Пермский Национальный Исследовательский Политехнический Университет

Электротехнический факультет

Кафедра «Информационные технологии и автоматизированные системы»

Лабораторная работа № 11.1

по дисциплине

«Основы алгоритмизации и программирования»

Тема: «Динамические структуры данных. Двунаправленный список»

Вариант № 17

Выполнил:

студент группы РИС-20-2б

Пономарев Егор Витальевич

Проверила:

доцент кафедры ИТАС

Полякова О. А.

Пермь, 2021

**Цель работы**

Получить практические навыки работы с двунаправленными списками.

**Постановка задачи**

* Сформировать двунаправленный список с помощью структуры и библиотеки STL.
* Распечатать полученную структуру.
* Выполнить обработку структуры в соответствии с заданием.
* Распечатать полученный результат.
* Удалить соответствующую структуру из памяти.

Задание варианта:

1. Тип информационного поля char\*.
2. Добавить в список элемент с заданным номером.

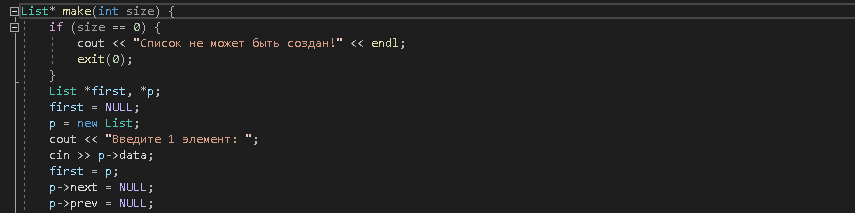
**Анализ задачи**

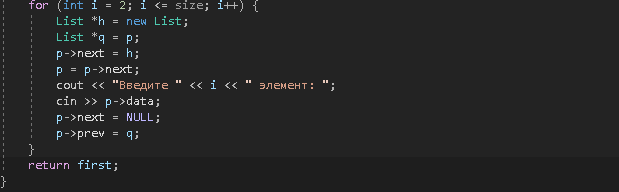
**Реализация через структуру:**

1. Необходимые действия:
2. Объявляем структуру List с информационным полем типа char\*. (так как работать необходимо со строками, нужно выделить динамическую память с помощью new.



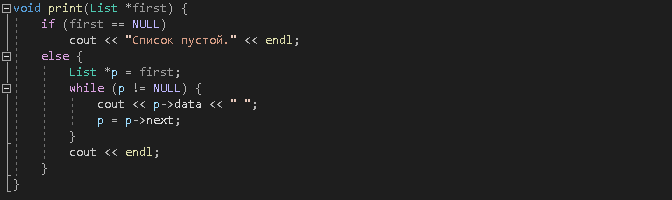
1. Функция создания списка почти ничем не отличается от однонаправленного списка, кроме того, что мы должны помимо указателя на следующий элемент объявить указатель на предыдущий.

****

****

****

1. Функция печати точно такая же, как и у однонаправленного списка.

****

1. Чтобы добавить элемент в заданное место, переходим к последнему элементу с помощью цикла. В конце списка создаем новый элемент и смещаем влево все элементы до того, куда необходимо добавить элемент, введенный пользователем.



1. В ходе работы использовались следующие типы данных:
2. Int для счетчиков и размера.
3. Структура данных List с информационным полем char\* и указателями на следующий List\* next и предыдущий List\* prev элементы.

3. В программе использованы циклы while, for и условные ветвления if-else.

4. Ввод и вывод осуществляется с помощью cin и cout.

**Реализация через STL:**

1. В первую очередь необходимо подключить библиотеку <list> для работы с двунаправленным списком и библиотеку <iterator> (Итератор — это такая структура данных, которая используется для обращения к определенному элементу в контейнерах STL – по сути указатель).



1. Объявляем список по следующему шаблону:

list < тип данных > <имя контейнера>



1. Создаем список с помощью метода push\_back ().



1. Для добавления элемента в заданное место объявляем итератор it и с помощью метода begin () возвращаем начало списка и приравниваем его к итератору.

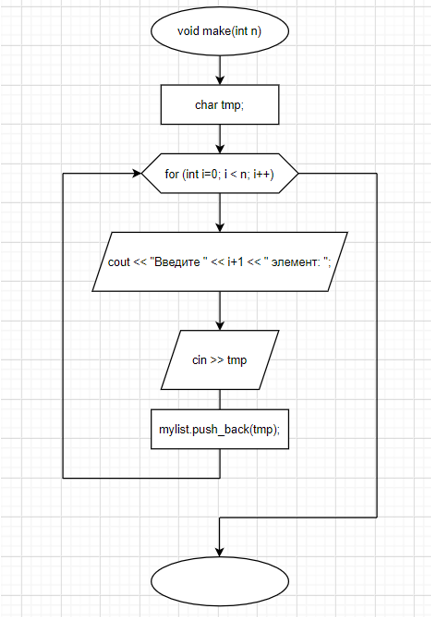


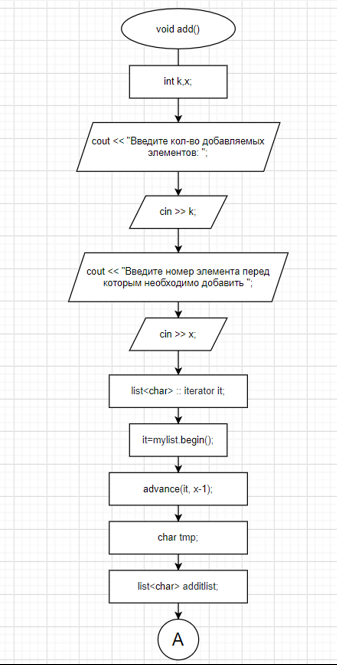
1. С помощью функции advance () передвигаем итератор в нужное место и вставляем туда введенный элемент.

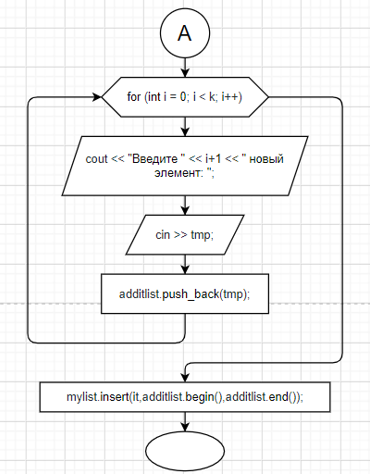
****

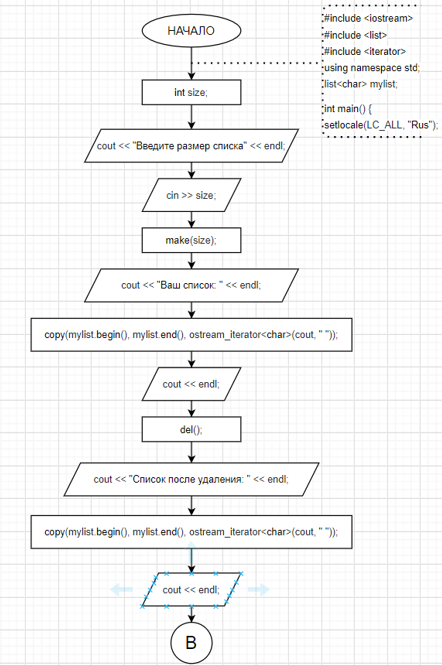
1. В ходе работы использованы следующие типы данных:
2. Int для счетчиков и размера.
3. Итератор **it** по списку.

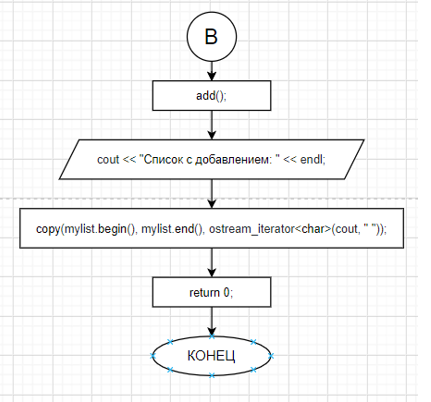
**Блок-схема**

****

****

****

****

****

**Код С++**

**Реализация через структуру:**

#include <locale.h>

#include <iostream>

using namespace std;

struct List {

char \*data = new char;

List \*next;

List \*prev;

};

List\* make(int size) {

if (size == 0) {

cout << "Список не может быть создан!" << endl;

exit(0);

}

List \*first, \*p;

first = NULL;

p = new List;

cout << "Введите 1 элемент: ";

cin >> p->data;

first = p;

p->next = NULL;

p->prev = NULL;

for (int i = 2; i <= size; i++) {

List \*h = new List;

List \*q = p;

p->next = h;

p = p->next;

cout << "Введите " << i << " элемент: ";

cin >> p->data;

p->next = NULL;

p->prev = q;

}

return first;

}

void print(List \*first) {

if (first == NULL)

cout << "Список пустой." << endl;

else {

List \*p = first;

while (p != NULL) {

cout << p->data << " ";

p = p->next;

}

cout << endl;

}

}

void add(List\* first, int size) {

int n;

char\* k = new char;

cout << "Введите номер элмента, который нужно добавить: ";

cin >> n;

while (n <= 0 || n > size) {

cout << "Неверный номер! Введите ещё раз: ";

cin >> n;

}

cout << "Введите этот элемент: ";

cin >> k;

List \*p = first;

while (p->next != NULL) {

p = p->next;

}

List \*h = new List;

h->next = NULL;

h->prev = p;

p->next = h;

p = p->next;

for (int i = size + 1; i > n; i--) {

p->data = p->prev->data;

p = p->prev;

}

p->data = k;

}

int main()

{

setlocale(LC\_ALL, "rus");

int size;

cout << "Введите размер списка: ";

cin >> size;

List \*list = make(size);

cout << "Ваш список: " << endl;

print(list);

add(list, size);

print(list);

return 0;

}

**Реализация через STL:**

#include <locale.h>

#include <iostream>

#include <list>

#include <iterator>

using namespace std;

list<char>mylist;

void make(int n) {

char tmp;

for (int i = 0; i < n; i++) {

cout << "Введите " << i + 1 << " элемент: ";

cin >> tmp;

mylist.push\_back(tmp);

}

}

void add(int size) {

int n;

char k;

cout << endl << "Введите номер элемента, который нужно добавить: ";

cin >> n;

while (n <= 0 || n > size) {

cout << "Неверный номер! Введите ещё раз: ";

cin >> n;

}

cout << "Введите этот элемент: ";

cin >> k;

list<char> ::iterator it;

it = mylist.begin();

advance(it, n - 1);

mylist.insert(it, k);

}

int main()

{

setlocale(LC\_ALL, "rus");

int size;

cout << "Введите количество элементов: ";

cin >> size;

while (size < 1) {

cout << "Список не может быть создан. Введите ещё раз: ";

cin >> size;

}

make(size);

cout << "Ваш список: " << endl;

copy(mylist.begin(), mylist.end(), ostream\_iterator<char>(cout, " "));

add(size);

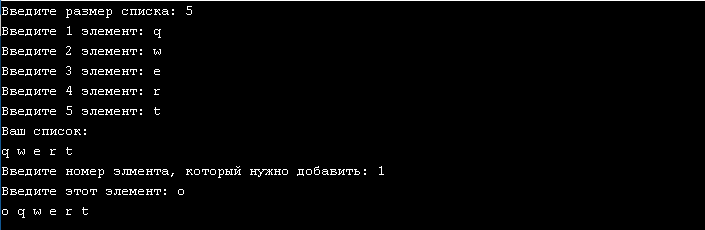
copy(mylist.begin(), mylist.end(), ostream\_iterator<char>(cout, " "));

return 0;

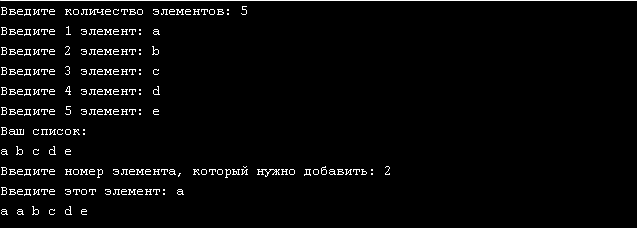
}

**Результаты выполнения**

**Структура:**

****

**STL:**

****