1.Where const variables[Read-Only] are stored?
Ans: const variables are stored in text/code segment
2.Where static variables are stored?
Ans: static variables are stored in data segment
3.Where do x,y,z,m,k,t store in Memory in below code?

```
int x; //uninitialized Global variables stores in uninitialized data segment[BSS]
int t=10;/initialized Global variable stores in initialized data segment

int main()
{
static int y; //unitialized static variable stores in uninitialized data segment[BSS]
static int m =20;//initialized static variable stores in initialized data segment
const int k=10; // Read only type stored in text/code segment
int z; //local variables store in stack memory
}
```

2.Without deleting the memory,How can we allocate the memory?
Ans: int *ptr = (int*)realloc(ptr,size);
3.Main function itself is a thread,so whenever we run other threads from main,use pthread_join() to join working threads with the main thread
4.Memory allocated in
a)constructor is freed in destructor
b)copy constructor is freed in destructor
c)assignment operator is freed in destructor
NOTE:We can use the same destructor for all member functions of the class
Deep copy of one object to other is done in copy constructor and assignment operator

5.Does compiler provided copy constructor and assignment operator have Deep copy?
Ans:
No, they have only shallow copy
we have to define our own deep copy for this members

6.Does compiler provided copy constructor and assignment operator have self assignment check?
Ans:
Copy constructor does not need self assignment check,since compiler throws **redeclaration Error** ,x a(2),a=a;
Compiler provided assignment operator does not have self assignment check,we need to define self assignment check condition

```
if(this != &obj)
{
}
return *this;
```

7.Implement String class with all members[Cons,Parameterized cons,Copy cons,Assignment Operator] with i/p string "Global" .

8.Can we do obj=obj for copy constructor?
No, the compiler throws error saying re-declaration of same object
We can avoid this using self assignment check in Assignment operator

9. Difference b/w Copy constructor and Assignment Operator ?
Copy constructor:
=>We need to write our own Deep copy code for preventing default shallow copy of objects[which leads to segmentation fault]
=>we cannot do obj=obj,compiler throws error saying re-declaration of same object,So no self assignment check condition
=>It does not return anything
=>It allows Initializer list

Assignment Operator:
=>We need to write our own Deep copy code for preventing default shallow copy of objects[which leads to segmentation fault]
=>We need to write our own self assignment check condition[if(this != &obj)] to prevent same object[obj=obj] initialization[which leads to Memory wastage for the entire class creation]
=>It returns *this,this,*temp,&temp
=>It does not allow initializer list

10.Example for using initializer list for copy constructor[Mandatory for const and reference variables]

```
class A
{
int x;
const int y;
public:
A(int a,int b):a),y(b)
{
cout<<x<<","<<y<<endl;
}

A(const A &obj):obj.x),y(obj.y)
//A(const A &obj):A(obj.x,obj.y) //Calling Parameterized constructor using Copy constructor
{
cout<<x<<","<<y<<endl;
}

A& operator =(const A &obj)
{
x=obj.x;
//y=obj.y;
return *this;
}
};

int main()
{
A k(2,3),l(k);
}
```

11.What is passed in catch block arguments?value or reference? If value then explain why?

12.Difference b/w size() and capacity() in vector?
13.How does vector allocate memory?

14.How to find duplicate value related keys in map?
15.How to find duplicate keys in multimap?
16.How to create map for class?
17.Difference between static_cast and normal casting?
18.What is the main use of initializer list?
19.Name the containers[sequential and associative] that use **index []** and **function at()** access?
Ans:**vector,deque,map**

20.Push(),Pop() and Emplace() API's are used only for which containers?
Ans: Sequence containers like vector,deque, list,forward_list

21.Name the containers which store the elements in contiguous memory?
Ans:vector and deque

22.What is the difference b/w vector and forward_list?
vector:
Insertion/deletion at back side
Stores elements in contiguous memory
forward_list:
Insertion/deletion at front side
Stores elements in non-contiguous memory

23.How to traverse in reverse in the containers?
Ans: Use iterator with rbegin(),rend()

24.Name the member functions or API's that return iterators ?
Ans:begin()/rbegin()/cbegin()/crbegin() -returns the iterator/reverse iterator/const iterator pointing to the first element
    end()/rend()/cend()/crend() -returns iterator/reverse iterator/const iterator pointing to the last element
    find()
    lower_bound()
    upper_bound()
Eg:
it =m.begin()
it =m.end()
it = m.find()
erase(it); //Only API that takes iterator as input parameter
it =m.upper_bound()->first/second
it = m.lower_bound()->first/second

25.Difference between emplace_front() and push_front(),emplace_back() and push_back()
push_front()- This does not directly place the element in the container,
It first creates a temporary memory,copies the element to it and then places it in the container
emplace_front()-This directly places the element in the container and takes less time compared to push_front()
This same explanation is applicable to emplace_back() and push_back()

26.Explain emplace() and splice()
emplace(position,value) -used to insert values at specific position of same container
splice(position,containerobject)-It transfers/moves complete data of 1 container after specific position of existing container

27.Name the libraries used in map/multimap,set/multiset,list/forward_list,vector/deque
map/multimap -#include<map>
set/multiset -#include<set>
list -#include<list>
forward_list -#include<forward_list>
vector -#include<vector>
deque -#include<deque>

28. Common API's used exclusively for list and forward_list
unique()
merge()
sort()
splice()
reverse()
remove()
remove_if()

29.How to remove duplicate nodes in list?
Ans: unique()

30.Common API's used in map and multimap,set and multiset
find()
count() -returns the Count of no. of times a number is repeating
lower_bound()
upper_bound()
equal_range()

31.Difference b/w upper_bound(x) and lower_bound(x)?
upper_bound(x) -It returns the iterator pointing to after x
lower_bound(x) -It returns the iterator pointing to x

32.What is the default order of set and multiset?
Ans: Both are in Ascending order only. The only difference is multiset will have duplicate values

33.How to traverse in Single,double,circular linked lists?

```
Ans:node->next;
     node->prev;
```

34.Can we create one container from the other of same type? use copy constructor,range based constructor

```
vector<int>v1;

vector<int>v2(v1); //vector<int>v2(v1.begin(),v1.end());
       (or)
vector<int>v2;
 v2=v1;
```

35.How to get the elements printed, when you pop the elements in sequence containers?
Ans:

```
for(int i=0;i<v.size();i++)
{
```

```
cout<<v.back(); //vectors,deque,list
pop_back();
}

for(int i=0;i<v.size();i++)
{
cout<<v.front(); //forward_list,deque,list
pop_front();
}
```

35.How to pass array to a vector
eg:

```
 int x[]={2,4,6,7};
vector<int>v(x,x+4);
v.insert(v.begin(),5); //inserting/replacing index 0 with 5
v.insert(v.begin()+1,3);//inserting/replacing index 1 with 3
v.erase(v.begin());//delete 0th index
v.erase(v.end());//delete last index
v.erase(v.end()-1);//delete index before last element
```

36.How to traverse in BST?
Ans: There are 3ways to traverse in BST

    Preorder[root,root->left,root->right]
    Inorder[root->left,root,root->right]
    Postorder[root->left,root->right,root]

Few questions:
Can we pass map to map?
Can we pass vector/list/set to map?

Additional Info:
Vector/Stack
Deque/Queue
List/DLL
forward_list/SLL
set/multiset, map/multimap -BST
Iterator/Pointer
Algorithms:
search()[Linear,Binary] -2
sort(),merge()[Quick Sort,Bubble Sort,Selection Sort,Insertion Sort,Merge Sort,Heap Sort] -6
find()
count()

1.Generally Global variables are stored in data segment,local variables are stored in stack memory
Then where do Global and local variables of static type stay?
And where do Global and local variables of const type stay?
Where do extern,auto,register,mutable variables stay?

1.What is object slicing?It occurs for upcasting or downcasting?
2.What is initializer list ?Where is it mostly used and why?How is it different from initialization inside the block?
Ans:
This is used for faster initialization of variables in all constructors[default,Parameterised,copy]
Also used for passing the parameters from one cons to other constructor
This is mandatorily used for initialization of const and reference variables

```
A(int a,int b):a),y(b) //Parameterised constructor
{
cout<<x<<","<<y<<endl;
}

A(const A &obj):obj.x),y(obj.y) //copy constructor
{
cout<<x<<","<<y<<endl;
}

int main()
{
A i(1,2),j(i),k;
k=i; //Cannot do this since const variable is there,Compiler throws error
}
```

NOTE:
We cannot use initializer list for Assignment operator
We cannot use copy constructor to call default/parameterised cons using initializer list[DELEGATING Constructor C++11 TBD]

2.what is static linked libraries and dynamic linked libraries?Difference b/w them?
3.what is splice() Api in list?
4.What is capacity(),reserve() Api in vector do?
5.What is the difference b/w capacity() and size() in vector?
6.What is emplace(),emplace_front(),emplace_back(),cbegin(),cend(),crbegin(),crend() api's in stl?
7.what does find(),count(),upper_bound(),lower_bound,equal_range() do in list and map?
8.Difference b/w vector,deque,list,forward_list?
9.Difference b/w set,multiset,map,multimap?
10.Can we reverse vector/deque as list do?

11.Explain each of the below common api's used in all STL containers

push_back(),pop_back()
push_front(),pop_front() - STACKS ...Not present in vector
front(),back()
[]   -Not there in list,forward_list,set,multiset
begin(),end()
rbegin(),rend()
cbegin(),cend()
crbegin(),crend()
insert(),erase()
swap(),clear()
emplace(),emplace_front(),emplace_back()
size(),resize(),max_size(),empty()
capacity(),reserve()   -Only for vector
find(),count(),upper_bound(),lower_bound(),equal_range() -Only for set,multiset,map,multimap

12.Write code for thread free singleton?
13.Can we inherit singleton class?
14.write code for Factory design pattern?
15.How to achieve virtual constructor in C++?
16.write code for Abstract Factory design pattern?
17.what is the difference b/w Factory and Abstract Factory design pattern
18.Write code for Observer design pattern?
19.Write code for Proxy design pattern?
20.Write code for Adapter design pattern?
21.What is the difference b/w malloc() and calloc()?
malloc()
1.non-contiguous memory
2.Unintialised memory
3.No need to specify the no. of blocks

calloc()
1.contiguous memory
2.initialized memory to zero
3.Need to specify the no. of blocks while allocating memory

22.static members in the class can be accessed by other class[neednot be inherited] of the same file
----

1.Can we do obj=obj for copy constructor? If yes/no ,explain

2.Can we do obj=obj for Assignment constructor? If yes/no ,explain

3.What is passed in catch block arguments?value or reference? If value then explain why?

4.If catch block is not written,when any exception occurs and throw statement tries to throw the exception,then What happens to the program?Whether it Runs or terminates?

7.How to find duplicate value related keys in map?

8.How to find duplicate keys in multimap?

9.How to create map for class?

10.What are the different reasons for deadlock conditions?

11.Difference b/w semaphore and mutex?

12.Can we say mutex as Binary semaphore?

13.Difference b/w binary and counting semaphore?

14.How to make a class to have only dynamically created objects and not static objects?

15.Mention few scenarios where only reference must be used?

16.Few differences b/w pass by value and difference?

17.What is the use of function ptr?

```
18.int *ptr =new[];
      delete ptr; //instead of delete [] ptr;
```

Now what happens to this code snippet?

19.Explain the scenarios where you faced Dangling ptr ?

20.Explain the scenarios where segmentation fault occurs?

21.What is the main use of void ptr?

22.Write the code for Dynamic memory allocation and deallocation for 2dimensional Array??

23.What is the scope of static within the class?

```
24. class Base
{
static int x;
}
cout<<sizeof(Base);
```

25.why size of empty class is 1byte?

26.How to find whether the Linked list is circular or not? How to find the middle node?How to find the nth node from given L nodes?

27.How to make the class restrict to only specified number of objects?

28.

```
class Base{
public:
virtual void func1();
virtual void func2();
void func3();
};

class Derived :public Base
{
public:
virtual void func1();
virtual void func4();
void func5();
};
```

Now the Base class ptr can access which of these functions of both the classes?

29.can static functions access normal members??Can normal functions access static members

30.Difference b/w inline function and macro?

31.What is the use of namespace?

32.can you write your own strcpy(),strcat(),strcmp(),strrev(),strlen(),memset(),memcpy(),atoi(),itoa()?

33.

```
int main(int argc,char*argv[])
{
cout<<argc;
}
```

What is the value of argc?

34.without using itoa() function,
   convert decimal(50) to
   a)binary
   b)octal
   c)Hexadecimal

35.Write the code for converting binary/octal/Hexadecimal to decimal formats?

36.Can we have a pure virtual destructor?if Yes, then can that class be called as Abstract class?

37.What is the main use of abstract class?

38.Can we use initializer list for copy constructor and assignment operator?

39.Explain string literals,enum class,nullptr,uniform initialization,final,override,auto,range based for loops,tuple,lambdas,smart pointers[unique,shared,weak,auto],assertions?

40.what is the main difference b/w shared and weak ptr?why the referral count does not change in weak ptr?

41.Write the login for the below i/p strings to get the expected ouput

```
  i/p: "AAABBBBCCD"
    o/p:"A3B4C2D1"

i/p: "ABAMALAYALAMLIRILMADAMHAAH"
  o/p  rint the longest palindrome
```

42.i/p: "Mike get ups in the morning.
            Mike drinks tea and takes bath.
            After Bath,Mike starts to office"

write Logic to Count the no. of unique words from this paragraph

1.Implement Stack using array/ptr/Class/Linked List/C++ STL[vector]
2.Reverse a string using stack
3.Implement Stack using Queue/Queues
4.Implement Queue using array/ptr/Class/Linked List/C++ STL[deque]
5.Implement Queue using Stack/Stacks
6.Implement Singly Linked list using Class/C++ STL[forward_list]
7.Implement Doubly Linked List using Class/C++ STL[list]
8.Implement Circular Linked List using Class
9.Implement Binary Search Tree using class/C++ STL[set/multiset/map/multimap]

NOTE:
Single Linked List always traverses from Left to Right
Circular Single Linked List always traverses from Left to Right
Double Linked List always traverses in both directions
Circular Double Linked List always traverses in both directions

=>If we want to traverse to the last node in linked list

```
while(temp->next !=null)     //struct node *temp =head;
   {
       temp = temp->next;
   }
  struct node *last =temp;
  cout<<temp->data;
```

=>If we want to traverse to the specific node(X) in linked list

```
int count =1;   //since temp points to head[1st node]
while(temp->next !=null && count<X)
   {
       count++;
       temp = temp->next;
   }
  cout<<temp->data;
```

=>If we want to count the nodes in linked list

```
struct node*temp =head;
int count =0;
do
   {
    count++;
    temp=temp->next;
    } while(temp !=null);
```

=>If we want to count the nodes in circular linked list

```
struct node*temp =head;
int count =0;   //since do while() loop is there,count starts from 0
do
  {
     count++;
     temp=temp->next;
    } while(temp!=head);
     cout<<count;
```

=>If we want to Add new nodes in circular linked list

```
struct node *newnode=(struct node*)malloc(sizeof(struct node));
newnode->next= head;
if(head!=null)
    {
     struct node*temp = head;
     while(temp->next !=head)
         {
       temp = temp->next;
         }
  temp->next =newnode;
     }
else //If this is the first node
  {
    newnode->next= newnode;
  }
head =newnode;
```

=>If we want to compare the nodes in linked list

```
            while(temp !=null && temp->next !=null)   //Delete duplicate nodes
                   {
                        temp = temp->next;
                   }

            while(fast=null && fast->next!=null)   //middle,circular,intersecting point
                  {
                      fast = fast->next->next;
                      slow = slow->next;
                   }
```

=>How to find the nth node from the end of linked list?

```
                 struct node*temp=head;
                 int count =1;
                 while(temp!=null && count<N-pos+1)
                {
                 count++;
                 temp=temp->next;
                }
               cout<<temp->data;
```

Comparison of Nodes is used in few cases like:

Finding the middle node
Finding whether the node is circular
Deletion of nodes in sorted/unsorted Linked list
Pairwise swapping of all nodes in the Linked list

NOTES:
=>head ptr,top ptr will point to the latest node and will have the last node as null
=>front ptr,rear ptr will point to the latest node having null
=>Push[Stack], InsertNewNodeAtBegin [Single Linked list] operations are same
=>Pop[Stack],Deque[Queue] and DeleteNewNodeAtBegin[Single Linked list] are same

=>Enque[Queue] is totally different

10.Insertion/Deletion[Needs extra ptr] of nodes at any position(begin,middle,end) in Single Linked List
11.Delete duplicate nodes in sorted and unsorted linked list[takes more time complexity],Also count the no. of duplicates.
[In sorted/unsorted linked list, Tranversing is very important step]
12.Delete alternate nodes in single linked list
13.Reverse Single linked list using Iterative and Recursive method
14.Rotate single linked list by K nodes
15.Given only a single ptr to the specific node in a Linked list,Now how do you delete that node?
Note: Generally to delete a node in SLL, we need 2ptrs

16.Do Pairwise swapping of all nodes in the Linked list

```
   i/p: 2->4->6->9->3->null [Odd nodes]
   o/p:4->2->9using Iterative and Recursive method->6->3->null
   i/p: 2->4->6->9->3->7->null [Even nodes]
   o/p:4->2->9->6->3->7->null
```

17.How to find middle node in Linked list[even and odd number of nodes] {N/2+1}. Implement using 2methods[with 2 ptrs,with only 1ptr]
Ans: 2ptrs:

```
fast_ptr=fast_ptr->next->next;
slow_ptr=slow_ptr->next;
```

1ptr:

```
if(count %2 ==0)
ptr = ptr->next;
```

18.How to find whether the linked list is circular or having loop?If loop is present,then count the number of nodes/length of the loop & how to remove it?
19.How to find whether the linked lists are intersecting with less time complexity
[should not compare every node of one linked list with the other, since it leads to high time complexity]?
20.Find the Intersection/MERGE point of 2 linked lists

case 1: [Y shape Intersection]

```
 L1 : 2->4->7->6->3->null
 L2 : 1->7->6->3->null
```

case 2: [X shape Intersection.This is wrong,Node cannot store 2 addresses]

```
 L1 : 2->4->7->6->3->null
 L2 : 1->7->5->9->null
```

21.How to say that the 2 linked lists are non-intersecting?
22.How to say that the linked list is a palindrome
23.How to say that the linked list is a Anagram.
24.Separate the linked list into 2 separate linked lists i.e., 1st linked list should have even nodes , 2nd linked list should have odd nodes
25.Swap nodes in the linked list without swapping the data [you should swap only links]
26.Given 2 no's 6789, 345

```
      6789   //9->7->6->null
      +345  //5->4->3->null
------------
      7134 //4->3->1->7->null
-------------



   765  //5->6->7
   321  //1->2->3
---------
  1086 //6->0->1
---------
```

Now use 2 separate linked lists and perform addition to get the expected result
The linked list nodes representation/insertion WILL BE IN REVERSE of that number representation
Due to which the result of linked list representation is also in Reverse
27. Add 2 linked lists

```
         9>7->6
         5->6->3
 -----------------------
         4->5->1->7
------------------------
```

NOTE:HEAD TO HEAD addition(From LEFT TO RIGHT addition) is possible so be careful

28.Insertion/Deletion[Does not need extra ptr] of nodes at any position(begin,middle,end) in Double Linked List
29.Reverse Double linked list using Iterative and Recursive method

30.Implement Singly & Doubly Circular linked list using class
31.Insertion/Deletion of nodes at any position(begin,middle,end) in Singly circular linked list and Doubly circular linked list
32.How to Print and count the number of nodes in circular linked list
33.Convert single linked list to circular linked list
34.Split the singly circular linked list into 2halves[should be circular only] till the middle node when number of nodes is even and odd

35.Implement Linear search and Binary search
36.What is Time complexity?Types[Ω θ O]?Which is the most used time complexity
Ans:
Types:
Best   -Omega **Ω**
Average -Theta **θ**
Worst   -Big o **O**

Bubble sort **O(n2)**
Selection sort **O(n2)**
Insertion sort **O(n2)**
Quick sort **O(n2)**
Merge sort **O(nlogn)**
Heap sort **O(nlogn)**

37.Name the Sorting technique that does not use swapping method? Insertion sort,Merge sort

38.Implement Bubble sort,Quick sort,Merge sort,Selection sort,Insertion sort,Heap sort and also mention their time complexity?

39.Insertion/Deletion of nodes at any position(begin,middle,end) in Binary Search Tree[BST]
40.What are the traversals in BFS[Breadth First Search] and DFS[Depth First Search].Explain Each
Ans:
BFS types:
Preorder -root,left,right
Inorder -left,root,right
Postorder -left,right,root

DFS:
Level order traversal

41.How to say that one BST is exactly the mirror image of the other BST?
42.Write the code for finding whether the given Tree is BST or not?
43.How to find the Height of BST

1.How to set a particular bit in a given number?
Ans:num | = (1<<pos); Here pos range is [0 to 7]

2.How to clear a particular bit in a given number?
Ans:num & = ~(1<<pos);

3.How to toggle a particular bit in a given number?
Ans:num ^= (1<<pos);

4.Get the count of no. of bits set in a given number?
Ans:

```
for(int i =0;i<8;i++) [0 to 7 for 8bit machine] //traverse through all
the bits
     {
          if( num&(1<<i) !=0) //It Means some value you get other than 0
             setcount++;
         else if( num&(1<<i) ==0)
             clearcount++;
      }
    cout<<setcount;    //no. of bits set
   cout<<clearcount;  //no. of bits clear
```

(or)

This below method takes less time complexity compared to above

```
        int count =0;
             while(num)
              {
                 num &=(num-1); // unset/clear each bit(1) to 0,from the
right most side
                 count++;
              }
      cout<<count;
```

5.How to do nibble swap in a given number?
short -2bytes
word-4bytes
nibble-4bits

i/p: 0000 0010
o/p: 0010 0000

num=num<<4 | num>> 4;

5.How to reverse the given binary number

6.How to do bitwise swap?

7.Do swapping of 2 no's using bitwise operations

Ans:

```
  int x=10,y=20;
x=x^y;
y=x^y;
x=x^y;
cout<<x<<y; //x=20,y=10
```

Memory Leak:
1.If allocated memory is not freed
2.Allocating memory using malloc() again to the same pointer
3.freeing the same memory again of same pointer

Instead of using malloc() again,use realloc()

```
/*===================================
   Function ptr and Callback functions
   =====================================*/
/*
    int get(int x)
    {
    return x;
    }

    int callback(int(*fptr)(int x),int y)
    {
    return(fptr(3)+y);
    }

    int main()
    {
    int(*fptr)(int)=get; //int(*fptr)(int)=&get;
    cout<<fptr(1)<<endl; //cout<<(*fptr)(1)<<endl;
    cout<<callback(get,2)<<endl; //Call back function,function passed as
argument in other function
    }
  */

/*==============================
   Array of function ptrs
   ===============================*/
/*
    int get1(int x)
    {
    return x;
    }

    int get2(int y)
    {
    return y;
    }

    int get3(int z)
    {
    return z;
    }

    int main()
    {
    int(*fptr[])(int)={get1,get2,get3};
    cout<<fptr[0](5)<<endl;
    cout<<fptr[1](6)<<endl;
    cout<<fptr[2](7)<<endl;
    }
  */

/*========================================
   Array of class objects
   ========================================*/
/*
    class Base
    {
    int x;
    static int y;
    public:
    Base()
    {
    x=1;
    y=2;//We can intialize static member without scope resolution operator
    }
    Base(int a,int b)
    {
    x=a;
    y=b;
    }

    void display()
    {
    cout<<x<<endl;
    cout<<Base::y<<endl;
    }

    };

    int Base::y=10;//definition of static variable is must

    int main()
    {
    Base arrayobj[3];//We can call only default constructor using Array
of class objects
//arrobj[5]={{7,8}}; //Calling parameterised constructor using Array of
class objects,C++11 feature
arrobj[0].display();
}
  */
```

```
/*====================================
   Array of ptrs
   =====================================*/
/*
    int main()
    {
    int arr[5]={2,4,6,8,9};
    int (*ptr)[5]= &arr; //&arr,here '&' should be used definitely,or
else thorws Error[casting of int to int*]
    cout<<(*ptr)[4]<<endl;

    for(int i=0;i<5;i++)
    {
    cout<<(*ptr)[i]<<",";
    }
    }
  */

/*=====================================
   ptr to Array
   ======================================*/

int main()
{
     /*
        int arr[5]={2,4,6,8,9};
        int *ptr =arr;
        cout<<*ptr<<endl;

        for(int i=0;i<5;i++)
        {
        cout<<*ptr<<",";
        ptr++;
        }
      */

     char x[]="GLOBAL";
     char *ptr=x;

     while(*ptr!= '\0') //while(*ptr)
     {
         cout<<*ptr<<",";
         ptr++;
     }
}
```

Arrays: [Insertion/Deletion/Reverse]
1.Count the repeated element and delete it from the array
eg: x[]={2,4,5,6,2,2,2,2,7,8,2,9,2};
 o/p: {4,5,6,7,8,9};
2.Insert an element in the given array
3.Reverse the given array

NOTE:
char x[]="Global"; //string as null character internally
    (or)
char x[]={'G','l','o','b','a','l','\0'}; //Here we have to write '\0' EXPLICITLY

NOTE:Always use NULL macro and not null

Strings:
Take a string and implement String class for cons,copy cons,assignment operator
Implement your own string library functions like
strcpy()
strcat()
strcmp()
strrev()
strlen()
strstr()
strchr()

1.Convert Upper case to Lower case and vice-versa in the given string
2.Get the substring from the string [strstr(),strchr]
3.Remove all vowels from the string
4.Remove all spaces in the string
5.Put spaces between words starting with capital letters
Input : BruceWayneIsBatman
Output : bruce wayne is batman
6.Prove that the given string is a palindrome
7.Find the lengthiest Palindrome from the given string
8.a->bc->d->cba->null,Prove that this is a palindrome
9.a->b->c->d->c->b->a->null,Prove that this is a palindrome
9.Find the number of unique/distinct/non-repeated words from the given string/paragraph
10.Find the count of repeated words in the string and remove these repeated words
11.Find the first repeated word in a string
12.I/p string is char x[] ="AAAABBBCCDEE"; O/P is "A4B3C2DE2"
13.Remove the repeated characters in the sting, i/p is char x[]="Raajjbccccd"; O/p is Rajbcd
14.Split numeric,alphabetic and special symbols from the string
i/p is "geeks01for02geeks03!!!";

```
    o/p is
      geeksforgeeks
      010203
      !!!
```

15.Extract words from the given string
Input : geeks for geeks
Output :

```
       geeks
       for
```

geeks
16.Get the count of Upper/Lower/Special case characters and numeric values
Input : #GeeKs01fOr@gEEks07\
Output :
Upper case letters : 5
Lower case letters : 8
Numbers : 4
Special Characters : 2
17.Reverse the vowels in the given string
18.Program to count the occurrence of a given character in a string

How to find size of any variable without using sizeof() operator?
Ans:

```
#define SIZE(x) (char*)(&x+1) -(char*)(&x) //Type casting any type to char*

int main()
{
long m; //long,double size is 8bytes
cout<<SIZE(*y)<<endl;
char *y;//y is address,size is 4bytes,*y is dereferenced variable of char type,size is 1byte
cout<<SIZE(*y)<<endl;
}
```

===

```
 int *func()
{
static int k =10;//If we do not use static,it throws Compilation Error n not Segmentation fault
return &k;
}

int main()
{
int *x = func();
cout<<*x<<endl;
x++;
}
```

===

strlen