Shri Vile Parle Kelavani Mandal's
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING
(Autonomous College Affiliated to the University of Mumbai)
NAAC Accredited with "A" Grade (CGPA : 3.18)

Department of Information Technology

COURSE CODE: DJS22ITL302 **DATE:08/10/2023**

COURSE NAME: Data Structure Laboratory **CLASS: I1-Batch1**

NAME: Anish Ashok Sharma **CLASS: I1- 60003220045**

Experiment No. 1

CO/LO: CO1

Aim: Implement stack and queues using arrays

Theory: A stack is a data structure that follows the Last-In, First-Out (LIFO) principle. This means that the last element added to the stack is the first one to be removed.
A queue is a data structure that follows the First-In, First-Out (FIFO) principle. This means that the first element added to the queue is the first one to be removed.

Program:

Queues:

```
#include <stdio.h>


int queue[5];

int n=5;

int rear,front;

rear=-1;

front=-1;

void insert(int x){

   if(rear==n-1)

      printf("Overflow\n");
```

```
    else if(front==-1 && rear==-1){
        front=rear=0;
        queue[rear]=x;
    }
    else{
        rear++;
        queue[rear]=x;
    }
}
void delete(){
    if(rear==-1 && front==-1){
        printf("Underflow\n");
    }
    else if(front==rear){
        printf("\nDelete element:%d",queue[front]);
        rear=-1;
        front=-1;
    }
    else{
        printf("\nDelete element:%d\n",queue[front++]);;
    }
```

```
}
void display(){
    int i;
    printf("\nPrinting queue:\n");
    for(i=front;i<=rear;i++){
        printf("%d ",queue[i]);
    }
    printf("\n");
}
int main()
{
    int choice=1;
    while(choice){
        printf("1-insert\n2-delete\n3-display\n0-exit\n");
        int check;
        scanf("%d",&check);
        if(check==0)
            break;
        else if(check==1){
            printf("\nEnter element:\n");
            int x;
```

```
            scanf("%d",&x);
            insert(x);
        }
        else if(check==2){
            delete();
        }
        else if(check==3){
            display();
        }
    }
    return 0;
}
```

Output screenshots:

```
1-insert
2-delete
3-display
0-exit
1

Enter element:
56
1-insert
2-delete
3-display
0-exit
1

Enter element:
78
1-insert
2-delete
3-display
0-exit
1

Enter element:
90
1-insert
2-delete
3-display
0-exit
45
1-insert
2-delete
3-display
0-exit
1

Enter element:
56
```

```
1-insert
2-delete
3-display
0-exit
2

Delete element:56
1-insert
2-delete
3-display
0-exit
2

Delete element:78
1-insert
2-delete
3-display
0-exit
3

Printing queue:
90 56
1-insert
2-delete
3-display
0-exit

Printing queue:
90 56
1-insert
2-delete
3-display
0-exit
```

Stack:

```
#include <stdio.h>
#include <conio.h>
int arr[4];
int n = 4;
int top = -1;
void push(int x)
{
   if (top == n - 1)
   {
      printf("Overflow\n");
   }
   else
   {
      top++;
      arr[top] = x;
   }
}
void pop()
{
   if (top == -1)
   {
      printf("Underflow\n");
   }
   else
```

```
    {
        printf("\nPop element:%d", arr[top]);
        top--;
    }
}
void peek()
{
    printf("\nPeak element:%d", arr[top]);
}
void main()
{
    int choice = 1;
    printf("Implementation of stack\n");
    // Push
    while (choice)
    {
        printf("\nEnter 1-push\n2-pop\n3-peek,\n0-exit\n");
        scanf("%d", &choice);
        if (choice == 0)
            break;
        else if (choice == 1)
        {
            // int check;
            // printf("\nDo want to push a element in stack?\n1 for yes 0 for no\n");
            // scanf("%d",&check);
            // if(check==0){
            //     break;
```

```
        // }
        printf("\nEnter a element: ");
        int x;
        scanf("%d", &x);
        push(x);
    }
    else if (choice == 2)
    {
        pop();
    }
    else if (choice == 3)
    {
        peek();
    }
    else
    {
        printf("\nEnter valid number");
    }
}
// pop
//  while(1){
//      printf("\nDo you want to pop the element?\n1 for yes no for 0\n");
//      int check;
//      scanf("\n%d",&check);
//      if(check==0)
//          break;
//      pop();
```

```
  //  }
  // peak
  //  printf("\nDo you want to check peak element?\n1for yes no for 0");
  //  int check;
  //  scanf("%d",&check);
  //  if(check==1){
  //      peek();
  //  }else{
  //      printf("\nThank you");
  //  }
}
```

```
Implementation of stack

Enter 1-push
2-pop
3-peek,
0-exit
1

Enter a element: 45

Enter 1-push
2-pop
3-peek,
0-exit
1

Enter a element: 78

Enter 1-push
2-pop
3-peek,
0-exit
1

Enter a element: 90

Enter 1-push
2-pop
3-peek,
0-exit
2

Pop element:90
Enter 1-push
2-pop
3-peek,
0-exit
2

Pop element:78
```

Conclusion:

In conclusion, you can implement a stack and a queue using arrays by employing specific operations to mimic their respective Last-In, First-Out (LIFO) and First-In, First-Out (FIFO) behaviors.

REFERENCES:

No references