(57) 12 (1)

PAGE:
DATE: / /

* Assignment No. 12 *

Aim :-

write a program to implement a piropity queue in ctt using an order list/array to store the item in the queue. create a class that include the data item (which should be templatel and the priority (which should be int). the order list (array should Contain these objects. with operators <= overloaded so that the item with highest priority appears at the beginning of list/array (which will make it realitely easy to retreve the highest item).

objective :-

1. to understand concept of priority queue.
2. to understand how this can be used to implement specific application
3. to understand the concept of operator overloading.

outcomes :-

1. To implement operations on priority queue.
2. To write function to use queue for job scheduling and prioritizing jobs

Os programming tools used:

64 bit os, eclipse with ctt

theory :-

priority Queue :-

PAGE:
DATE: / /

- It is an abstract data type that is similar to a queue & every element has some priority value assigned to it.
- The priority of the element in a priority queue determines the order in which elements are served.

properties of priority queue:-
① Every element has a priority associated with it.
② An element with high priority is dequed before an element with low priority. If two elements have the same priority, they are served according to their order in the queue.

Implementation of priority Queue:-
① using Array.
② Using linked list.
③ using heap data structure

operations on priority queue:-
① enqueue(): Insert new data.
② dequeue(): delete the dat of highest priority
③ peek()/top(): return the highest priority data without deleting

(5) 12 (3)

PAGE:
DATE: / /

Pseudo code:-

1) Enqueue operation:

Algorithm enqueue (a[N], f, r, item (process no, priority))

```
if (r == N-1)
    print "overflow"
if (f == -1)
    f = 0, r = 0
    Enter process no of priority, item
    q[r] = item

else:
    r = r+1
    Enta the process no & priority item
    a[r] = item
        j = r = -1
while ((a[j] <= item) && (j = F)).
    q[j+1] = q[j]
        j = j-1
        q[j+1] = item;
}
```

2) Dequeue operations:-

Algorithm dequeue [q[N], item, f, r) {

```
if ((F == -1) || (F == r+1))
    print "Queue empty"
else.
    F = F+1.
}.
```

int operation <= (item, obj 1, item obje 2) { if (ob1, p<=ob2)

return 1;

PAGE:
DATE: / /

```
else
    return 0;
    }.
```

time complexity.

enqeue () = o(n)
dequeue () = o(1)
peek() || top() = o(1)
operator %= overloaded.

Conclusion:-

the various operation of priority queue are implemented successfully using array data structure.