**ОТЧЕТ  
по лабораторной работе** **№1**  
по дисциплине «Технология программирования»

Выполнил:

Студент группы НПИбд-02-22

Студенческий билет № 1132227130

Яковин Илья Андреевич

Москва 2023

**ЗАДАНИЕ**

Написать компьютерную программу, содержащую

- Описание структуры, содержащей поля типа string, int, double;

- Набор функций для работы со списком на базе этой структуры:

- Добавление элемента в начало списка;

- Добавление элемента в конец списка;

- Добавление элемента в список после заданного элемента;

- Добавление элемента в список перед заданным элементом;

- Удаление из списка элемента с заданным именем;

- Вывод содержания списка на экран;

- Функцию main, содержащую сценарий работы со списком, использующий разработанный инструментарий.

**ВЫПОЛНЕНИЕ ЛАБОРАТОРНОЙ РАБОТЫ**

#include <iostream>

#include <list>

using namespace std;

struct Node

{

string name;

int age;

double salary;

Node\* next;

};

class LinkedList

{

private:

Node\* head;

public:

LinkedList() {

head = NULL;

}

// Добавление элемента в начало списка

void add\_to\_begin(string name, int age, double salary)

{

Node\* new\_node = new Node;

new\_node->name = name;

new\_node->age = age;

new\_node->salary = salary;

new\_node->next = head;

head = new\_node;

}

// Добавление элемента в конец списка

void add\_to\_end(string name, int age, double salary)

{

Node\* new\_node = new Node;

new\_node->name = name;

new\_node->age = age;

new\_node->salary = salary;

new\_node->next = NULL;

if (head == NULL)

{

head = new\_node;

}

else

{

Node\* current = head;

while (current->next != NULL)

{

current = current->next;

}

current->next = new\_node;

}

}

// Добавление элемента после заданного элемента

void add\_after(string name, int age, double salary, string after\_name)

{

Node\* new\_node = new Node;

new\_node->name = name;

new\_node->age = age;

new\_node->salary = salary;

Node\* current = head;

while (current != NULL && current->name != after\_name)

{

current = current->next;

}

if (current != NULL)

{

new\_node->next = current->next;

current->next = new\_node;

}

else

{

cout << "Element not found!" << endl;

}

}

// Добавление элемента перед заданным элементом

void add\_before(string name, int age, double salary, string before\_name)

{

Node\* new\_node = new Node;

new\_node->name = name;

new\_node->age = age;

new\_node->salary = salary;

if (head == NULL)

{

head = new\_node;

}

else if (head->name == before\_name)

{

new\_node->next = head;

head = new\_node;

} else {

Node\* current = head;

while (current->next != NULL && current->next->name != before\_name) {

current = current->next;

}

if (current->next != NULL) {

new\_node->next = current->next;

current->next = new\_node;

} else {

cout << "Element not found!" << endl;

}

}

}

// Удаление элемента с заданным именем

void remove(string name)

{

if (head == NULL) {

return;

} else if (head->name == name)

{

Node\* temp = head;

head = head->next;

delete temp;

}

else

{

Node\* current = head;

while (current->next != NULL && current->next->name != name)

{

current = current->next;

}

if (current->next != NULL)

{

Node\* temp = current->next;

current->next = temp->next;

delete temp;

}

else

{

cout << "Element not found!" << endl;

}

}

}

// Вывод содержания списка на экран

void print()

{

if (head == NULL)

{

cout << "List is empty!" << endl;

return;

}

Node\* current = head;

while (current != NULL)

{

cout << "Name: " << current->name << " Age: " << current->age << " Salary: " << current->salary << endl;

current = current->next;

}

}

};

int main()

{

LinkedList list;

return 0;

}

Этот код представляет собой реализацию связного списка на языке C++. Связный список — это динамическая структура данных, состоящая из узлов, которые содержат в себе в классическом варианте два значения: первое — это какое-либо данное, а второе — это указатель на следующий узел в списке.

Структура **Node** содержит информацию об узле списка. Также каждый узел содержит указатель на следующий элемент в списке.

Класс **LinkedList** содержит функции для добавления, удаления и вывода элементов списка. Конструктор класса создает пустой список.

Функции для добавления элементов в список могут добавлять элементы в начало, в конец, перед или после указанного элемента. Функция **remove** удаляет элемент с заданным именем из списка.

Функция **print** выводит содержимое списка на экран

В основной функции **main** вызывается объект класса **LinkedList**.