COP 2535: Data Structures

Lab 02, Doubly Linked List

1. Read pages 68 - 71 in Mastering Algorithms with C
2. Implement the following program.
3. Upload the output of your execution as text.

```
#include <iostream>

template<class T>
class DoublyLinkedList
{
    struct Node
    {
        T data;
        Node* next;
        Node* prev;
        Node(T val) : data(val), next(nullptr), prev(nullptr) {}
    };
    Node* head, * tail;

public:
    DoublyLinkedList() : head(nullptr), tail(nullptr) {}

    ~DoublyLinkedList()
    {
        Node* tmp = nullptr;
        while (head)
        {
            tmp = head;
            head = head->next;
            delete tmp;
        }
        head = nullptr;
    }

    DoublyLinkedList(const DoublyLinkedList<T>& dll) = delete;
    DoublyLinkedList& operator=(DoublyLinkedList const&) = delete;

    void insertFront(T val)
    {
        Node* node = new Node(val);
        Node* tmp = head;
        if (head == nullptr)
        {
            head = node;
            tail = node;
        }
        else
        {
            node->next = head;
            head = node;
            node->next->prev = node;
        }
    }

    void insertBack(T val)
    {
        Node* node = new Node(val);
        if (tail->next == nullptr)
        {
            tail->next = node;
            node->prev = tail;
            tail = node;
        }
    }


    void deleteVal(T val)
    {
        Node* find = findVal(val);
        Node* tmp = head;

        if (tmp == find)
        {
            head = tmp->next;
        }
        else
        {
            while (find != nullptr)
            {
                if (tmp->next == find)
                {
                    tmp->next = find->next;
                    find->next->prev = tmp;
                    delete find;
                    return;
                }
                tmp = tmp->next;
            }
        }
    }

    template <class U>
    friend std::ostream& operator<<(std::ostream& os, const DoublyLinkedList<U>& dll) {
        dll.display(os);
        return os;
    }

private:
```

```cpp
    Node* findVal(T n) //returns node of the given number
    {
        Node* node = head;
        while (node != nullptr)
        {
            if (node->data == n)
                return node;

            node = node->next;
        }
        std::cerr << "No such element in the list \n";
        return nullptr;
    }

    void display(std::ostream& out = std::cout) const
    {
        Node* node = head;
        while (node != nullptr)
        {
            out << node->data << " ";
            node = node->next;
        }
    }
};

int main() {
    DoublyLinkedList<int> l1;
    l1.insertFront(3);
    l1.insertBack(5);
    l1.insertBack(12);
    l1.insertFront(6);
    l1.insertBack(88);
    std::cout << l1 << "\n";
    l1.deleteVal(11);
    std::cout << l1 << "\n";
    return 0;
}
```