

Министерство образования Республики Беларусь  
Учреждение образования  
«Белорусский государственный университет  
информатики и радиоэлектроники»

Кафедра инженерной психологии и эргономики

Основы алгоритмизации и программирования  
Отчет по лабораторной работе №11  
«Списки. Стек»

Выполнил: Усов А.М.  
Студент группы 310901  
Преподаватель: Кабариха В. А.

Минск 2023

Цель: Сформировать умения и навыки написания программ с использованием стека, списков.

Задание 8. Создать список из случайных целых чисел и удалить из него записи с четными числами.

```
#include <iostream>
```

```
#include <time.h>
```

```
using namespace std;
```

```
template <typename T>
```

```
// двусвязанный список
```

```
class LinkedList {
```

```
private:
```

```
    class Node {
```

```
    private:
```

```
        T data;
```

```
        int index;
```

```
        Node* next;
```

```
        Node* prev;
```

```
public:
```

```
    Node(T data, int index, Node* next = nullptr, Node* prev = nullptr) {
```

```
        this->data = data;
```

```
        this->next = next;
```

```
        this->prev = prev;
```

```
        this->index = index;
```

```
    }
```

```
    T getData() {
```

```
        return data;
    }
```

```
Node* getNext() {
    return next;
}
```

```
Node* getPrev() {
    return prev;
}
```

```
void setData(T data) {
    this->data = data;
}
```

```
void setNext(Node* next) {
    this->next = next;
}
```

```
void setPrev(Node* prev) {
    this->prev = prev;
}
```

```
int getIndex() {
    return index;
}
```

```
};
```

```
Node* head;
```

```
Node* tail;
```

```
int size;
```

```
public:
```

```
LinkedList() {  
    head = nullptr;  
    tail = nullptr;  
    size = 0;  
}
```

```
void push_first(T data) {  
    Node* temp = new Node(data,size);  
    if (head == nullptr) {  
        head = temp;  
        tail = temp;  
    }  
    else {  
        temp->setNext(head);  
        head->setPrev(temp);  
        head = temp;  
    }  
    size++;  
}
```

```
void push_back(T data) {  
    Node* temp = new Node(data,size);  
    if (tail == nullptr) {
```

```

        head = temp;
        tail = temp;
    }
    else {
        temp->setPrev(tail);
        tail->setNext(temp);
        tail = temp;
    }
    size++;
}

```

```

int GetSize() {
    return size;
}

```

```

void Delete(T data) {
    Node* temp = head;
    while (temp != nullptr) {
        if (temp->getData() == data) {
            if (temp == head) {
                head = head->getNext();
                head->setPrev(nullptr);
            }
            else if (temp == tail) {
                tail = tail->getPrev();
                tail->setNext(nullptr);
            }
            else {

```

```

        temp->getPrev()->setNext(temp->getNext());
        temp->getNext()->setPrev(temp->getPrev());
    }
    size--;
    delete temp;
    return;
}
temp = temp->getNext();
}
}

```

```

void print() {
    Node* temp = head;
    while (temp != nullptr) {
        cout << temp->getData() << " ";
        temp = temp->getNext();
    }
    cout << endl;
}

```

```

Node* GetById(int index) {
    Node* temp = head;
    while (temp != nullptr) {
        if (index == temp->getIndex()) {
            return temp;
        }
        temp = temp->getNext();
    }
    cout << "nothing this element in list";
}

```

```
return nullptr;
```

```
}
```

```
void DeleteById(int index) {
```

```
    Node* temp = head;
```

```
    while (temp != nullptr) {
```

```
        if (temp->getIndex() == index) {
```

```
            if(temp == head) {
```

```
                head = head->getNext();
```

```
                head->setPrev(nullptr);
```

```
            }
```

```
            else if (temp == tail) {
```

```
                tail = tail->getPrev();
```

```
                tail->setNext(nullptr);
```

```
            }
```

```
            else {
```

```
                temp->getPrev()->setNext(temp->getNext());
```

```
                temp->getNext()->setPrev(temp->getPrev());
```

```
            }
```

```
            size--;
```

```
            delete temp;
```

```
            return;
```

```
        }
```

```
        temp = temp->getNext();
```

```
    }
```

```
    cout << "nothing element in the list";
```

```
}
```

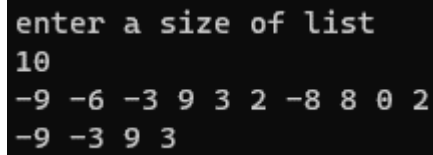
```
};
```

```
int main() {  
    // указание рандома  
    srand(time(NULL));  
    // объявление списка  
    LinkedList<int> list;  
    cout << "enter a size of list" << endl;  
    int n;  
    cin >> n;  
  
    // задание случайных чисел в список  
    for (int i = 0; i < n; i++) {  
        int a = -10 + rand() % (10 - (-10) + 1);  
        list.push_back(a);  
    }  
    // вывод списка  
    list.print();  
  
    // удаление четных записей в списке  
    for (int i = 0; i < n; i++) {  
        if (list.GetById(i)->getData() % 2 == 0) {  
            list.DeleteById(i);  
        }  
    }  
    // вывод списка
```



```
list.print();  
}
```

Результат работы программы представлен на рисунке 1.



```
enter a size of list  
10  
-9 -6 -3 9 3 2 -8 8 0 2  
-9 -3 9 3
```

Рисунок 1 – Результат выполнения программы

Вывод: лабораторная работа №11 «Списки. Стек» помогла мне сформировать умения и навыки написания программ с использованием стека и списков, а также развить навыки алгоритмизации и программирования. Полученные навыки будут полезны мне в дальнейшем при решении более сложных задач и разработке программных систем.