* Assignment No. 11 *

Aim:-

Queue are frequently used in computer programming and a typical example is the creation of a job queue by an operating system. If the operating system does not use priorities then the jobs are processed in the order they enter the system. write C++ program for simulating job queue. write functions to add job and delete job from queue.

Hardware & Software Requirements:-

lab Computer, C++ complier, Eclipse for C++.

Prerequisities:-

Basic Skills of C++ programming language C++ queue.

Theory:-

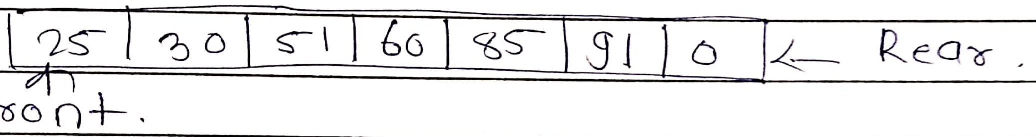
Queue is a linear data structure in which the insertion & deletion operations are performed at two different levels. In a queue data structure adding & removing of elements are performed at two different position. The insertion operation is performed at one and deletion is performed at other end. In a queue data structure the at other end. In a queue data structure, the insertion operation is performed at a position which is known as per rear and the deletion operation is performed at a position

which is known front.

In a queue data structure, the insertion operation is performed using a function called 'enque()'. & deletion operation is performed using a function called 'dequeue()'.

"Queue data structure is a collection of similar data item in which insertion & deletion operation are performed based on FIFO principle".

eg:-



Queue data structure can be implemented in two ways.

(1) using Array
(2) Using linked list.

Algorithm:

Queue operation using array.

StepI: Include all the header filis, which are used in the program and define a Constant (SIZE) with speific value

Step II: Declare all the user defined function which are used in queue implementation

Step III: Create a one dimensional array with above defined size (int queue [size])

Step IV: Define two integer variable front & rear and initialize both with -1.

Step V : Then implement main method by display menu of operation list and make suitabl function calls to perform operation selected by the user on queue.

enqueue ():-

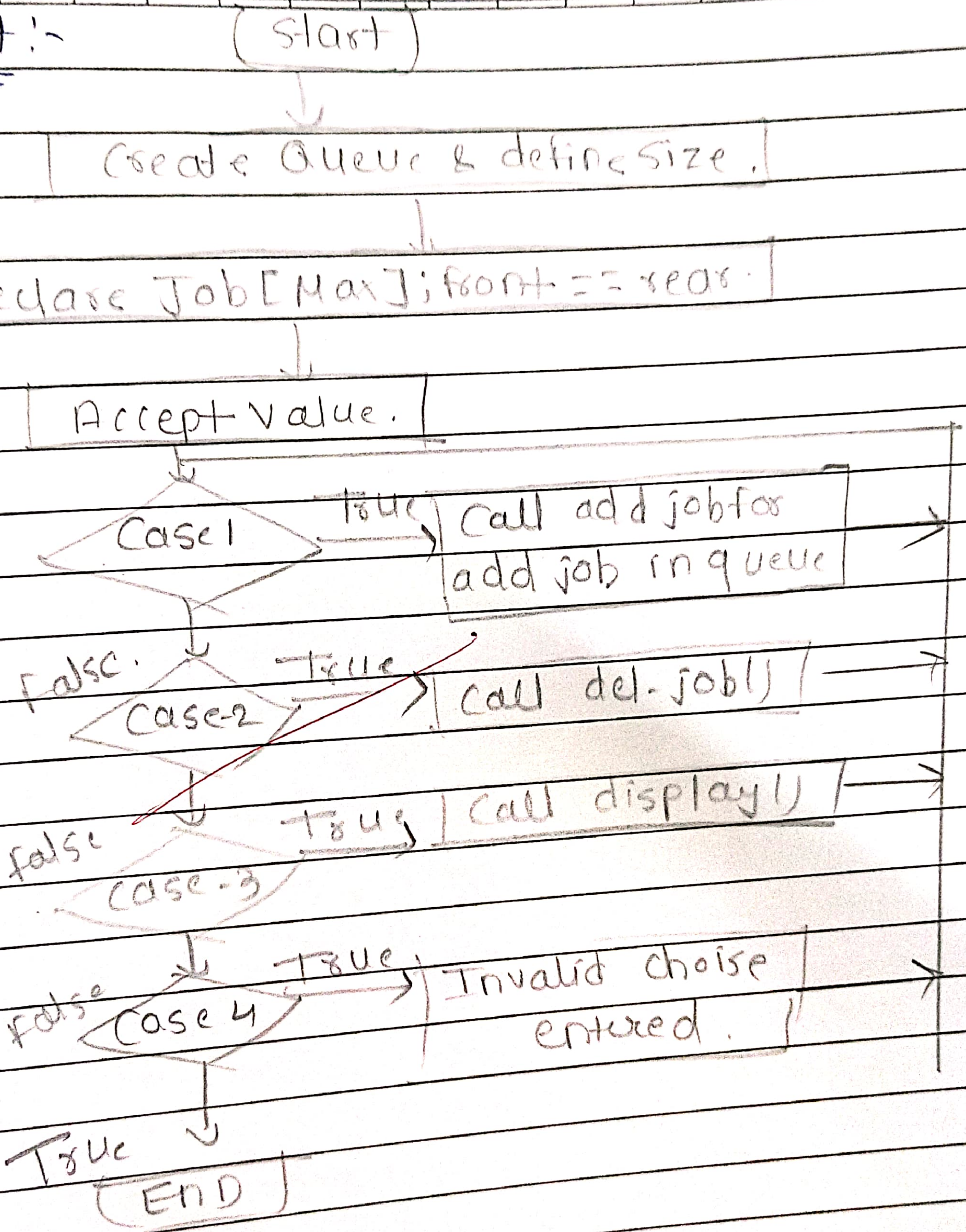
Step I :- Check whether queue is full (rear = = size)
Step II : If it is full, the display " It is full" & terminate.
Step III : If it is not full, then increment rear value by one & set queue [rear] = value.

display () :

Step I : check whether queue is empty : (front = rear)
Step II : If it is empty, terminate.
Step III : If it is not empty, terminate.
Step IV : Display que [i] value & increment value by one (i++). Repeat the same until (i) value is equal to rear (i = = rear).

dequeue() :
Step I : Check whether the queue is empty. (front = rear).
Step II : If it is empty, terminate.
Step III : If it is not empty
front ++
display queue [front].

Flowchart:-



Conclusion:-

Hence we have studied & implemented Queue & performed various operation on it.