Assignment-6

UCS540 (Data Structures and Algorithms)

Submitted by, Harpartap Singh, 102104119

Group- 3EE3

1. **Write a menu driven program with 4 options (Insert, Delete, Display, and Exit) to demonstrate the working of Queues using arrays.**

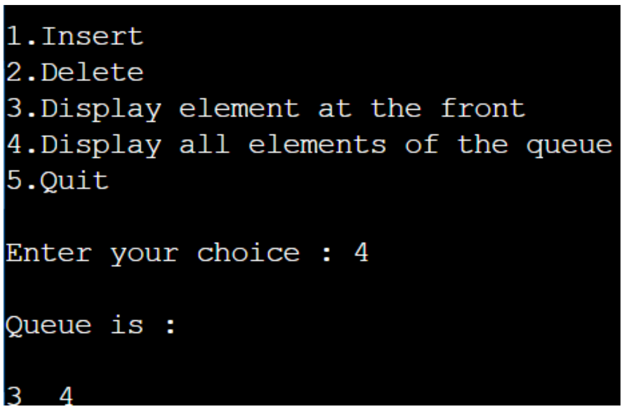
Code:

```
#include<stdio.h>
#include<stdlib.h>
#define MAX 10
int queue_arr[MAX];
int rear=-1;
int front=-1;
void insert(int item);
int del();
int peek();
void display();
int isFull();
int isEmpty();
int main()
{int choice,item;
while(1){printf("\n1.Insert\n");
printf("2.Delete\n");
printf("3.Display element at the front\n");
printf("4.Display all elements of the queue\n");
printf("5.Quit\n");
printf("\nEnter your choice : ");
scanf("%d",&choice);
switch(choice) {case 1:
```

```
printf("\nInput the element for adding in queue : ");
scanf("%d",&item);
insert(item);
break;
case 2: item=del();
printf("\nDeleted element is  %d\n",item);
break;
case 3:printf("\nElement at the front is %d\n",peek());
break;
case 4:display();
break;
case 5:exit(1);
default:printf("\nWrong choice\n");}}
return 0;}
void insert(int item){if( isFull() )
{printf("\nQueue Overflow\n");
return;}
if( front == -1 )
front=0;
rear=rear+1;
queue_arr[rear]=item ;}
int del(){ int item;
if( isEmpty()){printf("\nQueue Underflow\n");
exit(1);}
item=queue_arr[front];
front=front+1;
return item;}
int peek(){ if( isEmpty() ){printf("\nQueue Underflow\n");
exit(1);}
return queue_arr[front];}
int isEmpty(){ if( front==-1 || front==rear+1 )
```

```
return 1;

else return 0;}

int isFull(){if( rear==MAX-1 )

return 1;

else return 0;}

void display(){int i;

if ( isEmpty() )

{printf("\nQueue is empty\n");

return;}

printf("\nQueue is :\n\n");

for(i=front;i<=rear;i++)

printf("%d  ",queue_arr[i]);

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

Output:



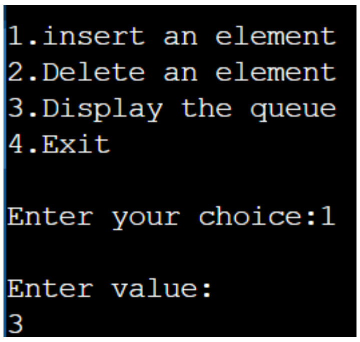
```
1.Insert
2.Delete
3.Display element at the front
4.Display all elements of the queue
5.Quit

Enter your choice : 4

Queue is :

3  4
```

2. **Write a menu driven program with 4 options (Insert, Delete, Display, and Exit) to demonstrate the working of Queues using linked-list.**

Code:

```
#include<stdio.h>

#include<stdlib.h>

struct node{int data;

struct node *next;};

struct node *front;

struct node *rear;

void insert();
void dequeue();
void display();
int main ()
{int choice;
while(choice != 4)
{printf("\n1.insert an element\n2.Delete an element\n3.Display the queue\n4.Exit\n");
printf("\nEnter your choice:");
scanf("%d",& choice);
switch(choice)
{case 1: insert();
break;
case 2:dequeue();
break;
case 3:display();
break;
case 4:exit(0);
break;
default:printf("\nEnter valid choice??\n");}}
return 0;}
void insert(){struct node *ptr;
int item;
ptr = (struct node *) malloc (sizeof(struct node));
if(ptr == NULL)
{printf("\nOVERFLOW\n");
return;}
else{printf("\nEnter value:\n");
scanf("%d",&item);
ptr -> data = item;
if(front == NULL){front = ptr;
rear = ptr;
front -> next = NULL;
rear -> next = NULL;}

else{rear -> next = ptr;

rear = ptr;

rear->next = NULL;}}}

void dequeue (){struct node *ptr;

if(front == NULL){printf("\nUNDERFLOW\n");

return;}

else {ptr = front;

front = front -> next;

free(ptr);}}

void display(){struct node *ptr;

ptr = front;

if(front == NULL){printf("\nEmpty queue\n");}

else{printf("\nprinting values .....\n");

while(ptr != NULL){printf("\n%d\n",ptr -> data);

ptr = ptr -> next;}}}
```

Output:



```
1.insert an element
2.Delete an element
3.Display the queue
4.Exit

Enter your choice:1

Enter value:
3
```

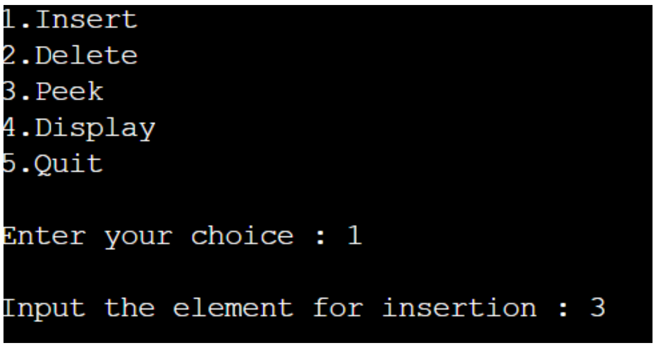
3. **Write a menu driven program with 4 options (Insert, Delete, Display, and Exit) to demonstrate the working of Circular Queues (arrays.)**

Code:

```
#include<stdio.h>

#include<stdlib.h>

#define MAX 10

int cqueue_arr[MAX];

int front=-1;
int rear=-1;
void display( );
void insert(int item);
int del();
int peek();
int isEmpty();
int isFull();
int main(){int choice,item;
while(1){printf("\n1.Insert\n");
printf("2.Delete\n");
printf("3.Peek\n");
printf("4.Display\n");
printf("5.Quit\n");
printf("\nEnter your choice : ");
scanf("%d",&choice);
switch(choice){case 1:printf("\nInput the element for insertion : ");
scanf("%d",&item);
insert(item);
break;
case 2:printf("\nElement deleted is : %d\n",del());
break;
case 3:printf("\nElement at the front is  : %d\n",peek());
break;
case 4:display();
break;
case 5:exit(1);
default:
printf("\nWrong choice\n");}}
return 0;}
void insert(int item){if(isFull())
{printf("\nQueue Overflow\n");
return;}
if(front == -1 )
front=0;
if(rear==MAX-1)
rear=0;
else rear=rear+1;
cqueue_arr[rear]=item;}
int del(){int item;
if(isEmpty()){printf("\nQueue Underflow\n");
exit(1);}
item=cqueue_arr[front];
if(front==rear)
{front=-1;
rear=-1;}
else if(front==MAX-1)
front=0;
else
front=front+1;
return item;}
int isEmpty(){if(front==-1)
return 1;
else
return 0;}
int isFull(){if((front==0 && rear==MAX-1) || (front==rear+1))
return 1;
else
return 0;}
int peek(){if( isEmpty()){printf("\nQueue Underflow\n");
exit(1);}
return cqueue_arr[front];}
void display(){int i;
```

```
if(isEmpty())
{printf("\nQueue is empty\n");
return;}
printf("\nQueue elements :\n");
i=front;
if( front<=rear )
{while(i<=rear)
printf("%d ",cqueue_arr[i++]);}
else{while(i<=MAX-1)
printf("%d ",cqueue_arr[i++]);
i=0;
while(i<=rear)
printf("%d ",cqueue_arr[i++]);}
printf("\n");}
```

Output:



```
1.Insert
2.Delete
3.Peek
4.Display
5.Quit

Enter your choice : 1

Input the element for insertion : 3
```