stack array

1.(a)

Aim:

To implement stack operations using array.

Algorithm:

* start
* define a array stack of size max =5
* Initialize top = -1
* Display a menu listing stack operations.
* Accept choice
* IF choice = 1 then

IF top < max -1

increment top

Store element at current Position of top

Else

Print stack overflow

Else IF choice = 2 then

IF top < 0 then

Print stack underflow

Else

Display current top
Element Decrement top
Else IF choice = 3 then
Display Stack elements starting from top
*stop.

Program:

```
/* Stack operations using arrays */
# include <studio.h>
# include
<conio.h> # define
max 5
Static int stack [max];
int top: = -1;
void Push (int x)
{
stack [++ top] =x;
}
int POP ()
{
return (stack [top--1]);
}
void view ()
```

952319104301 k.ARULSELVI

```
{
int i;
IF (top < 0)
PrintF ("\n stack empty \n");
else
{
PrintF ("\n top -->");
For (i = top; i > 0; i--);
{
PrintF ("\n stack empty \n");
else
{
PrintF ("\n top -->");
for (i = top; i >= 0; i--);
{
printf (" & 4d", stack [i]);
 }
printf ("\n");
 }
}
main ()
{
```

```
int ch=0 , val;
ClrScr ();
while (ch != 4)
{
Printf ("\n Stack OPERATION \n");
Printf (" 1. PUSH ");
Printf (" 2. POP ");
Printf (" 3. VIEW");
Printf ("4. QUIT");
Printf ("Enter choice : ");
 Scanf (" &d", &ch);
 Switch (ch);
 {
case 1:
 if (top < max -1)
 {
printf ("\n Enter stack element : ");
Scanf (" & d", &val;
 Push (val);
  }
 else
printf ("\n stack overflow \n");
```

```
 break;
case 2:
 if (top<0)
Printf ("\n stack underflow \n");
else
{
val =pop ();
Printf ("\n Popped elements is %d \n", val);
 }
break;
case 3:
  view ();
  break;
case 4:
   exit (0);
  default:
   printf ("\n Invalid choice \n");
        }
     }
   }
```

952319104301 k.ARULSELVI

output

```
STACK OPERATION
1. PUSH 2. POP 3. VIEW 4. QUIT
Enter choice: 1
Enter stack element:
12 STACK operation
1. PUSH 2. POP 3. VIEW 4. QUIT
Enter choice: 1
Enter stack element:
34 STACK operation
1. PUSH 2. POP 3. VIEW 4. QUIT
Enter choice: 1
Enter stack element:
45 STACK operation
1. PUSH 2. POP 3. VIEW 4. QUIT
Enter choice: 3
TOP -->45, 34, 23
```

12 STACK OPERATION

1. PUSH 2. POP 3. VIEW 4. QUIT

Enter choice : 2

POPPed elements is

45

STACK OPERATION

1. PUSH 2. POP 3. VIEW 4. QUIT

Enter choice : 3

POP Ped elements is

TOP --> 34 , 23

12 STACK

OPERATION

1. PUSH 2. POP 3. VIEW 4. QUIT

Enter choice : 4

Result:

Thus Push and POP operations of a stack was demonstrated using array verified successfully.

1(b)

Queue array

Aim:

To implement queue operations using array

Algorithm:

* Start
* define a array queue of size max = 5
* initialize front = rear = -1
* Display a menu listing queue operations
* Accept choice
* IF choice = 1

Then IF rear < max - 1

Increment rear

Store element at current position of rear

else

print queue Full

Else IF choice = 2

then IF front = -1

then

952319104301 k. ARULSELVI

print Queue empty

else

Display current front element Increment Front

Else IF choice = 3 then

Display queue elements starting From Front to rear.

* stop

Program:

```
/* Queue operation using arrays */
#include <studio.h>
# include < conio.h>
# define max 5
static int queue [max];
int front = -1;
int rear = -1;
void insert (int x)
{
queue [++ rear ]= x;
if (front == -1)·
    Front = 0;
```

```
}
int remove ()
{
int val;
val = queue [front];
if (front == rear && rear == max -1);
front = rear = -1;
else
 front ++;
return (val);
}
 void view ()
{
int i;
if (front == -1)
Printf ("\n Queue Empty \n");
else
 {
Printf ("\n front -->");
For (i= front; i< rear; i++);
Printf ("& 4d", queue [i]);
```

```
Printf (" <-- Rear \n");
    }
  }
 main ()
{
int ch=0, val;
clrscr ();
while (ch != 4);
{
Printf ("\n Queue operation \n");
Printf (" 1. INSERT ");
Printf (" 2. DELETE");
Printf (" 3. VIEW");
printf (" 4. QUIT \n");
 Printf ("Enter choice : ");
 Scanf (" %d", &ch);
 switch (ch)
{
case 1:
 if (rear <max -1)
 {
Printf ("\n Enter elements to be inserted :");
```

```
scanf ("%d", &val);
insert (val);
  }
else
PrintF ("\n Queue Full \n");
break;
Case 2:
 iF (front == -1)
PrintF ("\n Queue empty \n");
else
{
Val = remove ();
PrintF ("\n Element deleted: %d \n", val);
  }
break;
case 3:
   view ();
  break;
Case 4:
   exit (0);
  default:
printf ("\n Invalid choice \n");
          }
```

```
    }
}
```

Output

```
Queue operation
1. INSERT 2. DELETE 3. VIEW 4. QUIT
Enter choice : 1
Enter element to be inserted:
12 Queue operation
1. INSERT 2. DELETE 3. VIEW 4. QUIT
Enter choice : 1
Enter element to be inserted:
23 QUEUE operation
1. INSERT 2. DELETE 3. VIEW 4. QUIT
Enter choice : 1
Enter element to be inserted:
34 QUEUE operation
1. INSERT 2. DELETE 3. VIEW 4. QUIT
Enter choice : 1
Enter element to be inserted:
```

9523191043O1 K. ARULSELVI

45 QUEUE Operation

1. INSERT 2. DELETE 3. VIEW 4. QUIT

Enter Choice : 1

Enter element to be inserted:

56 QUEUE Operation

1. INSERT 2. DELETE 3. VIEW 4. QUIT

Enter choice : 1

1 Queue Full

QUEUE Operation

1. INSERT 2. DELETE 3. VIEW 4. QUIT

Enter choice : 3

Front --> 12, 23, 34, 45, 56 <--- Rear

Result:

Thus insert and delete operations of a queue was demonstrated using array. verified successfully.