```
1  #ifndef DOUBLYLIST_H
2  #define DOUBLYLIST_H
3
4  #include <string>
5  #include <iostream>
6  using namespace std;
7
8  class Node
9  {
10 public:
11     Node() : data(0), prev(nullptr), next(nullptr) {}
12     Node(int newData, Node *newPrev, Node *newNext)
13         : data(newData), prev(newPrev), next(newNext) {}
14     int getData() const { return data; }
15     Node *getPrev() const { return prev; }
16     Node *getNext() const { return next; }
17     void setData(int newData) { data = newData; }
18     void setPrev(Node *newPrev) { prev = newPrev; }
19     void setNext(Node *newNext) { next = newNext; }
20     ~Node() {}
21 private:
22     int data;
23     Node *prev;
24     Node *next;
25 };
26
27
28 class DoublyList
29 {
30 public:
31     DoublyList();
32
33     void insertFront(int newData);
34
35     bool isEmpty() const;
36
37     ~DoublyList();
38
39     void destroyList();
40
41     /*********************************************************
42         Functions to implement
43     *********************************************************/
44
45     // Declaration function print
46     void print() const;
47
48     // Declaration function reversePrint
49     void reversePrint() const;
50
51     // Declaration function front
52     int front() const;
```

```
53
54      // Declaration function back
55      int back() const;
56
57      // Declaration function copyToList
58      void copyToList(DoublyList& otherList) const;
59
60      // Declaration function insertInOrder
61      void insertInOrder(int insertElement);
62
63  private:
64      Node *first;    // pointer to the first node on the list
65      Node *last;     // pointer to the last node on the list
66      int count;      // number of nodes in the list
67  };
68
69  #endif
```