enum

PageWork

D D M M Y Y Y Y

Data Structures

> Mathematical or logical representation of data into comp memory.

Data Structures

↙ Set of Values

↘ Organisation

Data — single item
group item

```
for (i=1; i<=5; i++)
    printf("%d", i);
```

executes 6 times

Executes 5 times

> time complexity
> Space Complexity

PageWork
D D M M Y Y Y Y

Classification of Data Structures

1) Linear & non linear

Linear → Sequential
Non-Linear → Non Sequential

2) Homogenous & Heterogenous Non Homogenous

3) Static vs Dynamic

Static → Mem allo at Compile time
∴ Fixed

Dynamic → Runtime allo

4) Primitive vs Non primitive

PageWork
D D M M Y Y Y Y

Diff types of DS

1. Arrays

→ Collection of homogenous elements in a continuous memory

→ Arrays are linear in nature

→ Static

→ Non primitive

```
int A[10];
```

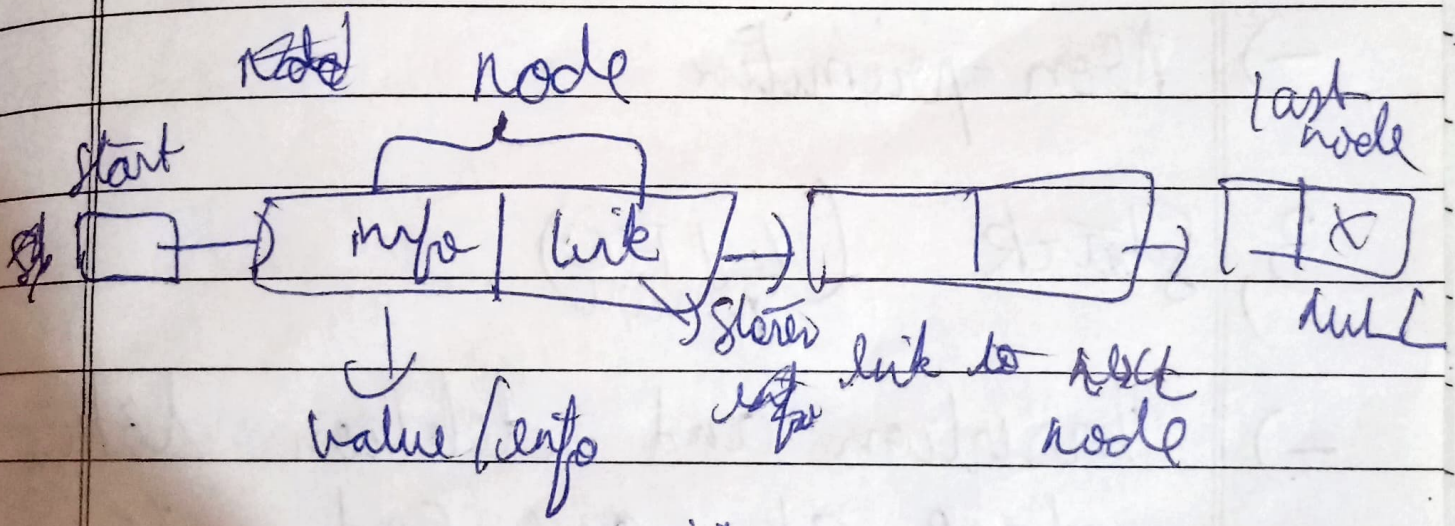
Time required to read any element of an array is same.

2) Linked List

node

→ Collection of ~~nodes~~ where a node has two parts:

(i) Info and (ii) Link



- There's special ~~node~~ pointer of node type known as start → stores address of 1st node.
- End of LL is indicated by NULL ptr.

PageWork
D D M M Y Y Y Y

-) Storage is non linear,
-) Access is linear

-) LL is homogenous

-) Dynamic

-) Non-primitive

3) Stack (LIFO) principle

-) Insertion and deletion takes place at one end ie 'top'

-) homogenous
-) Linear

PageWork
D D M M Y Y Y Y

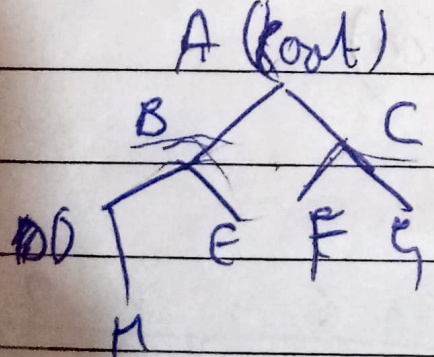
4) Queue (FIFO)

→ Insertion take place at the rear end and deletion take place at front end.

→ homogenous
→ Linear

5) Trees

→ hierarchical representation of data



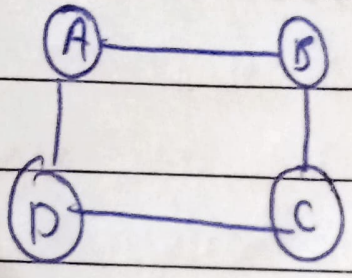
A → Root node
H, E, F, G → Leaf nodes

Def DSA Explain diff DSA Classification

PageWork D D M M Y Y Y Y

6) Graphs

-) Representation of data in vertices and edges.

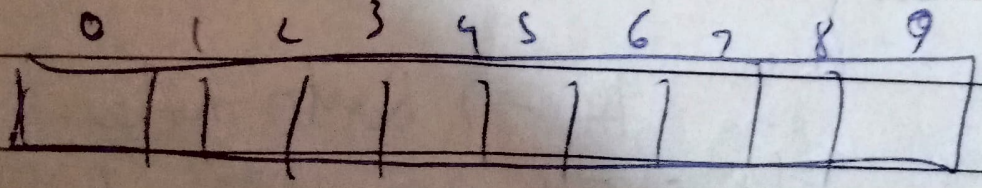


- incidence matrix
-

Different ~~Application~~ Ops on DS

1) Create int A[10];
(Declaration)

2) Insertion



Traversing : Accessing each element atleast once

3, Deletion : removal, num of elements decrease

4, Updation : Change value, num of elements remain same

5, Searching : To find whether an element is present or not.

6, Sorting (arranging elements in particular order
(bubble, insertion, merge, Selection)

7, Copying : one to another

8, Merging :

9, Reversing :

10, Destruction

Step by step procedure to solve a problem in English literal language

↑

Algorithms

PageWork
D D M M Y Y Y Y

? Literal representation of problem solving methanism.

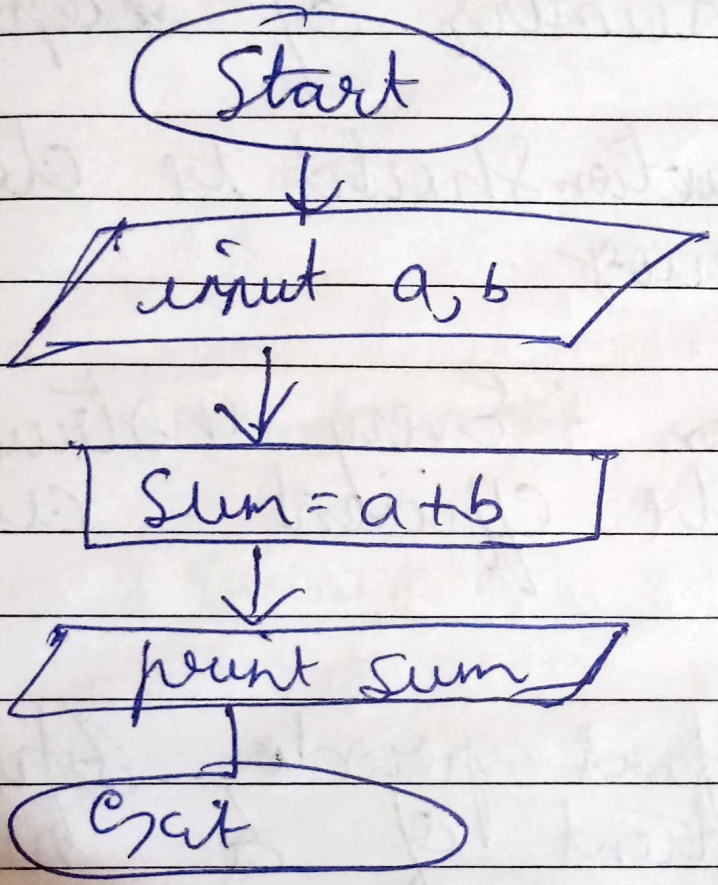
e.g two add two nos.

1. Input two numbers
2. Calculate sum
3. display sum.

```
#include <stdio.h>

int main ()
{
    int a, b;
    scanf ("%d", a);
    scanf ("%d", b);
}
```

printf("%d", a+b);
}



Alogrithm

1, Zero or more inputs

2) Should return an expected output. (finite)

3) finite number of steps

4) Each instruction should be clear and unambiguous.

5) Effectiveness: Every instruction must be efficient and feasible.

9) WAA to find product, sum, diff and quotient of 2 numbers.

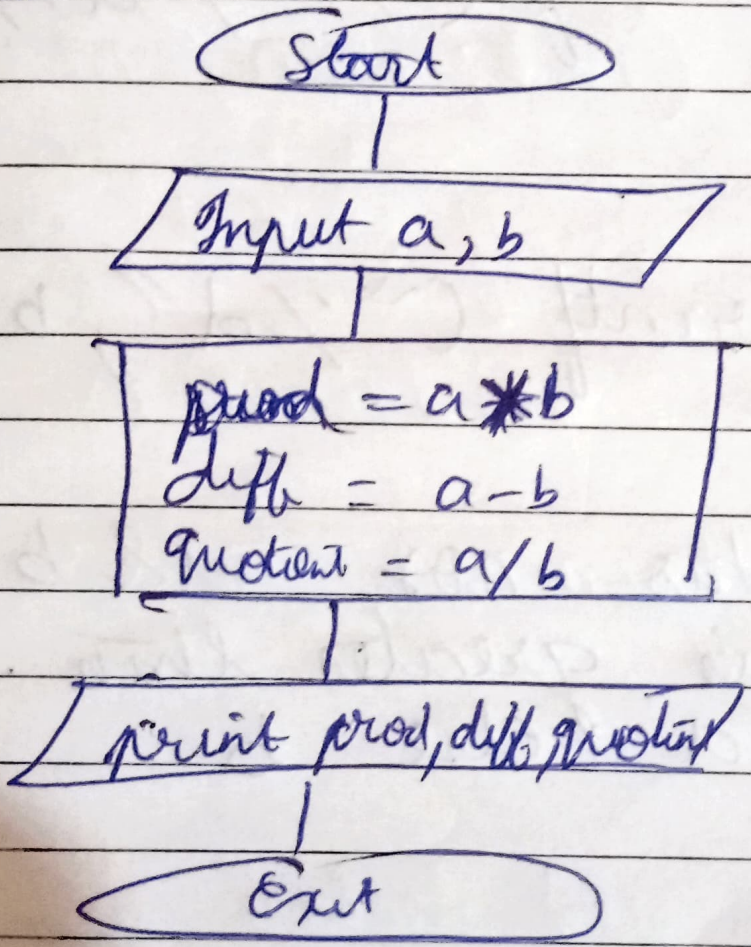
1. Input two numbers
2. Calculate product, difference and quotient of them.

PageWork

D D M M Y Y Y Y

3. Display product, difference and quotient on screen

4. Exit



```
if (a>b) {
    printf ("%d", a);
}
else {
    printf ("%d", b);
}
```

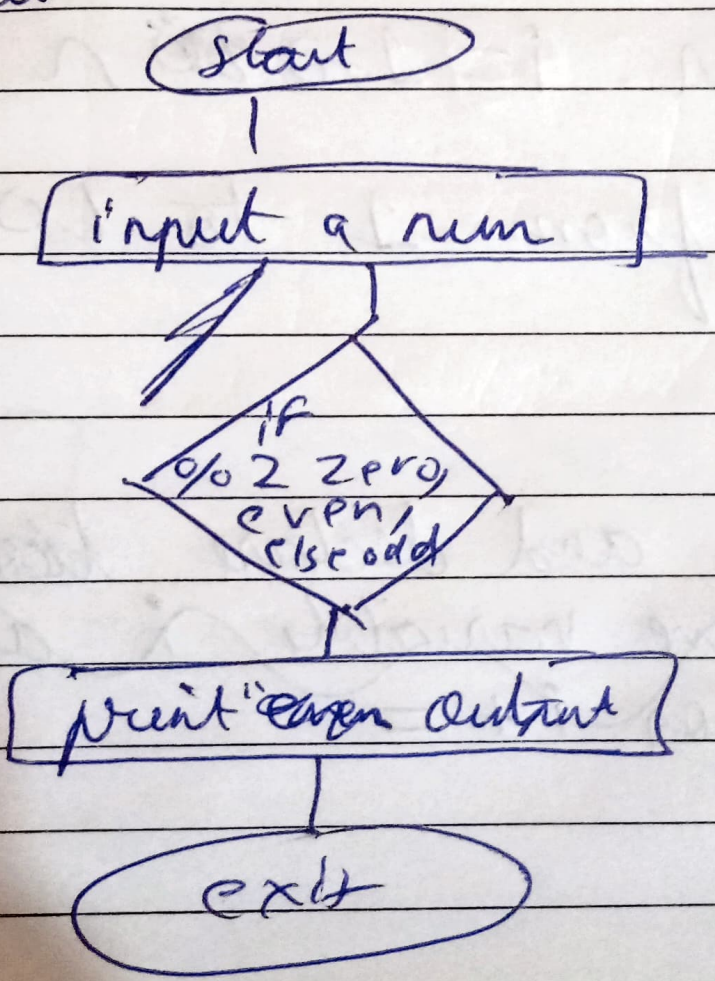
1. Read two nos. a & b
2. If a is greater than b, print a else b
3. End

PageWork

D D M M Y Y Y Y

Algo to check if even or odd

1. Start
2. Read a number
3. If remainder when divided by 2 is zero, print "Even", else "odd"
4. Exit



PageWork
D D M M Y Y Y Y

Q. Write print first n natural numbers

2. Starts int i=0;
   for (i=0; i<=n; i++)
       printf ("%d", i);

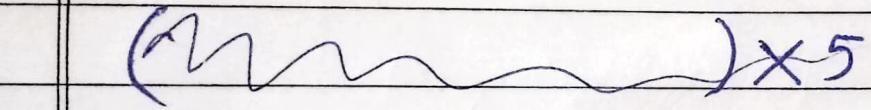
- Repeat for i=1 to n
  or
  Repeat i from 1 to n

- Start
- Input n and declare iterative variable i as zero
- Repeat for i+=

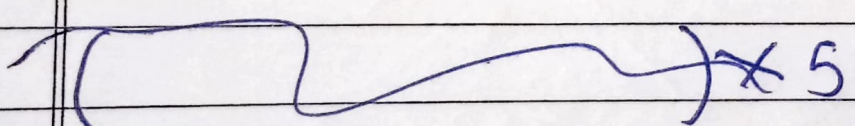
Sum of n

PageWork
D D M M Y Y Y Y

- Start
- Input n and declare iterative variable i
- Declare sum = 0
- Repeat for sum+ = i from i to n
- Display sum
- Exit



Recursion
Brute force
Divide and Conquer
Greedy algo
Dynamic programming



ARRAYS

PageWork
D D M M Y Y Y Y

A[10]

LB = 0, UP = 9

Length = (UB − LB) + 1

= (9 − 0) + 1 = 10

X

A[−15 : 5] X