**PROGRAM-5**

***Q. Design and implement a given type of queue in C (Ordinary queue/Circular queue) using Array implementation and linked list implementation. Also demonstrate the working with suitable inputs. Display appropriate messages in case of exceptions.***

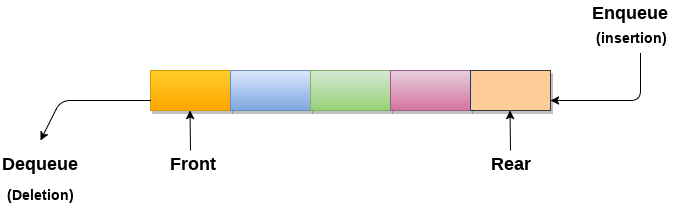
**“Queues Implementation using Arrays”**

* **Theory:**

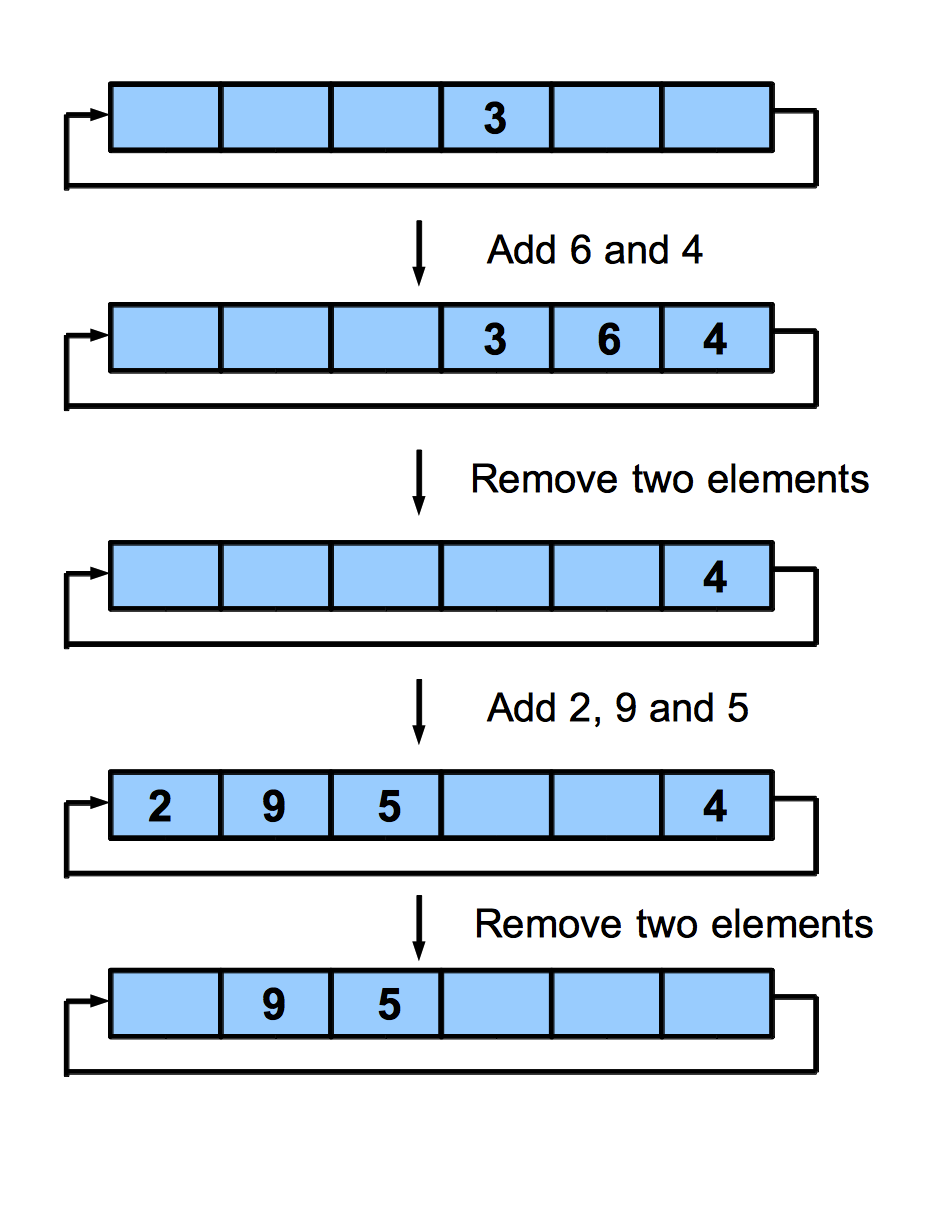
 A queue is a linear data structure can which enables insert operations to be performed at one end called **REAR** and delete operations to be performed at another end called **FRONT**.

Queue follows FIFO (First In First Out list) principle.

Example: people waiting in line for a rail ticket form a queue.



With the use of linear arrays, queue can be readily represented. Every queue has two variables, front and back, which are put into practise. The positions from which insertions and deletions are made in a queue are indicated by the front and rear variables. Front and queue are initially set to -1, which symbolises an empty queue.



//-----------------------------------------------------------------------------

* **Algorithm:**

**Insertion:**

Step 1: IF REAR = MAX - 1

Write OVERFLOW

Go to step

[END OF IF]

Step 2: IF FRONT = -1 and REAR = -1

SET FRONT = REAR = 0

ELSE

SET REAR = REAR + 1

[END OF IF]

Step 3: Set QUEUE[REAR] = NUM

Step 4: EXIT

**Deletion:**

Step 1: IF FRONT = -1 or FRONT > REAR

Write UNDERFLOW

ELSE

SET VAL = QUEUE[FRONT]

SET FRONT = FRONT + 1

[END OF IF]

Step 2: EXIT

//-----------------------------------------------------------------------

* **Code:**

#include<stdio.h>

#include<stdlib.h>

#define size 10

int front=-1;

int rear=-1; // front and rear are set to -1

int queue[size]; // a queue of specific size is defined

void enqueue();

void dequeue();

void display(); //function prototypes

int main()

{

int choice;

while(choice !=4) //while loop for choice

{

printf("---------------Main Menu---------------\n");

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("\n 1. Insert \n 2. Delete \n 3. Display \n 4.Exit\n");

printf("Enter your choice:"); //user is asked to enter hischoice

scanf("%d",&choice); //choice is accepted from the user

switch(choice)

{

case 1: enqueue();

break;

case 2: dequeue();

break;

case 3: display(); //switch case for operations

break;

case 4: exit(0);

break;

default: printf("please enter valid choice");

break;

}

}

}

void enqueue() //insert operation

{

int item;

printf("Enter the item:");

scanf("%d",&item); //user enters the value to be inserted

if(rear==size-1)

{

printf("\n OVERFLOW"); // queue is full

return;

}

if(front==-1 && rear==-1)

{

front=0; //if front and rear are equal to -1 make them as zero.

rear=0;

}

else

{

rear=rear+1; //rear is incremented

}

queue[rear]=item; //item is inserted at the rear end

printf("value inserted\n");

}

//------------------------------------------------------------------

void dequeue() // delete operation

{

int item;

if(front==-1 || front>rear)

{

printf("\n UNDERFLOW"); //queue is empty

return;

}

else

{

item=queue[front]; // item is dequeued

}

if(front==rear)

{

front=-1;

rear=-1; //front and rear =-1

}

else

{

front=front+1; //front is incremented

}

printf("Value deleted is %d\n",item); //value deleted is printed

}

//-------------------------------------------------------

void display() //display function

{

if(front==-1)

{

printf("Queue is empty\n"); // no elements are there to print

}

else

{

int i;

printf("The elements are...\n");

for(i=front;i<=rear;i++) //for loop for printing

{

printf("%d\n",queue[i]); //elements are printed

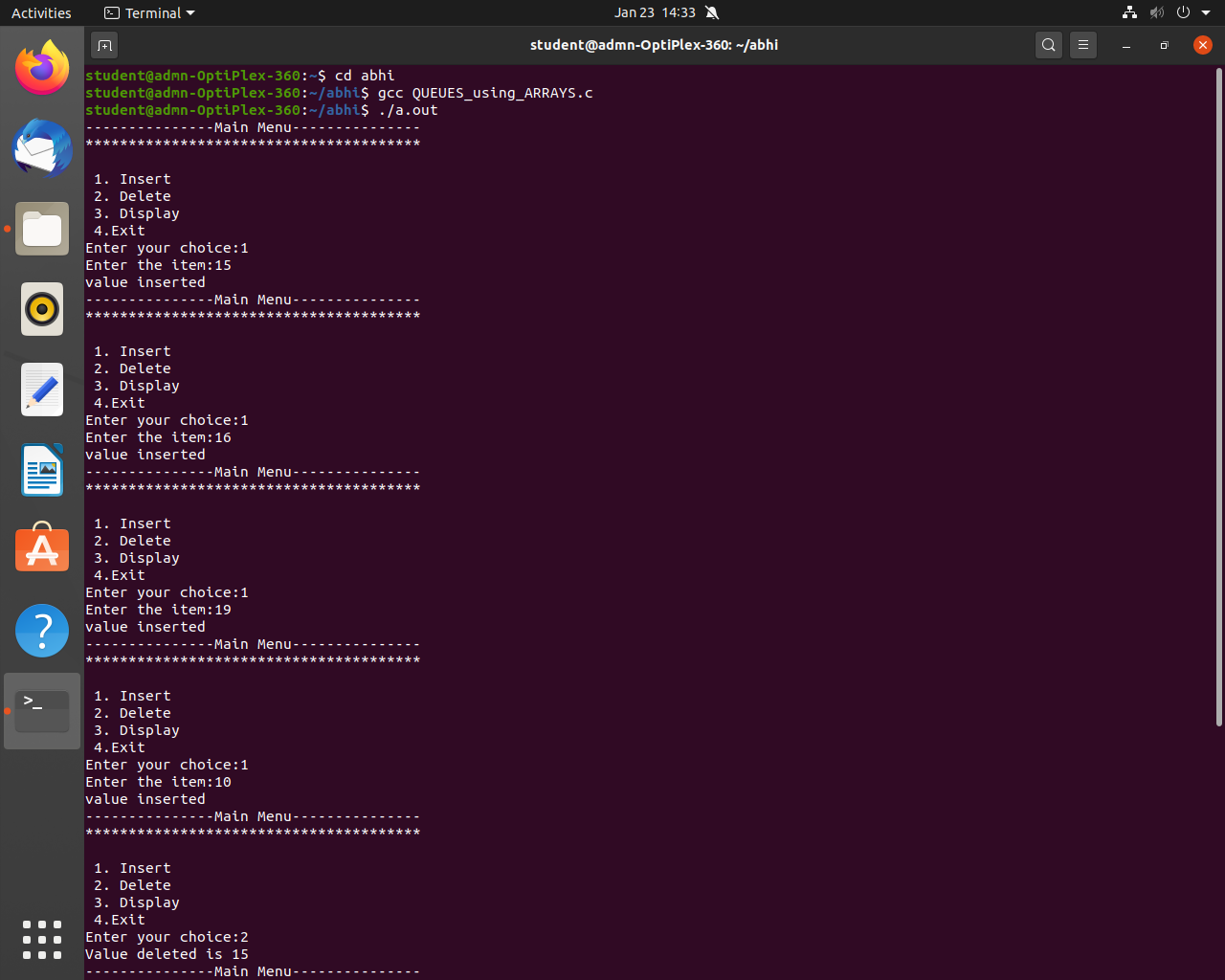
}

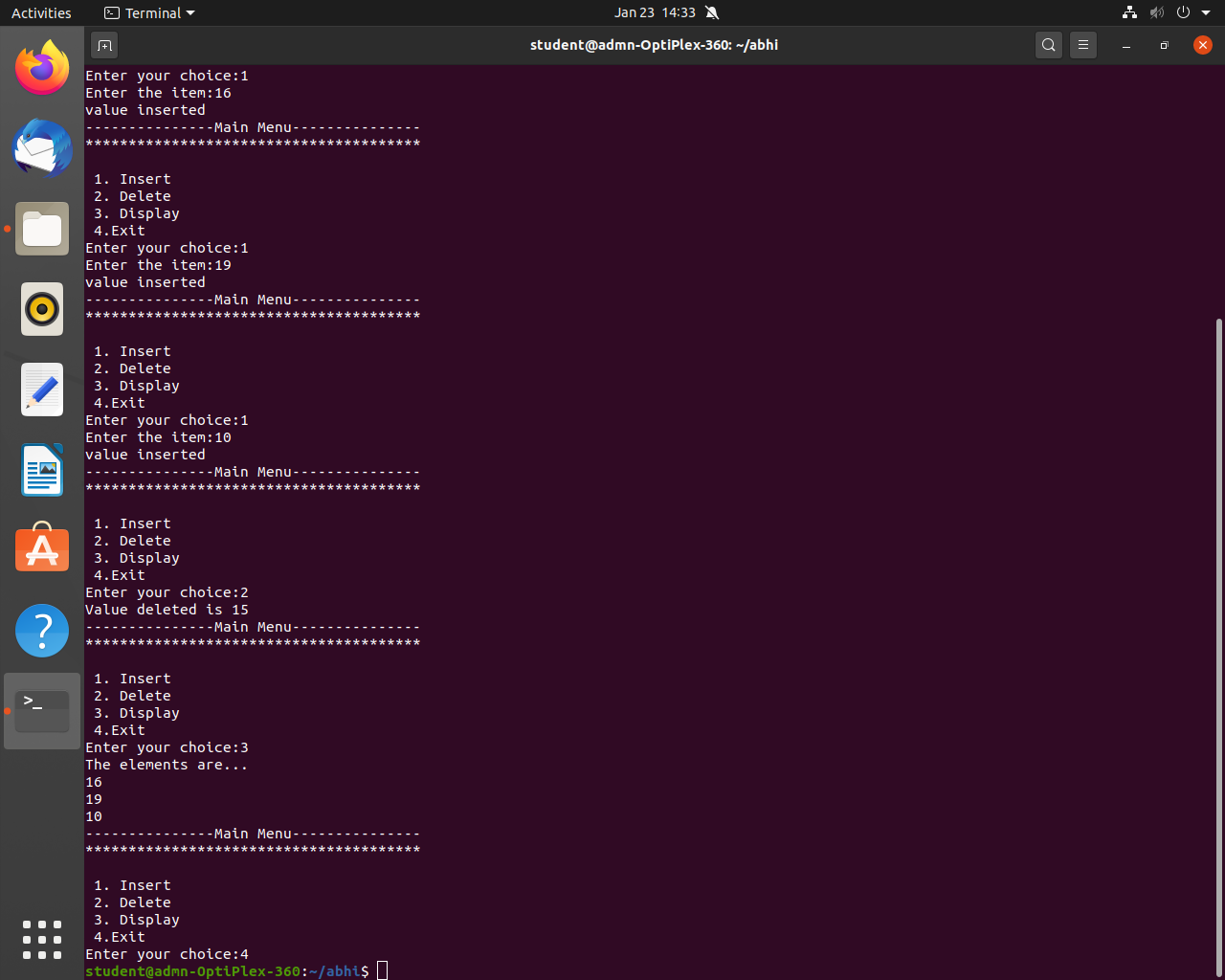
}

}

//END OF THE PROGRAM

**“SCREENSHOTS OF OUTPUT”**





//-----------------------------------------------------------------