**ARRAYS**

Why to use array?

What if we need to store data , then what to do?

Let’s make variable but if user needs to store 100+ variable

Then we can’t make 100 variable (this is not feasible )

So we request a block that holds 100 variable space

Ex int a ;

Here we are requesting space for one variable

* int a[] = new int [100];
* here we are requesting a block of 100 variable

a[i]=refer to value at ith index

a[10]=refer element at 10th index but represent 11 element of our data

Let’s use STACK and HEAP

Local Variable goes to stack

Object goes to heap

(arrays are represented by object)

If I used => int a=5

Int b[]=new int[5]

* Heap contain block of 5 variable
* Stack contain variable a (Local Variable)
* And it also contain variable with name b and it holds address of first index of memory allocated in heap for block of 5 variable
* As main gets end => a and b lost from memory but heap remain contain it
* Left memory in heap is removed by garbage collector

Lets take a situation

Int a=2

Int b =3

b=3

sop(a)

sop(b)

it will give output 2 3

i.e change in value of b , doesn’t affects value of variable a

Situation 2 =>

Int arr1;

Got empty variable in stack with name arr1

Arr1=new int[10];

Now heap got memory of block having 10 variable space

And stack got reference of block in heap inside arr1 variable of stack

Int arr2=arr1;

Now stack got another variable with name arr2 and having reference of block made in heap earlier

Now both arr1 and arr2 is pointing to same reference so change in value of arr1 , also affect changes in arr2

Indeed => there exist only one memory i.e both are making changes in one block

Therefore changes cause by one, also affects others

Now as main gets end => heap memory gets unreferenced memory

And it will be removed by garbage collector

**DEFAULT VALUE OF ARRAY ELEMENT**

Arrays By Default => give values to array

Like int a[] holds 0 in all variable

Bool a[] holds false in all variable

Float and double hold 0.0

//now in code

Array Demo

Multiple References in Arrays

Traversing Array

Flow of Arry