МИНЕСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

УЧЕРЕЖДЕНИЕ ОБРАЗОВАНИЯ

«Брестский государственный технический университет»

Кафедра «Интеллектуальные информационные технологии»

Лабораторная работа №5

По дисциплине «Объектно-ориентированное программирование и проектирование»

За 4 семестр

Тема: «Перегрузка операций»

Выполнила:

студентка 2 курса

группы АС-56

Карпенко М.В.

Проверил:

Давидюк Ю.И.

Брест 2021

Цель работы: Получить практические навыки создания абстрактных типов данных и перегрузки операций в языке C++.

Вариант 7

АТД – однонаправленный список с элементами типа **char.** Дополнительно перегрузить следующие операции:

\+ – объединить списки (list+list);

-- – удалить элемент из начала (типа --list);

= = – проверка на равенство.

Код программы:

```
#include <iostream>

using namespace std;

class LIST
{
private:
    struct list
    {
        char field;
        list* next;
    };

    list* head;
    int count = 0;

public:
    LIST();
    ~LIST();
    void print();
    void add(char);
    bool operator ==(const LIST&); //проверка на равенство
    void operator +=(const LIST&);
    void operator --();
};

LIST::LIST()
{
    head = NULL;
}

LIST::~LIST()
{
    list* nd;
    while (head != NULL)
    {
        nd = head;
        head = head->next;
        delete nd;
    }
}

void LIST::operator +=(const LIST& obj)
{
    list* p;
    p = obj.head;
    do
```

```
    {
        this->add(p->field);
        p = p->next; // переход к следующему узлу
    } while (p != NULL);
}

void LIST::add(char ch)
{
    list* lst = new list;
    lst->field = ch;
    lst->next = NULL;

    if (head == NULL)
    {
        head = lst;
    }
    else
    {
        list* current = head;

        while (current->next != NULL)
        {
            current = current->next;
        }
        current->next = lst;
    }
    count++;
}

void LIST::operator--()
{
    head = head->next;
}

bool LIST::operator==(const LIST& lest)
{
    if (this->count != lest.count)
    {
        return false;
    }
    else
    {
        int countCheck = 0;
        list* currentFirst = this->head;
        list* currentSecond = lest.head;
        while (currentFirst != NULL && currentSecond != NULL)
        {
            if (currentFirst->field == currentSecond->field) countCheck++;
            currentFirst = currentFirst->next;
            currentSecond = currentSecond->next;
        }
        return countCheck == count;
    }
}

void LIST::print()
{
    list* current = head;

    while (current != NULL)
    {
        cout << current->field << endl;
        current = current->next;
    }
}

int main()
{
```

```
    LIST A;
    A.add('1');
    A.add('a');
    A.print();

    LIST B;
    B.add('1');
    B.add('2');
    if (B == A)
        cout << "work)" << endl;
    else cout << "don't work" << endl;

    A += B;
    A.print();
    cout << endl;
   --A;
    A.print();

    return 0;
}
```

Результат работы программы:

```
1
a
don't work
1
a
1
2

a
1
2
```

Вывод: Получила практические навыки создания абстрактных типов данных и перегрузки операций в языке C++.