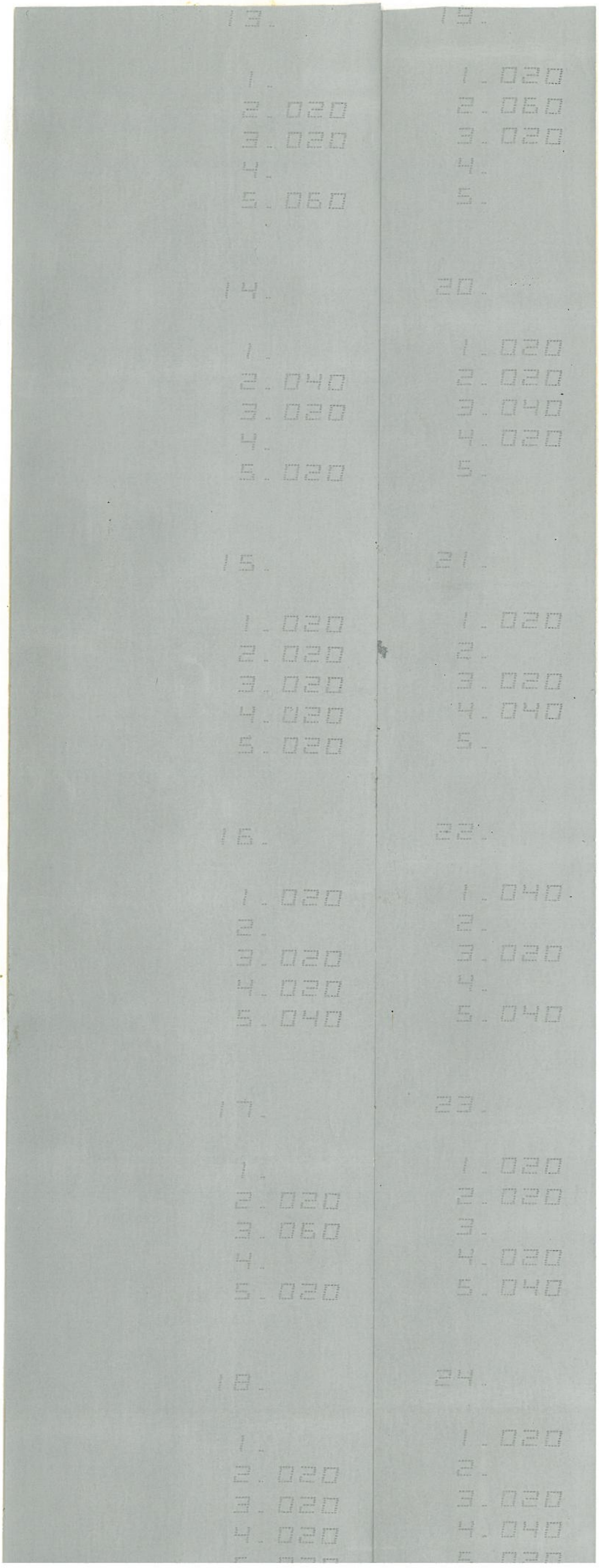
Sample Output



```
    1.          6.          11.

    1.020       1.          1.020
    2.040       2.040       2.020
    3.          3.020       3.020
    4.020       4.          4.040
    5.          5.040       5.


    2.          7.          12.

    1.          1.020       1.020
    2.          2.          2.020
    3.040       3.060       3.
    4.040       4.          4.020
    5.020       5.020       5.040


    3.          8.

    1.          1.020
    2.          2.040
    3.040       3.020
    4.020       4.020
    5.          5.


    4.          9.

    1.020       1.020
    2.020       2.020
    3.040       3.040
    4.020       4.
    5.          5.020


    5.          10.

    1.020       1.020
    2.040       2.020
    3.020       3.020
    4.020       4.040
    5.          5.
```

These are sample outputs from 5 cards randomly filled out, and run on this program. The two cards on page 19, were used in this. These responses are in per centage form. The question number is given first, followed by response one of that question with its percentage. The numbers used for this tally are:

```
0.246133121345
0.244315322434
0.453325533112
0.637442313245
0.133222345421
```



```
13.

1.
2.020
3.020
4.
5.060


14.

1.
2.040
3.020
4.
5.020


15.

1.020
2.020
3.020
4.020
5.020


16.

1.020
2.
3.020
4.020
5.040


17.

1.
2.020
3.060
4.
5.020


18.

1.
2.020
3.020
4.020
5.020
```

```
19.

1.020
2.060
3.020
4.
5.


20.

1.020
2.020
3.040
4.020
5.


21.

1.020
2.
3.020
4.040
5.


22.

1.040
2.
3.020
4.
5.040


23.

1.020
2.020
3.
4.020
5.040


24.

1.020
2.
3.020
4.040
5.020
```

Appendix C

Sample Output

These reproductions of computer tape are a second set of sample responses done as were those on page 20.

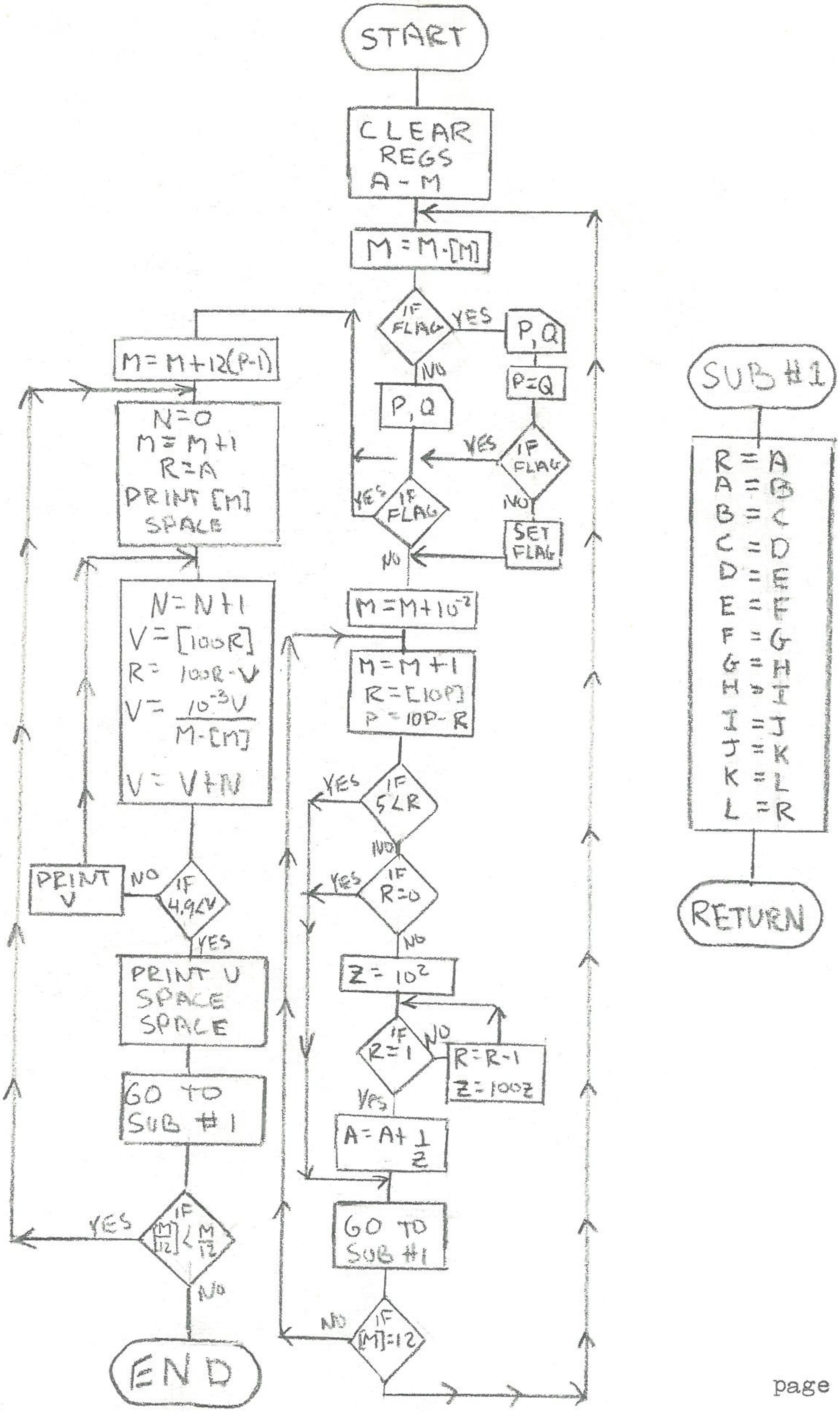
The percentages are from 5 cards randomly filled out as before. Note, as in 18, 19, and 21, the percentages don't add up to 100%.

The numbers used to obtain these responses were:

0.563324214551
0.221133244553
0.552455233111
0.534536336324
0.325532121145

Flowchart

Appendix D



Appendix E

Sample Questions

Some sample questions directly from the survey as used at Saint Elizabeth's, are as follows:

If your response is "1", then fill in box 1.

If your response is "2", then fill in box 2.

If your response is "3", then fill in boxes 1 & 2.

If your response is "4", then fill in box 4.

If your response is "5", then fill in boxes 4 & 1.

1: Does this teacher dress neatly (refers to clothes, shoes, ect.)

1.) Always

2.) Usually

3.) Sometimes

4.) Seldom

5.) Never

2: Does this teacher explain lessons clearly?

1.) Always

2.) Usually

3.) Sometimes

4.) Seldom

5.) Never

Other questions, with the same codes for the responses were:

3. Does this teacher keep pupils from bothering others?

4. Does this teacher allow pupils to give their own ideas?

5. Does this teacher make lessons interesting?

6. Does this teacher make pupils want to learn?

7. Is this teacher respected by the students in the class?

Applications in Business

This program, of the Teacher Evaluation, can be used for any type of survey. It can be used with any number of questions in which up to six responses can be answered. By altering the original program slightly, six responses can be tallied, as explained on page 11. In a similar manner the original program can be change to print out the numbers of votes instead of the percentages. This change is step -a.a to code # 47, continue. This same basic program may be rewritten, by removing the print out program, and the section that clears the registers, to tally up to 9 or 10 different responses.

This type of program may be used for voting, in a small business. It may also be used by manufacturers to simplify order forms, the answer number filled in would be an item while the responses were the quantities. The number of uses of this program is unlimited.