

PRN: 2020BTEIT00041

Priority Queue – Using LinkedList:

```

1  /*
2  |   PRN: 2020BTEIT00041
3  |   Name: Om Vivek Gharge
4  */
5
6  /*
7  |   Priority Queue - Implementation using linkedlist
8  */
9
10 #include <bits/stdc++.h>
11 using namespace std;
12
13 // Node class
14 class Node{
15 public:
16     int data, priority;
17     Node *next;
18
19     Node(){
20         this->data = 0;
21         this->priority = 0;
22         this->next = NULL;
23     }
24
25     Node(int data, int priority){
26         this->data = data;
27         this->priority = priority;
28         this->next = NULL;
29     }
30 };
31
32 // Priority Queue class
33 class PriorityQueue{
34 public:
35     Node* head;
36
37     PriorityQueue(){
38         this->head = NULL;
39     }
40
41     void enqueue(int data, int priority);
42     int dequeue();
43     void Display();
44 };
45
46 // enqueue function
47 void PriorityQueue::enqueue(int data, int priority){
48     // create a new node

```

```

49     Node *newNode = new Node(data, priority);
50
51     // if queue is empty
52     if(this->head == NULL){
53         this->head = newNode;
54         return;
55     }
56
57     // if queue is not empty
58     Node *temp = this->head;
59     Node *prev = NULL;
60
61     // traverse the queue to find the right position to insert the new node in the queue
62     while(temp != NULL){
63         if(temp->priority > priority){
64             break;
65         }
66         prev = temp;
67         temp = temp->next;
68     }
69
70     // if new node's priority is less than head's priority
71     if(prev == NULL){
72         newNode->next = this->head;
73         this->head = newNode;
74     }
75     // if new node's priority is greater than head's priority
76     else{
77         prev->next = newNode;
78         newNode->next = temp;
79     }
80
81     return;
82 }
83
84 int PriorityQueue::dequeue(){
85     // if queue is empty
86     if(this->head == NULL){
87         cout << "Queue is empty" << endl;
88         return -1;
89     }
90
91     // if queue is not empty
92     // store the head node in a temp variable
93     Node *temp = this->head;
94
95     // store the head node's data in a temp variable

```

```

96     int data = temp->data;
97
98     // make the front as the next of the front
99     this->head = this->head->next;
100
101     // delete the temp node
102     delete temp;
103
104     return data;
105 }
106
107 void PriorityQueue::Display(){
108     // if queue is empty
109     if(this->head == NULL){
110         cout << "Queue is empty" << endl;
111         return;
112     }
113
114     // if queue is not empty
115     Node *temp = this->head;
116
117     // traverse the queue and print the data of each node
118     while(temp != NULL){
119         cout << temp->data << " ";
120         temp = temp->next;
121     }
122     cout << endl;
123 }
124
125 int main(){
126     PriorityQueue pq;
127
128     // Menu driven program to implement a priority queue
129
130     int choice, data, priority;
131
132     do{
133         cout<<"-----Menu-----\n";
134         cout << "1. Enqueue" << endl;
135         cout << "2. Dequeue" << endl;
136         cout << "3. Display" << endl;
137         cout << "4. Exit" << endl;
138
139         cout << "Enter your choice: ";
140         cin >> choice;
141
142         switch(choice){
143             case 1:
144                 cout << "Enter data: ";

```

```
145         cin >> data;
146         cout << "Enter priority: ";
147         cin >> priority;
148         pq.enqueue(data, priority);
149         break;
150     case 2:
151         cout<<"Dequeued element: "<<pq.dequeue()<<endl;
152         break;
153     case 3:
154         pq.Display();
155         break;
156     case 4:
157         break;
158     default:
159         cout << "Invalid choice" << endl;
160     }
161 }while(choice != 4);
162
163 return 0;
164 }
```

## OUTPUT:

```
-----Menu-----
1. Enqueue
2. Dequeue
3. Display
4. Exit
Enter your choice: 1
Enter data: 1
Enter priority: 3
-----Menu-----
1. Enqueue
2. Dequeue
3. Display
4. Exit
Enter your choice: 1
Enter data: 2
Enter priority: 2
-----Menu-----
1. Enqueue
2. Dequeue
3. Display
4. Exit
Enter your choice: 1
Enter data: 3
Enter priority: 1
-----Menu-----
1. Enqueue
2. Dequeue
3. Display
4. Exit
Enter your choice: 3
3 2 1
-----Menu-----
1. Enqueue
2. Dequeue
3. Display
4. Exit
Enter your choice: 2
Dequeued element: 3
-----Menu-----
1. Enqueue
2. Dequeue
3. Display
4. Exit
Enter your choice: 3
2 1
```