Stack(Linked-List)

```
#include<iostream>
#include<conio.h>
using namespace std;
int item;
struct stack
{
   int data;
   stack *link;
}*ptr,*my_TOP=NULL;
void push()
{
   cout<<"enter the item to be inserted"<<endl;
   cin>>item;
   if(my_TOP==NULL)
   {
   ptr=new stack;
   ptr->data=item;
   ptr->link=NULL;
   my_TOP=ptr;
   }
   else
   {
      ptr=new stack;
      ptr->data=item;
      ptr->link=my_TOP;
      my_TOP=ptr;
   }
}
void pop()
{
   ptr=my_TOP;
   my_TOP=ptr->link;
   delete ptr;
}
void display()
{
   ptr=my_TOP;
   while(ptr!=NULL)
   {
      cout<<ptr->data<<endl;
      ptr=ptr->link;
   }
}
int main()
{
   int option;
   while(option!=4)
```

```
   {
   cout<<"enter 1. to push \nenter 2. to display \nenter 3. to pop"<<endl;
   cin>>option;
   switch(option)
   {
      case 1: push();
      break;
      case 2: display();
      break;
      case 3: pop();
      break;
      default: cout<<"wrongly entered"<<endl;
   }
   }
   getch();
   return 0;
}
```

2. Stack(Array)

Stacks are the data structures that follow the concept of LIFO (Last In First Out). Here are the algorithms for the push() and pop() operations in the Stack. Stack can be accessed only from Top .

1. Push():- To insert an element in the stack.
2. Pop():- To delete an element from the stack.

```
Push(stack , top , max , item)
  1. If top=max
  Print overflow and return      // Stack is full
  2. Set top:=top+1                  // increase top by 1
  3. stack[top]:=item                // insert item at the top
  4. Exit

Pop(stack , top , item)
  1. If top=0
   Print underflow and return   // stack is empty
  2. Set item:=stack[top]            // assign top element to item
  3. top:=top-1                      // decrease top by 1
  4. Exit                            // exit
```

PROGRAM

```
#include<iostream>
#include<conio.h>
using namespace std;
int item;
int size;
int stack[15];
int TOP=-1;
int c=0;
```

```
void push()
{

   cout<<"enter the item to be inserted in the stack"<<endl;
   cin>>item;
   if(TOP!=size)
   {
      TOP=TOP+1;
      stack[TOP]=item;
   }
}
void pop()
{
   item=stack[TOP];
   cout<<"ITEM popped is"<<item<<endl;
   TOP=TOP-1;
}
void display()
{
   for(int i=0;i<=TOP;i++)
   {
   cout<<stack[i]<<"  ";
   }
}
int main()
{
   int option;
   cout<<"enter the size of the stack"<<endl;
   cin>>size;
   while(option!=4)
   {
   cout<<"enter 1. to push \n 2. to display \n 3. to pop \n 4. to exit"<<endl;
   cin>>option;
   switch(option)
   {
      case 1: push();
      break;
      case 2: display();
      break;
      case 3: pop();
      break;
      default: cout<<"wrongly entered"<<endl;
   }
   }
   getch();
   return 0;
}
```