Stack

Algorithm for PUSH operation on Stack

Consider PUSH (Stack [maxsize], Item) where maxsize is the maximum size of the stack and Item is the element to be inserted into stack

```
Step 1: Initialize the value of top to -1(minus one) i e top= -1 because stack is initially empty.
Step 2: Repeat Steps 3 to 5 until top < maxsize-1 ( i. e until stack is not full)
Step 3: Read the element to be inserted, Item.
Step 4: Increment the value of top by 1( i.e top=top+1.)
Step 5: Store the item at the top of the stack (i.e Stack[top]=Item).
Step 6: Display overflow of stack
```

Algorithm for POP operation on Stack

Consider POP(Stack[maxsize], Item) where maxsize is the maximum size of the stack and Item is the element to be removed or deleted.

```
Step 1: Repeat Step 2 to 4 until value of top is greater than or equal to zero.
Step 2: Select the top element from the stack for deletion (i.e item=Stack[top])
Step 3: The value of top is decremented by one (i.e top=top-1).
Step 4: Print the deleted element, Item.
Step 5: Display stack underflow.
```

Q.n. Write a menu driven program to implement Stack using array in C.

```
#include<stdio.h>

#include<conio.h>

#include<stdlib.h>

#define maxsize 10 //the maximum size of stack

void push();

int pop();

void traverse();

int stack[maxsize];

int top= -1;

void main()
```

```
{
int choice;
char ch;
do
{
printf("1. Push\n");
printf("2. Pop\n");
printf("3. Traverse\n");
printf("enter your choice\n");
scanf("%d",&choice);
switch(choice)
{
case 1: push();
break;
case 2: printf("the deleted element is:%d",pop());
break;
case 3: traverse();
break;
default: printf("invalid choice\n");
}
printf("do you wish to continue(Y/N)");
scanf("%c",&ch);
}while(ch= ='y' || ch= ='Y');
}
void push()
{
int item;
if(top= = maxsize-1)
{
printf("stack is full");
exit(0);
}
else
{
printf("enter the elements to be inserted");
scanf("%d",&item);
top=top+1;
stack[top]=item;
}
}
int pop()
{
int item;
if(top= = -1)
{
printf("stack is empty");
exit(0);
}
else
{
item=stack[top];
top=top-1;
}
return  item;
}
void traverse()
```

```
{
int i;
if(top= = -1)
{
printf("stack is empty");
exit(0);
}
else
{
for(i=top;i>=0;i--)
{
printf("the traverse element is:%d",stack[i]);
}
}
}
```

Q.n Write a menu driven program to implement Stack using pointer in C.

```
#include<stdio.h>
#include<conio.h>
struct stack
{
int num;
struct stack *next;
}*top=NULL;
typedef struct stack st;
void push();
int pop();
void main()
{
```

```
char ch;
int choice, item;
do
{
printf("1. Push\n");
printf("2. Pop\n");
printf("3. Display\n");
printf("enter your choice\n");
scanf("%d",&choice);
switch(choice)
{
case 1: push();
break;
case 2: printf("the deleted element is:%d",pop());
break;
case 3: display();
break;
default: printf("wrong choice");
}while(ch= ='y' || ch= ='Y');
}
}
void push()
{
st *p;
node= (st *) malloc (sizeof(st));
printf("enter the number\n");
scanf("%d",&p→num);
p→next=top;
```

```
top=node;
}
int pop()
{
st *p;
p=start;
if(top= =NULL)
{
printf("Stack is already empty\n");
exit(0);
}
else
{
top=top→next;
free(p);
}
return  (p→num);
}
void display()
{
st  *p;
temp=top;
while(p→next !=NULL)
{
printf("the number is:%d", p→num);
p=p→next;
}
printf("the number is:%d",p→num);
}
```