```
/*WAP for array implementation of Linear Queue*/
#include<iostream>
using namespace std;
#define SIZE 5
class Queue
{
   int A[SIZE];
   int Front;
   int Rear;
public:
   Queue()
   {
      Front=-1;
      Rear=-1;
   }
   bool isempty()
   {
      if(Front== -1 && Rear == -1)
         return true;
      else
         return false;
   }
   bool isfull()
   {
      if(Rear == SIZE-1)
         return true;
      else
         return false;
   }
   void enqueue(int value)
   {
      if(isfull())
      {
         cout<<"queue is full"<<endl;
      }
      else
      {
        if(Front==-1)//first element is inserted
        {
           Front=0;
        }
        Rear++;
        A[Rear]= value;
        //inserting value
     }
  }
  void dequeue()
  {
     if(isempty())
     {
        cout<<"Queue is empty\n";
     }
     else if(Front==Rear) //only one element)
     {
        Front=Rear=-1;
     }
     else
     {
        Front++;
     }
  }
  void showfront()
  {
     if(isempty())
     {
        cout<<"Queue is empty\n";
     }
     else
     {
        cout<<"element at front is"<<A[Front];
     }
  }
  void showQueue()
  {
      if(isempty())
      {
         cout<<"Queue is empty\n";
      }
      else
      {
         for (int i=Front; i<=Rear; i++)
         {
            cout<<A[i]<<endl;;
         }
      }
   }
};
void menu()
{
   Queue q;
   char choice;
   while (choice!='3')
   {
      cout<<"\n\n\n\n"<<endl;
      cout<<"___________MENU___________________"<<endl;
      cout<<"1. Enqueue an element\n"<<endl;
      cout<<"2. Dequeue an element\n"<<endl;
      cout<<"3. Display all elements\n"<<endl;
      cout<<"4. Display front elements\n"<<endl;
      cout<<"5. Exit\n"<<endl;
      cout<<"enter a choice \t"<<endl;
      cin>>choice;
      if(choice=='1')
      {
         int n;
         cout<<"Enter an element to be enqueued"<<endl;
         cin>>n;
         cout<<"\n\n\n"<<endl;
         q.enqueue(n);
      }
      else if(choice=='2')
```

```
        {
           cout<<"\n\n\n"<<endl;
           q.dequeue();
        }
        else if(choice=='5')
        {
           break;
        }
        else if(choice=='3')
        {
           q.showQueue();
        }
        else if(choice=='4')
        {
           q.showfront();
        }
    }
}
int main()
{
    menu();
}

/*WAP for array implementation of Linear Queue*/
#include<iostream>
using namespace std;
class queue
{
    int FRONT;
    int REAR;
    int MAX;
    int *arr;
public:
    queue()
    {
        cout<<"Enter the size of queue:\t";
        cin>>MAX;
        FRONT=-1;
        REAR=-1;
        arr = new int[MAX];
    }
    bool is_empty()
    {
        if ((FRONT==-1) || (FRONT>REAR))
        {
           return true;
        }
        else
        {
           return false;
        }
    }
    bool is_full()
    {
        if (REAR>=MAX-1)
        {
           return true;
        }
        else
        {
           return false;
        }
    }
    void enqueue(int num)
    {
        if (REAR ==-1 && FRONT==-1)
        {
           FRONT=0;
        }
        if (!is_full())
        {
           REAR+=1;
           arr[REAR]=num;
        }
```

```
        else
        {
           cout<<"****************Overflow****************"<<endl;
        }
    }
    int dequeue()
    {
        if(!is_empty())
        {
           int VALUE=arr[FRONT];
           FRONT+=1;
           return VALUE;
        }
    }
};
int main()
{
    queue q;
    int choice;
    int num;
    do
    {
        cout<<endl<<"Enter your choice:"<<endl;
        cout<<"1)enqueue"<<endl;
        cout<<"2)dequeue"<<endl;
        cout<<"3)quit"<<endl;
        cout<<"Enter your choice:\t";
        cin>>choice;
        switch(choice)
        {
        case 1:
           cout<<"\nEnter number to enqueue:\t";
           cin>>num;
           q.enqueue(num);
           break;
        case 2:
           if (!q.is_empty())
```

```
         {
            cout<<"\n*********Dequeued
number:"<<q.dequeue()<<"*********\n";
         }
         else
         {
            cout<<"***************Underflow***************"<<endl;
         }
         break;
      case 3:
         break;
      }
   }
   while (choice !=3);
}

/*WAP for array implementation of Linear Queue*/
#include <iostream>
#include <cstdlib>
#define  capacity 9
using namespace std;
template <class X>
class Queue
{
private:
   X data[capacity];
   int front1,rear;
   bool IsEmpty()
   {
      if(front1>rear || front1==-1)
         return true;
      else
         return false;
   }
   bool IsFull()
   {
      if(rear==(capacity-1))
```

```
            return true;
        else
            return false;
    }
public:
    Queue():front1(-1),rear(-1) {}
    void enqueue(X var)
    {
        if(IsFull())
            cout<<endl<<"Queue overflow"<<endl;
        else
        {
            if(front1 == -1)
                front1 =0;
            data[++rear]=var;
        }
    }
    void dequeue()
    {
        if(IsEmpty())
            cout<<"\nQueue underflow"<<endl;
        else
            cout<<"\nThe dequeued element is :  "<<data[front1++]<<endl;
    }
    void Front()
    {
        if(IsEmpty())
            cout<<"\nQueue underflow"<<endl;
        else
            cout<<"\nThe front element of queue is : "<<data[front1]<<endl;
    }
};
int main()
{
    Queue <int>Q1;
    int choice;
    while(1)
    {
        cout<<"1.Enqueue\n2.Dequeue\n3.View front element\n4.exit\nEnter
your choice  ";
        cin>>choice;
        switch(choice)
        {
        case 1:
        {
            cout<<"Enter the value:  ";
            cin>>choice;
            Q1.enqueue(choice);
            break;
        }
        case 2:
            Q1.dequeue();
            break;
        case 3:
            Q1.Front();
            break;
        default:
            exit(0);
        }
        cout<<"\n\n";
    }
    return 0;
}
```