

```

1  struct Node
2  {
3      int data;
4      Node *pNext;
5  };
6
7  Node* initNode(int value)
8  {
9      Node *p = new Node;
10
11     if (p == NULL)
12     {
13         cout << "Khong du bo nho de cap phat" << endl;
14         return NULL;
15     }
16
17     p->data = value;
18     p->pNext = NULL;
19
20     return p;
21 }
22
23 struct Queue
24 {
25     Node* pFront;
26     Node* pRear;
27 };
28
29 void initQueue(Queue &q)
30 {
31     q.pFront = NULL;
32     q.pRear = NULL;
33 };
34
35 bool isEmpty(Queue q)
36 {
37     return q.pFront == NULL;
38 };
39
40 /* 3. Viết hàm thêm phần tử vào Queue
41 */
42 void enqueue(Queue &q, Node *p)
43 {
44     if (q.pFront == NULL)
45     {
46         q.pFront = p;
47         q.pRear = p;
48     }
49     else
50     {
51         q.pRear->pNext = p;
52         q.pRear = p;
53     }
54 };
55

```

```

55
56 /* 4. Viết hàm lấy phần tử ra khỏi Queue
57 */
58 bool deQueue(Queue &q)
59 {
60     if (q.pFront == NULL)
61     {
62         return false;
63     }
64
65     Node *p = q.pFront;
66     q.pFront = q.pFront->pNext;
67     delete p;
68
69     if (q.pFront == NULL)
70     {
71         q.pRear = NULL;
72     }
73
74     return true;
75 };
76
77 void printfQueue(Queue q)
78 {
79     if (q.pFront == NULL)
80     {
81         cout << "Hang doi rong" << endl;
82         return;
83     }
84
85     cout << "<<Front<< ";
86     for (Node *p = q.pFront; p != NULL; p = p->pNext)
87     {
88         cout << p->data << " ";
89     }
90     cout << "<<Rear<<" << endl;
91 }

```