

Computer Science 120
Course Outline

21.

- I. Any questions, comments, etc.?
- II. Discuss any difficulties or questions about last test.
- III. Review grading procedure and W policy

IV. MAT functions --

A. DIM statement (determines space limit; includes zeroth element or row zero and column zero which are NOT used by MAT functions.)

B. MAT READ

```
10 DIM D(10,20)      !limit of 11 x 21 = 231 elements
20 MAT READ D        !reads from DATA into the array, row by row
```

MAT READ D(7,12) would change the size and shape of D but not the maximum number of elements --one subscript might even exceed the original limits.

C. MAT PRINT

punctuation: none, comma, semicolon
100 MAT PRINT \$;

D. MAT INPUT

```
200 MAT INPUT A$     !note the possibility of a string matrix
```

E. ZER, CON, IDN

```
100 MAT A = CON      !sets elements of A each equal to one
120 MAT B = ZER(5)   !to redimension B and zero the array
130 MAT C = IDN      ! sets elements on principal diagonal to 1, others to 0.
                    ! these are the elements for which the row # = col #.
```

F. Calculation with matrices.

1. requirement of conformability

2. Sample statements

```
1000 MAT A = B+C      !performs matrix addition
1000 MAT D = B - C    !performs matrix subtraction
1000 MAT E = (K)*C    !scalar multiplication--each element of E is
                     set equal to K times the corresponding element
                     of C

1000 MAT A = E        !copies E into A
1000 MAT E = TRN(A)   !E is A with rows and columns interchanged
1000 MAT F = C*B      !matrix multiplication
1000 MAT G = INV(A)   !finds the inverse of a square matrix
1020 V = DET          !after an inversion, the variable DEF contains
                     the value of the matrix which was inverted.
```

```
1600 MAT A = B * C
```

Reforms Matrix Multiplication

G. Simple Sample: Suppose G is a matrix of grades with a row for each student and a column for each test. Let W be a ~~row~~ ^{column} vector of matrix of weights for these grades. Then $H = G*W$ is a ~~row~~ ^{column} vector matrix containing the weighted sum of grades.

H. Assignment:

read about MAT in both texts
do problem #2 of Ch. 10 in Pressley

Day #2.

I. Program Efficiency Hints

A. to save storage space in core

1. Combine statements on a line to save statement header space
(Statement headers are 6 words long)

Certain statements require headers anyway

DATA, DEF, DIM, FEND, FOR, NEXT, and REM

use ! within statements lines for your comments and save the overhead

2. reduce the requirements for constant storage by requiring that each value be stored only once.

Consider the progressive improvements in program code:

```
10 AZ = 278Z
```

```
20 BZ = 278Z
```

vs. 10 AZ = 288Z : BZ = 278Z

vs. 10 AZ = 278Z : BZ = AZ

vs. 10 AZ, BZ = 278Z

3. reduce the need for subscripted addressing (this also saves time)

```
10 For iZ = 1Z to NZ
   : T = X(iZ)
   : S = S + T
   : S2 = S2 + T*T
20 NEXT iZ
```

4. eliminate unnecessary variables

```
10 A = B+C : D = A+E : F = D + G
```

becomes

```
10 F = B+C + E + G
```

if the intermediate values A and D are not needed elsewhere

5. use integer variables whenever possible (use % sign)
as subscripts, many FOR ... NEXT loops, etc.

6. Use variables with the same first character as much as you can
don't introduce a new first character unnecessarily

7. Re-use variables when you can. Make them do double duty --but you
pay the penalty of making it more probable that you will make
errors in your program logic.

xxxx.

B. to save computer time; writing faster code

1. use subroutines instead of user-defined functions
2. implied loops are faster than FOR ... NEXT loops (saves space also)
3. multiple IF statements might be replaced by compound IF statements
or by ON GO TO statements

```
80 ON INSTR(1Z, 'XKBY', A$) GO TO 100, 200, 300, 400
```

will go to 300 if A\$ is B

Day #2. (cont)

II. Ideas for homework.

- A. Write a user-defined function which is the inverse of DATE\$() -- i.e. which will give the date for any integer from 1 to 29365. Test your results by

FND(DATE\$(NZ)) = NZ ???

A more ambitious project is to make one which will work from 1 to 32365. Note how DATE\$ works for years after 1999.

- B. Compare multiplication with exponentiation -- which is faster???

$X \uparrow N$ or $X * X * X * \dots * X$

or $N = 0?$ for $N = 3?$ FOR $N = 6, 7, 8$ etc. Is there a crossover point ~~at~~ at which it is better to change your code??

- C. Write a program which is the inverse of the TIME\$() function.
D. Write a program to find the number of minutes and seconds since the beginning of the year.

Day #3. Information Storage Formats.

I. Binary number system

- A. used in computers because of its simplicity
- B. 2^n different symbols can be represented with n zeroes and ones
- C. one binary digit = 1 bit of information

II. Byte (8 bits)

- A. $2^8 = 256$ different things to be represented with one byte.
- B. ASCII character code, appendix D.2
allows for 256 different characters (plenty?)
- E. each character of a string is stored as one byte
Example: the character : is 58_{10} or 00111010_2 in the ASCII code

III. Word.

- A. in the PDP 11/40, 2 bytes or 16 bits make one computer word.
~~xxxxxx~~
- B. integers are stored in one word on our system
 - 1. the bits are numbered from 0 thru 15
 - 2. bit # 15 is the 'sign bit' : 0 for + and 1 for -
 - 3. bits 0-14 contain the value of the integer in binary notation
 - 4. limit on size:
 $2^{15} = 32768$ things (the numbers from 0 thru 32767)
 - 5. Complement form for negative numbers. (two's complement)
-3 is stored as the two's complement of 3 (3 = 000000000000011)

$$\begin{array}{r} -3 = 1111111111111100 + 1 = 1111111111111101 \\ -7 \qquad \qquad \qquad \qquad \qquad = 1111111111111001 \end{array}$$
 - 6. smallest integer is -32768.
-0 is not needed, so 1000000000000000 is used as -32768

1+largest integer = smallest integer

$$\begin{array}{r} 0111111111111111 \\ \hline 1 \\ \hline 1000000000000000 \end{array}$$

also $32767 + 2 = -32767$ etc...

IV. Two word storage of floating point numbers.

- 1. first word
 - a. bit 15 is again the 'sign bit' as before
 - b. bits 7-14 contain the binary exponent in excess 128 notation
 - c. bits 0-6 contain the high order (most significant part of the) mantissa
~~xxxx~~ hidden bit trick
- 2. second word
 - a. bits 0-15 contain the low order digits of the mantissa
- 3. effectively 24 bits of mantissa; $2^{24} = 16,777,216$ or 7^+ decimal digits
- 4. the power of 2 can be from -128 to 255

$$\begin{array}{l} 2^{-128} = 2.938735912 \times 10^{-39} \\ 2^{127} = 1.701411815 \times 10^{38} \end{array}$$

Day #31 (cont)

5. sample storage for floating point numbers

$$1385_{10} = 10101001110_2 = .1010100111 * 2^{11}$$

since 1385 is positive, the sign bit is 0

add 128 to the exponent: $139_{10} = 10001011_2$ (exponent bits)

omitting the first bit after the binary point, insert the mantissa of the number ('hidden bit')

word 1: 0100010110101001 (or as 2 bytes) 01000101 10101001

word 2: 1100000000000000 (or as 2 bytes) 11000000 00000000

V. Computer numbers

1. limited precision
2. not a continuous set --and the gaps are of different sizes in different parts of the number system.

as seen above, there is no number between 0 and about 1.4×10^{-39}

VI. functions which convert the storage format from one to another

1. $Y\% = \text{ASCII}(A\$)$ returns the decimal value of the first character in $A\$$, using the ASCII code
2. $Y\$ = \text{CHR}\$(X\%)$ returns the string character represented by the value of $X\%$, using the ASCII code

3. $\text{CVT}\$(, \text{CVTF}\$, \text{CVT}\$(X, \text{CVTF}\$)$

NB: each of these functions swaps the order of the bytes
the F functions swap the word order also so that a number in floating point format in bytes 1,2,3,4 is converted to a string with the bytes in 4,3,2,1 order.

4. $\text{CVT}\$(A\$, N\%)$ --see table in DEC manual for significance of various values of $N\%$. Values may be added for combined effects.
5. $Y\$ = \text{NUM}\(A) produces the string that would be printed by `PRINT A`
6. $Y = \text{VAL}(A\$)$ if $A\$$ contains the characters that form a valid number.
7. $Y\% = \text{SWAP}\$(N\%)$ interchanges the two bytes of $N\%$, giving new integer $Y\%$
8. CHANGE $A\$$ to $X\%$ where $X\%$ is a list and will contain the ASCII values of the characters of $A\$$. $X\%(0)$ is the length of $A\$$.
9. CHANGE $X\%$ to $A\$$
10. $\text{STRING}(L\%, C\%)$ creates a string of $L\%$ characters long where each is the character represented by $C\%$ in ASCII code.

VII. Write a program to determine and print each bit of the representation of the floating point number 678.456.

Computer Science 120
Course Outline

Day #4. Logic

I. Integer variables as logical variables

- A. truth values : T = 1Z; F = 0Z; any non-zero value will be True
- B. Program switches --set somewhere in program and test later

AZ = (p\$ = "stop")
 BZ = (X > 12)
 CZ = DZ and 4Z
 IF AZ and (BZ OR CZ) then 1280

II. bit manipulation --using less than full word for data storage

See truth tables in DEC manual

AZ = 0110 (data word) of course there are 16 bits in our computer word --shortened these for simplicity
 BZ = 1010 (mask word)

function	result	comment
AZ AND BZ	0010	logical product. copy selected bits of data word & set others to 0. can be used to copy part of a word, to see if certain bits are set or set certain bits to 0.
AZ OR BZ	1110	to set selected bits to 1 and copy all others. logical sum.
AZ XOR BZ	1100	logical difference --shows, bit by bit, where A and B differ.
AZ EQV BZ	0011	shows, bit by bit, where A and B match
AZ IMP BZ	1010	the order of add A and B matters here sets all bits of A except where A has a 1 and B has a 0.

Examples:

If (AZ AND NOT 7Z) = 0 same as IF AZ < 7Z ??
 IF (NOT AZ and 7Z) = 0 if last three bits of A are 1s
 CZ = AZ AND 240Z copy bits 4-7 of A into C; perhaps this part of the word represents data we want.
 CZ = AZ or 12Z sets bits 2 & 3 to 1s, copies all others
 CZ = AZ ornot 12Z copies bits 2&3 and sets all others to ones.

needed operation s:

- 1. copy word segment CZ = AZ and 240Z
- 2. set certain bits to 1 without changing others AZ = AZ or 12Z
- 3. set certain bits to 0 without changing others AZ = AZ AND NOT 12Z
- 4. test bit N IF (AZ and 2Z↑NZ) = ~~0~~ Then (or test for 0) ^{2Z↑NZ}
- 5. set bit N AZ = AZ and 2Z↑NZ
- 6. turn off bit N AZ = AZ and NOT 2Z↑NZ



Day #5. Random numbers.

I. Pseudo-random number generation (one way)

A. $R_n = C * R_{n-1} \pmod{N}$

$N=5$
 $C=2$
 $R_0=1$
1, 2, 4, 3, 1, 2, 4, 3, ...
Choose large N (32768)
Choose C=2
Choose $R_0 = \text{odd}$

B. concept of repetition length

1. define
2. depends on choice of N, C and R_0
 - a. repetition length increases with N, so choose $N = 2^b - 1$ (32768)
 - b. choose C of the form $8n \pm 3$ where n is any positive integer
 - c. choose R_0 to be an odd integer
 - d. choose C near the square root of N for better statistical properties, although not increased repetition length.

32767

C. divided R_n by N to get a number in the range 0 to 1

II. Statistical tests on random numbers (and random digits)

- A. mean test -- should be near 0.5 in the above case
- B. Frequency distribution test -- equal intervals should contain equal counts of randomly generated values.
- C. frequency of runs test. A run of length k is a series of k consecutive numbers which is monotonic (either increasing or decreasing)

To eliminate gap - CHANGE N every number of times.

there should be $\frac{2}{(k+3)} [N(k^2+3k+1) - (k^3+3k^2-k-4)]$

runs of length k in a series of N random numbers

runs above and below the mean test

--- look for sequences which are either all above or all below the mean. there should be

$\frac{1}{(k+3)} [N(k^2+3k+1) - (k^3+3k^2-k-4)] \frac{(N-k+3)}{2^{k+1}}$

such runs of length k in N numbers.

gap test for random digits

each digit should recur, in any one sequence, every 10 times (on the average) i.e. the average "gap" between repetitions of a digit should be 10. This should be true for each of the digits

$$\text{Variance} = \frac{\sum_{i=1}^N (g_i - \bar{g})^2}{N}$$

VARIANCE = average value of the squared deviation from the mean.

For random digits, the VARIANCE of the gaps (as above) should be 90

$g_i = \text{current gap}$
 $\bar{g} = \text{expected mean}$

The maximum test

In a set of three random digits, the center one will exceed the other two in 285 out of 1000 cases.

gap = 3
1 2 3

III. Uses of Random numbers.

1. Simulation

representing portions of a non-deterministic process or of one in which the deterministic rules are unknown or difficult to apply

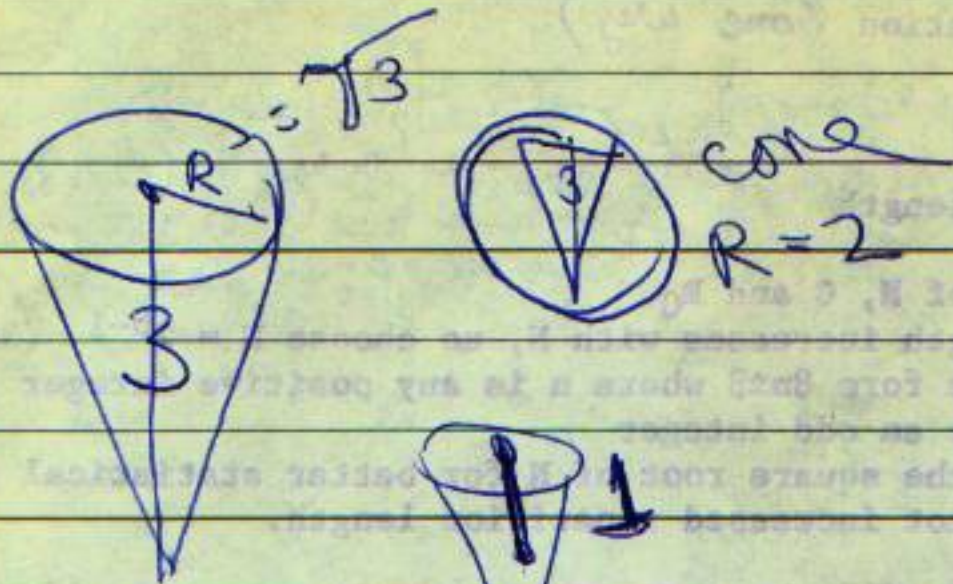
2. Monte Carlo techniques applied to definite integrals area, volume, center of mass, etc.



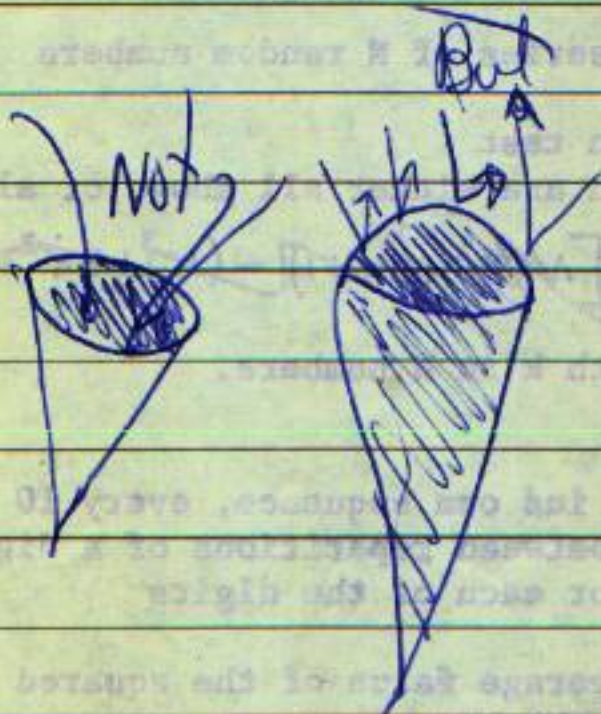
Possible program assignments:

1. Write your own pseudo-random number generator and select poor values (then good values) of R_0 , C and N, and find the repetition length in each of several cases.





find volume of cone



END

7 10 10 7

700 FORMAT (A210H)

READ A1Y 700H JPLY JSL

~~Assignment~~

Find from Int 1-200
those in which bits 2, 4 & 6
are ~~off~~ but #5 is on
& Int
Switch on/off states of 5 & 6
& Int (New)
Copy 4 low-order bits of
original #.