// implementation of queue using linked list //

ALGORITHM :

1: Allocate the space for the new node PTR

2 : SET PTR-> DATA=VAL

3 : IF FRONT = NULL SET FRONT = REAR= PTR

SET FRONT -> NEXT = REAR ->NEXT = NULL

ELSE SET REAR-> NEXT=PTR.SET REAR=PTR.

SET REAR-> NEXT = NULL [END OD IF]

4 : END

// program implementation of queue in linked list //

include<stdio.h>

#include<stdlib.h>

Struct node

{#

Int data ;

Struct node\*next;

};

Struct node\*front=NULL;

Struct node\*rear=NULL;

Void enqueue(int value)

{

Struct node\*ptr;

Ptr=(struct node \*)malloc(size of (struct node));

Ptr->data=value;

Ptr->next=NULL;

If((front==NULL)&&(rear==NULL));

{

Front=rear=ptr;

}

Else

{

Rear->next=ptr;

Rear=ptr;

}

Printf(“node is inserted\n\n”);

}

Int dequeue()

{

If(front==NULL)

{

Printf(“\n underflow\n”);

Return -1;

}  
else

{

Struct node\*temp=front;

Int temp\_data=front->data;

Front=front->next;

Free(temp);

Return temp\_data;

}

}

Void display()

Struct node\*temp;

If((front==NULL)&&(rear==NULL))

{

Printf(“\nqueue is empty\n”);

}

Else

{

Printf(“the queue is \n”);

Temp=front;

While(temp)

{

Printf(“%d🡪”,temp ->data);

Temp=temp->next;

}

Printf(“NULL\N\N”);

}

}

Int main()

{

Int choice,value;

Printf(“\n implementation of queue using linked list\n”);

While(choice!=4)

{

Printf(“1.enqueue\n 2.dequeue\n 3.display\n 4.exit\n”);

Printf(“\n enter the choice :”);

Scanf(‘%d”,&choice);

Switch(choice)

{

Case 1 :

Printf(“\n enter the value to insert :”);

Scanf(“%d”,&value);

Enqueue(value);

Break;

Case 2:

Dequeue()

Break:

Case 3:

Display()

Break;

Case 4:

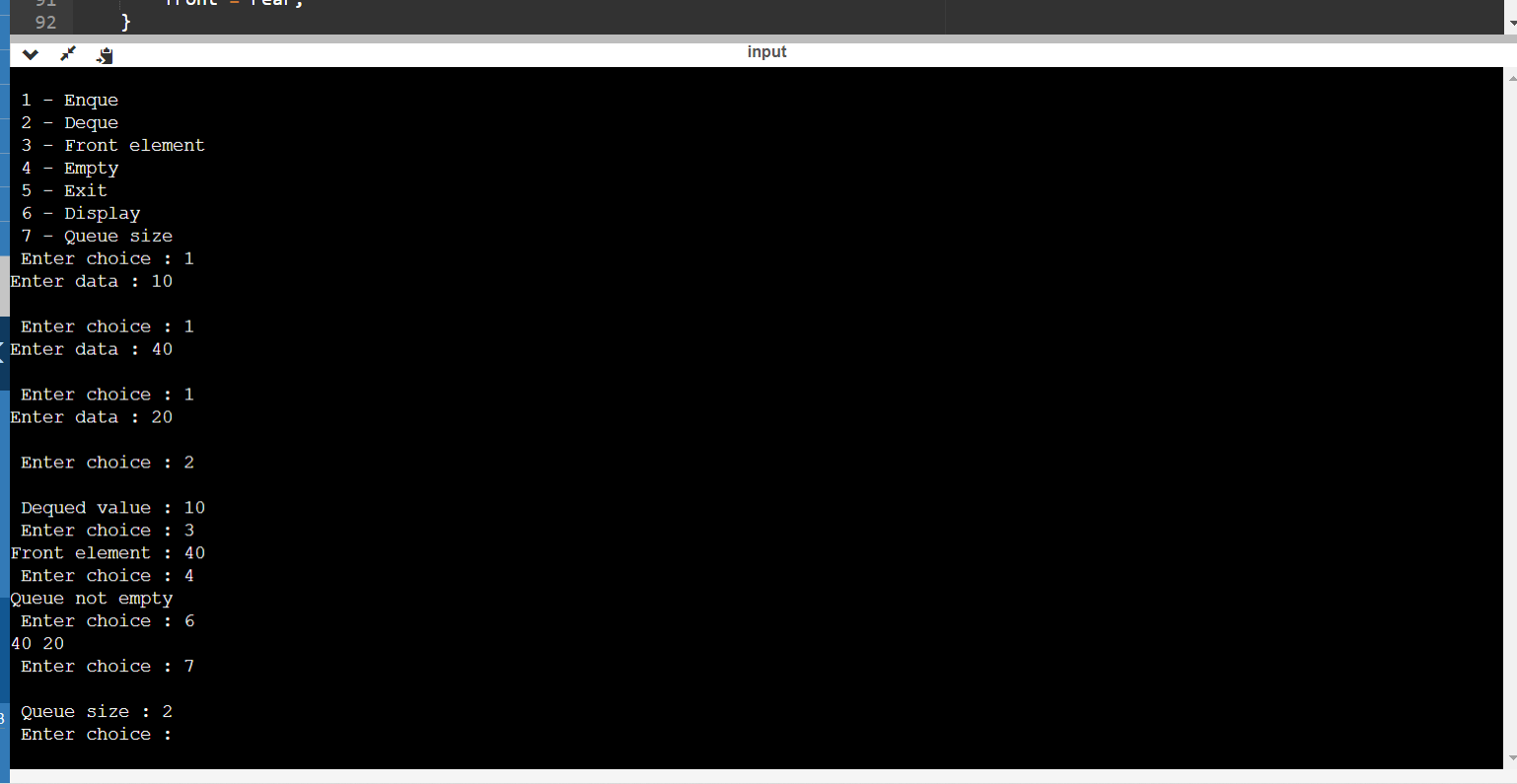
Exit

Return ;

}

}

OUTPUT :



GIT HUB LINK :