DATA REPRESENTATION

- In digital systems like computers, the quantities are represented by symbols called digits.

 # Digital Number System: In digital representation, various number systems are used. The most common number systems used are decimal, binary, octal and hexadecimal.

 # Decimal Number system

→ The decimal system is composed of 10 numerals or symbols. These 10 symbols are – 0, 1, 2, 3, 4, 5, 6, 7, 8, 9

→ It is also called base-10 system because it has 10 digits.

 # Binary Number system:

- unfortunately, the decimal number system does not lend itself to convenient implementation in digital systems for eg:- it is very difficult to design electronic equipment so that it can work with 10 different voltage levels. • On the other hand, it is very easy to design simple accurate electronic circuits that operate with only two voltage levels.

- In the binary system there are only two symbols or possible digit values, 0 and 1.

↳ The binary system is also a positional-value system, wherein each binary digits has its own value or weight expressed as power of 2.

- In the binary system, the term Binary digit is often abbreviated to the term bit.

 # Octal Number system

- The octal number system has a base of eight, meaning it has eight unique symbols : 0, 1, 2, 3, 4, 5, 6, 7.
- The octal system is also a positional value system, where in each octal digit has its own value or weight expressed as powers of 8

$(23)_{10}$ =

2	23	
2	11	1
2	5	1
2	2	1

Hexadecimal Number System ( )$_{16}$

- The hexadecimal system use base 16. Thus, it has 16 possible digit symbols.
- It uses the digits 0 through 9 plus the letters A, B, C, D, E, F.
- Hexadecimal system is also a positional-value system, wherein each hexadecimal digit has its own value or weigh expressed as power 16.

Number Conversions:

1. Decimal - to Binary conversion

There are two procedures for converting (integers) from decimal to Binary.

→ The first method :

Ex 1) 43_{10} to binary –

32 is nearest number without exceeding 43,

$\therefore 2^5 = 32$, Now Put 1 at $(5+1)^{th}$ Position and 0s at all other,

$32 = 100000$, Now, $43 - 32 = 11$

$2^3 = 8$, $11 = 1000$

$11 - 8 = 3$, Now, $2^1 = 2$

$3 = 10$

$1 = 1$

$\therefore$ Adding all the binary nos obtained, $\underline{101011}$ (Ans)

→ Ex. 2) 200_{10}

$128 = 2^7 = 10000000$

$200 - 128 = 72 = 2^6 = 1000000$

$72 - 64 = 8 = 2^3 = 1000$

$\therefore 200 = 11001000$.

Ex: 3) 832_{10}

$2^9 = 512 = 1000000000$

$832 - 512 = 320 = 2^8 = 100000000$

$320 - 256 = 64 = 2^6 = 1000000$

$832_{10} = (1101000000)_2$

1000000000
100000000
1000000
1101000000

The second method of converting decimal to binary is Repeated division method.

→ In this method number is successively divided by 2 and its remainders recorded.

Ex.1) 43_{10}

2 | 43		
2 | 21	1	↑ LSB
2 | 10	1	
2 | 5	0	
2 | 2	1	
2 | 1	0	
0	1	MSB

$\therefore 43_{10} = 101011$

Ex.2) $200_{10} = 11001000$

2 | 200	
2 | 100	0
2 | 50	0
2 | 25	0
2 | 12	1
2 | 6	0
2 | 3	0
2 | 1	1
0	1

Ex.3) 832_{10}

2 | 832	
2 | 416	0
2 | 208	0
2 | 104	0
2 | 52	0
2 | 26	0
2 | 13	0
2 | 6	1
2 | 3	0
2 | 1	1
0	1

$832_{10} = 1101000000$

Binary-to-Decimal Conversion: Method I :-

Ex:1- $(11011)_2$

Sol) $2^4.1+2^3.1+2^2.0+2^1.1+2^0.1 = 16+8+0+2+1 = 27_{10}$

Ex.2) $(101011)_2$

$2^5.1+2^4.0+2^3.1+2^2.0+2^1.1+2^0.1 = 32+0+8+0+2+1 = (43)_{10}$.

Method II ; Dibble-dobble method -

Ex.1) $(101011)_2$

$1\times2=2+0=2\times2=4+1=5\times2=10\times2=20+1=21\times2=42+1=43$.

Ex.2) $(11011)_2$

$1\times2=2+1=3\times2=6+0=6\times2=12+1=13\times2=26+1=27_{10}$ (Ans)

Decimal-to-Octal Conversion

→ A decimal integer can be converted to octal by using the same repeated division method that we used in the decimal-to-binary conversion, but with a division factor of 8 instead of 2.

Ex.1) $(266)_{10} = (412)_8$.

8 | 266		
8 | 33	2	↑ LSD
8 | 4	1	
0	4	↑ MSD

Octal to Decimal Conversion:

An octal number can be easily converted to its decimal equivalent by multiplying each octal digit by its positional weight.

Ex.1) $(412)_8 = 4\times8^2 + 1\times8^1 + 2\times8^0 = 256 + 8 + 2$
$= (266)_{10}$.

Ex.2) $(24.6)_8 = 2\times8^1 + 4\times8^0 + 6\times8^{-1} = 16 + 4 + \frac{3}{4} = (20.75)_{10}$

$= \frac{80+3}{4} = \frac{83}{4}$

4 | 83 (20.75
-8
30
-28
2

Octal to Binary Conversion:

The conversion from octal to binary is performed by converting each octal digit to its 3-bit binary equivalent.

Octal digit	0	1	2	3	4	5	6	7
Binary digit	000	001	010	011	100	101	110	111

Ex.1) $(412)_8 = 100001010_2$

Ex.2) $(5431)_8 = 101100011001_2$

Binary to Octal Conversion:

Ex.1) $101011_2 = 53_8$

Ex.2) $100011101 = 435_8$

NOTE: Sometimes, the binary no. will not have groups of 3 bits. For those cases, we can add one or two 0s to the left of MSB of the binary no.

Decimal-to Hex Conversion:

Decimal-to-hexadecimal conversion can be done using repeate division by 16.

Ex.1) $(423)_{10} = 1A7_{16}$

16 | 423
16 | 26 — 7
16 | 1 — A
0 — 1

Ex.2) $(214)_{10} = D6_{16}$

16 | 214
16 | 13 — 6
0 — D

Hex-to-Decimal Conversion:

A hex no. can be converted to its decimal equivalent by using the fact that each hex digit position has a weight that is a power of 16.

Ex.1) $(356)_{16} = 3\times(16)^2 + 5\times16 + 6\times16^0 = 256\times3 + 80 + 6$
$= 768 + 86 = 854_{10}$

Ex.2) $2AF_{16} = 2\times16^2 + 10\times16 + 15\times16^0 = 512 + 160 + 15$
$= 527 + 160 = 687_{10}$.

Binary-to-Hex Conversion:

Binary numbers can be easily converted to hexadecimal by grouping in groups of four starting at the binary point.

Hex	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Binary	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111

Ex.1) $1010111010_2 \rightarrow 0010\,1011\,1010_2 = 2BA_{16}$

Ex.2) $10101110.01011_2 \rightarrow 10101110.01011100_2$
$= AE.5C_{16}$

Hex-to-Binary conversion:

Each hex digit is converted to it 4-bit binary equivalent.

Ex.1) $9F2_{16} = 100111110010_2$

Representing Unsigned Integers in Binary:

An unsigned integer can be either a positive integer or zero but never a negative integer.

⇒ In n decimal digits, you can write a number between 0 and $10^n - 1$.

Analogously, in the binary number system,

→ an unsigned integer containing n bits can have value between 0 and $2^n - 1$.

Binary Addition

ALU's process only binary numbers as computers only understand only binary numbers.

There are several rules for addition –

i) $0 + 0 = 0$

ii) $0 + 1 = 1$

iii) $1 + 0 = 1$

iv) $1 + 1 = 10$ equal to 0 and carries 1

v) $1 + 1 + 1 = 11$ equals 1 and carries 1.

Ex) i) $52 + 40 = 92$

```
  110100
  101000
 1011100
```

2 | 52	2 | 40
2 | 26 0	2 | 20 0
2 | 13 0	2 | 10 0
2 | 6 1	2 | 5 0
2 | 3 0	2 | 2 1
2 | 1 1	2 | 1 0
0 1	0 1

1011100

$2^6 \times 1 + 2^5 \times 0 + 2^4 \times 1 + 2^3 \times 1 + 2^2 \times 0 + 2^1 \times 0 + 2^0 \times 0$

$= 64 + 0 + 16 + 8 + 4 = 64 + 28 = 92$

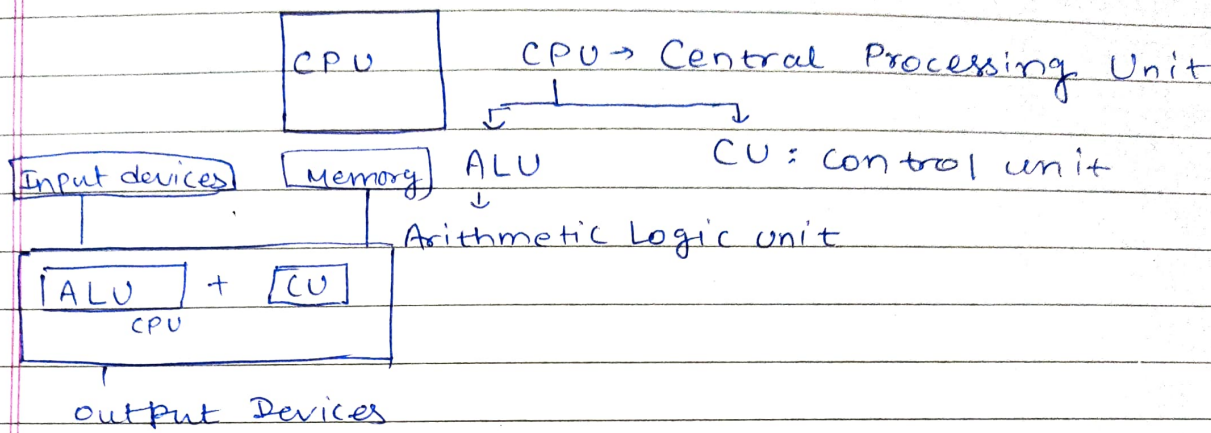
Ex.2)
```
 01010111
 00110101
 10001100
```

Ex.3)
```
  1011
   110
 10001
```

Ex.4)
```
  1111 0
    11
 100001
```

Ex.5)
```
  11.01
 101.11
1001.00
```

Introduction to Computers



Memory →

- CPU Registers
  Cache memory
- Primary memory (Ram) & (gfx)
- Secondary memory (HDD)
  (TB)

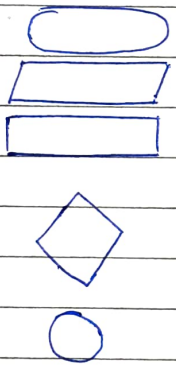
Page:
Date: 11 / 10 21

• An algorithm is a finite set of instructions that if followed accomplishes a particular task.

- Input : Zero or more input
- output : one or more
- Definitiness : each and every instructions is clear and unambigious.
- Finitiness :- the algorithm must terminates after sometime
- Effectiveness :- when a person follows the algorithm using pen and paper the result will be same.

FLOWCHART

Pictorial representation of algorithm.



Start / end

input / output

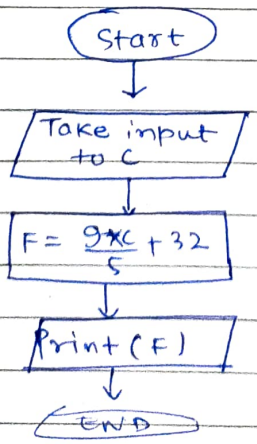
Calculation

if - else

Concatenator.

Q.1) Draw the flowchart that takes the celcius (C) temp. as input and produce Fahrenheit (F) temperature as output.

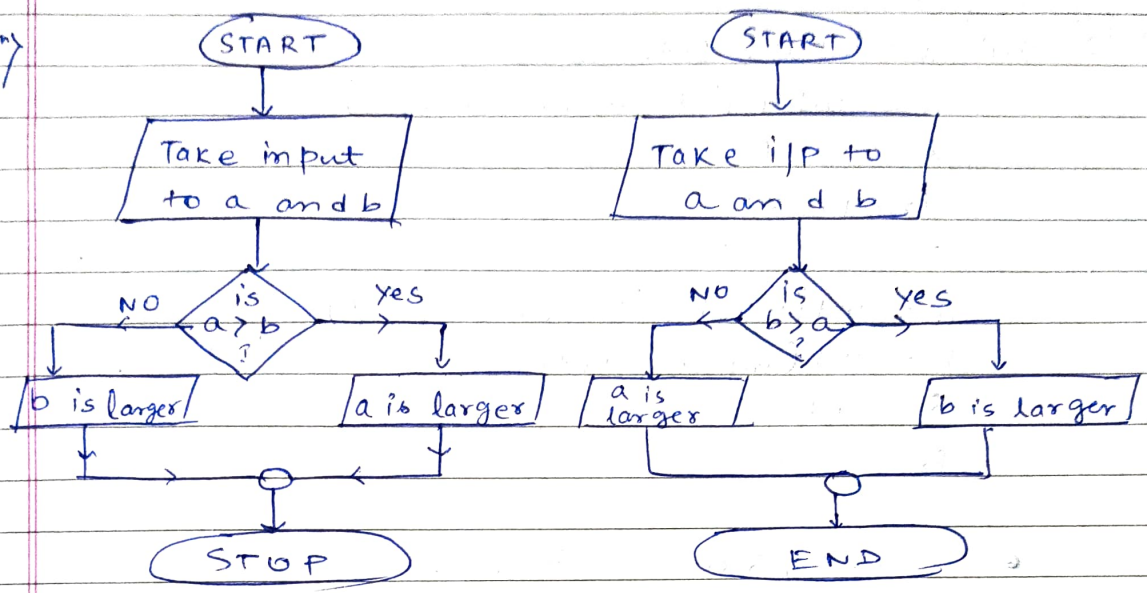
Soln >



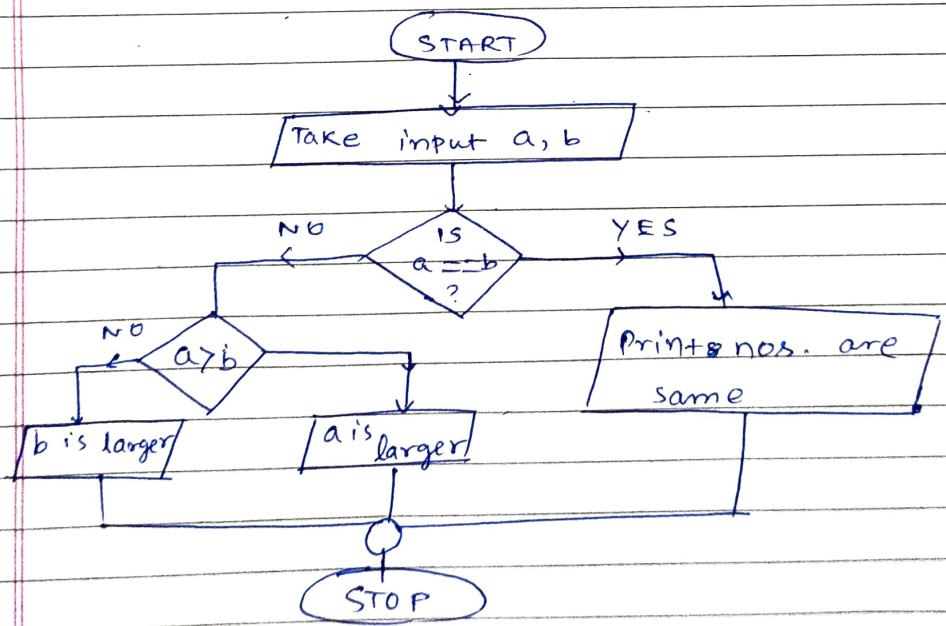
Page :
Date : / /

Q.2) Draw the flowchart to take two distinct nos. as input and return the large number as output

Sol^n)



Q.3) Draw the flowchart to take any two nos. as i/p and return the larger as o/p if they are differt otherwise return the numbers are same.



Page :
Date : / /

Q.4 WAP in python to take two numbers as i/p and calculate their sum.

Sol)
```
a = input("Enter a no.")
b = input("Enter another no.")
s = a+b
Print(s)
```

```
a = int(input("Enter a no."))
b = int(input("Enter another no."))
S = a+b
Print(s)
```

- 10, 20, 30 → integers
- 10.5 11.35 ⇒ float
- 'a', 'c', '2' ⇒ char
- 'abc', 'p,q,r' '10' ⇒ string

Page :
Date : 12 / 11 / 21

PYTHON ARITHMETIC OPERATIONS

```
a = int (input ("Enter the value of a"))
b = int (input("Enter the value of b"))
c = a+b
Print ("Sum =", c)
d = a-b
Print ("Subtraction =", d)
e = a* b
print ("Mult =", e)
f = a/b
Print ("Div =", f)
g = a % b
print ("Remainder =", g)
h = a** b
Print ("Power =", h)
i = a // b
Print ("Floor Division =", i)
```

a += b
⇒ a = a+b .

Assignments

Q.1) WAP in python to take Celcius temperature as input and produce Fahrenheit temperature as output.

Q.2) WAP in python to calculate the simple interest

Q.3) WAP in python to calculate the area and perimeter of a rectangle.

Q.4) WAP in python to calculate the area and perimeter of a circle.

Q.5) WAP in Python to calculate the salary of employee with the following information:

Total salary = Basic + DA + HRA

DA = 50% of Basic.

HRA = 15% of Basic

Page:
Date: 18 / 11 /21

Decision Making:

if Statement :
if - else statement :
if - elif - else statement :
Nested if statement :

Q. WAP in Python to take any integers as input and print the maximum number.

Sol)
```
a = int(input("Enter the value of a:"))
b = int(input("Enter the value of b:"))
if a>b :
    Print("a is larger")
else :
    Print("b is larger")
```

Q. WAP in Python to take any integers as input and return the larger as output if they are different otherwise return the numbers are same.

Sol)
```
a = int(input("Enter the value of a"))
b = int(input("Enter the value of b"))
if (a==b):
    Print("The numbers are same")
else:
    if (a>b):
        print("a is larger")
    else:
        Print("b is larger")
```

Q.3) WAP in python to take three distinct integers as input and print the maximum value.

Q.4) WAP in python to take one integer as i/p and Print whether it is even or odd.

Sol)
```
a = int (input ("Enter your Number"))
if a%2 == 0 :
    Print("The number is EVEN...")
else :
    Print("The number is ODD...")
```

Q.3) Ans)
```
a = int (input ("Enter the first number"))
b = int (input(" Enter the second number"))
C = int (input("Enter the third number"))
if (a>b) :
    if (a>c) :
        Print ("a is greatest")
elif (b>a) :
    if (b>c) :
        Print ("b is greatest")
else :
    Print ("c is greatest")
```

Page:
Date: 19 / 11 / 21

Q.1) WAP in python to take total marks as i/p and display the following:

if marks > 100 — invalid input

marks ≥ 90 and ≤ 100 → grade is outstanding

" " ≥ 80 " ≤ 90 → " " Excellent

" " ≥ 70 " < 80 → " " Fine

" " ≥ 60 " < 70 → " " good

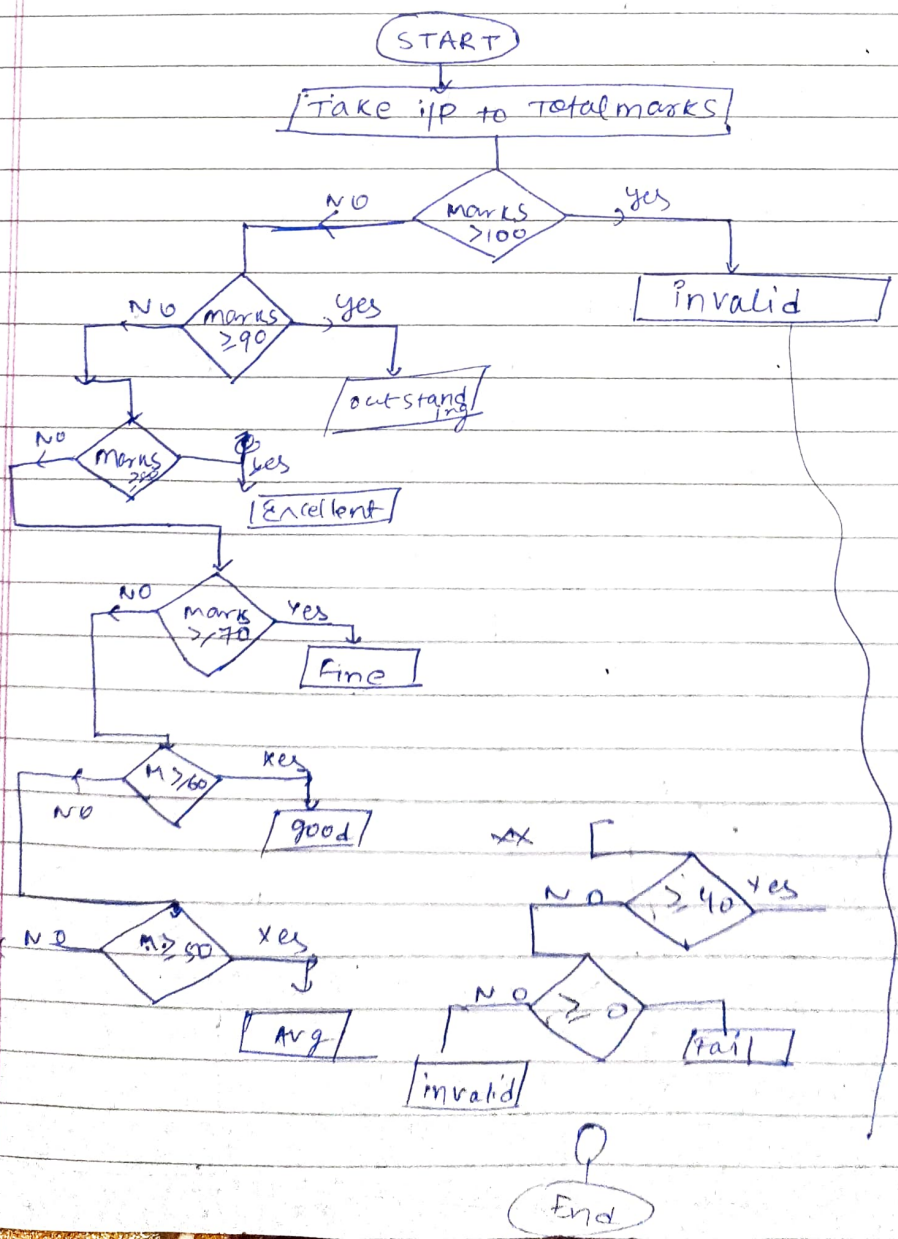
≥ 50 " < 60 → " " average

≥ 40 " < 50 → " " just pass

≥ 0 " < 40 → " Fail

< 0 - invalid input.

⇒ Flowchart

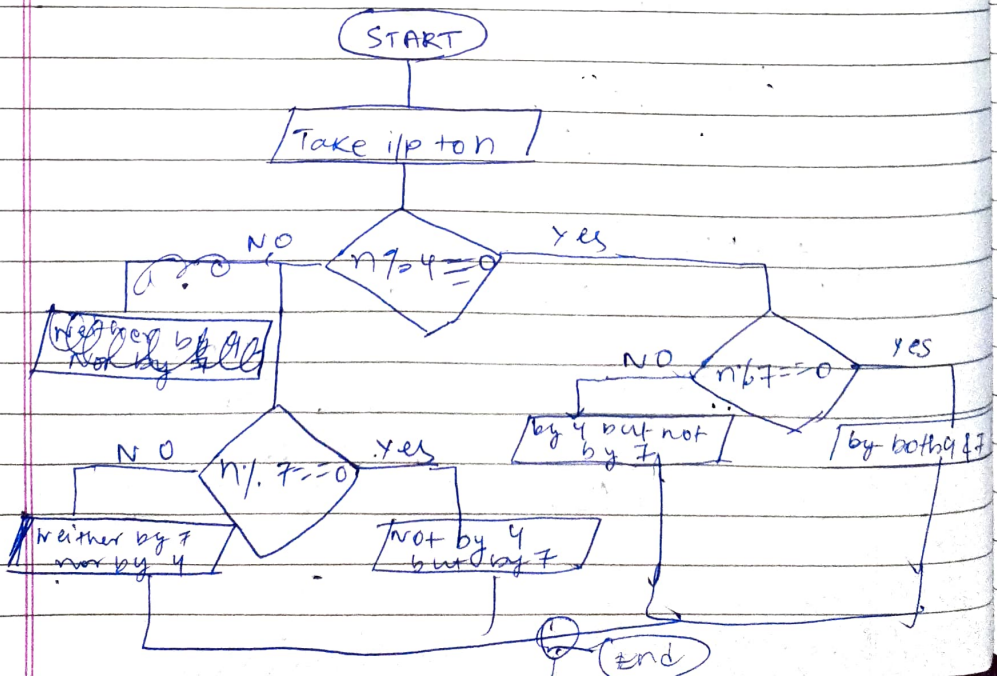


```
marks = int(input("Enter total no."))
if (marks >100):
    Print ("Invalid")
elif (marks >=90):
    Print("outstanding")
elif (marks >=80):
    Print("Excellent")
    : :
    : :
    : :
elif(marks >=40):
    Print("Just Pass")
elif (marks >=0)
    Print("Fail")
else :
    Print ("invalid")
```

Nested-if

28

if n=28, O/P : NO. is divisible by both 4 & 7
n=21, O/P : NO is " by 7 but not by 4.
n=16, O/P : NO is " " by 4 but " " 7.
n=9, " : NO. is neither by 4 nor by 7.



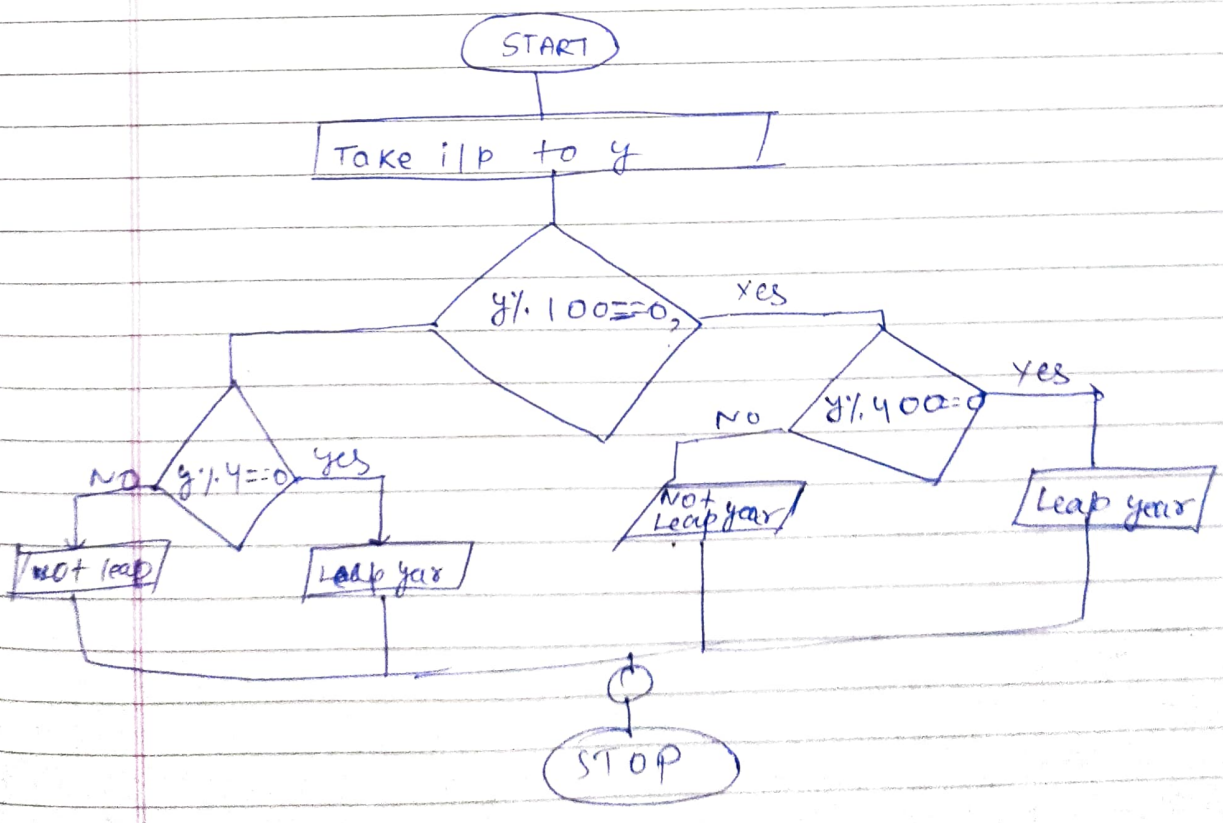
```
n= int(input("Enter a no."))
if (n%4==0):
    if (n%7==0):
        Print("Both by 4 and 7")
    else:
        Print("by 4 but not by 7")
elif (n%7==0):
    Print("Not by 4 but by 7")
else:
    Print("Neither by 4 nor by 7')
```

Q3) WAP in python to check whether a year is leap year or not.

300, 1900, 2100, 2000 by 400

2024, 2048 ⟶ by 4.

if century year then divide it by 400 & check otherwise divide 4 and check.



```
y=int(input("Enter year"))
if (y%.100==0):
    if (y%.400==0):
        Print ("Leap year")
    else:
        Print("Not Leap year")
elif (y%.4==0):
    Print("Leap year")
else:
    Print("Not Leap year")
```