**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ**

**РОССИЙСКОЙ ФЕДЕРАЦИИ**

федеральное государственное автономное образовательное учреждение высшего образования «Санкт-Петербургский политехнический университет Петра Великого»

(ФГАОУ ВО «СПбПУ»)

**Институт среднего профессионального образования**

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 11

по дисциплине: «Основы алгоритмизации и программирования»

**Студент группы № 219/2**

**Клопова Ульяна**

**Вариант 15**

**Преподаватель: Рябова В.В.**

Санкт-Петербург

2025 г.

[**1 ЧАСТЬ**](../программы/Пирамидка%20пирамидка/Пирамидка%20пирамидка.sln)  **LIST**

Создайте пустой список и добавьте в него n целых чисел. Выведите значения списка на экран.

1. Реализовать меню по работе с данным списком (учесть для некоторых задач, что могут быть повторы)
2. Вывод списка на консоль
3. Добавление числа
   * в начало списка
   * в конец списка
   * на произвольную позицию
4. Удаление элемента
   * Первого
   * Последнего
   * Произвольного
5. Проверка на наличие элемента
6. Замена найденного числа на введенное пользователем
7. Удаление дубликатов из списка
8. Реверс Списка
9. Сортировка списка
10. Поиск максимального и минимального значений
11. Подсчет количества четных и нечетных

#include <iostream>

#include <deque>

#include <list>

#include <string>

#include <algorithm>

#include<set>

#ifdef \_WIN32

#define CLEAR "cls"

#else

#define CLEAR "clear"

#endif

using namespace std;

int main()

{

setlocale(0, "");

int n;

int name;

list<int> b;

cout << "Размерность: " << endl;

int c;

cin >> c;

cout << "Наполнение: " << endl;

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

cin >> name;

b.push\_back(name);

}

cout << "Список: " << endl;

for (const int& s : b) {

cout << s << endl;

}

b.assign(b.begin(), b.end());

do {

system(CLEAR);

cout << "Список: " << endl;

for (const int& s : b) {

cout << s << endl;

}

cout << "Номер: " << endl;

cout << "1: Добавить число в начало, конец, на произвольную позицию\n";

cout << "2: Удалить число из начала, конца и произвольно\n";

cout << "3: Проверка наличия элемента\n";

cout << "4: Найти и земенить\n";

cout << "5: Удаление дубликатов\n";

cout << "6: Реверс Списка\n";

cout << "7: Сортировка списка\n";

cout << "8: Поиск максимального и минимального значений\n";

cout << "9: Подсчет количества четных и нечетных\n";

cout << "10: Выход\n";

cin >> n;

switch (n)

{

case 1: {

cout << "New list: ";

cin >> name;

b.push\_back(name);

b.push\_front(name);

for (int num : b) {

cout << num << endl;

}

cout << "После какой позиции вставить?: ";

int z;

cin >> z;

if (z > 0 && z < b.size()) {

auto iter1 = b.begin();

advance(iter1, z);

b.insert(iter1, name);

for (int num : b) {

cout << num << endl;

}

}

}

break;

case 2: {

cout << "Человечество победило, удаление системы>:)"<<endl;

b.pop\_front();

b.pop\_back();

for (int num : b) {

cout << num << endl;

}

cout << "После какой позиции удалить?: ";

int pop;

cin >> pop;

if (pop > 0 && pop < b.size()) {

auto iter1 = b.begin();

advance(iter1, pop);

b.erase(iter1);

for (int num : b) {

cout << num << endl;

}

}

break;

}

case 3:

{

cout << "Что найти? ";

int fin;

cin >> fin;

auto it = find(b.begin(), b.end(), fin);

if (it != b.end()) {

cout << "Элемент " << fin<< " есть" << endl;

}

else {

cout << "Элемента " << fin << ". нет " <<endl;

}

break;

}

case 4: {

cout << "Что найти? ";

int fin2, zam;

cin >> fin2;

auto it2 = find(b.begin(), b.end(), fin2);

cout << "На что заменям? ";

cin >> zam;

\*it2 = zam;

cout << "Тадааам: " << endl;

for (int num : b) {

cout << num << endl;

}

break;

}

case 5:

{

cout << "Уничтожение жалких пародий: " << endl;

set<int>sus(b.begin(), b.end());

b.assign(sus.begin(), sus.end());

for (int num : b) {

cout << num << endl;

}

break;

}

case 6:

{

cout << "Реверс: " << endl;

b.reverse();

for (int num : b) {

cout << num << endl;

}

break;

}

case 7:

{

cout << "Сортировка: " << endl;

b.sort();

for (int num : b) {

cout << num << endl;

}

break;

}

case 8:

{

auto mint = min\_element(b.begin(), b.end());

auto maxt = max\_element(b.begin(), b.end());

cout << "Мин: " << \*mint << endl;

cout << "Макс: " << \*maxt << endl;

}

case 9:

{

int chet = 0;

int nech = 0;

for (int num9 : b) {

if (num9 % 2 == 0) {

chet++;

}

else {

nech++;

}

}

cout << "Количество четных чисел: " << chet <<endl;

cout << "Количество нечетных чисел: " << nech << endl;

}

case 10:

{

cout << "Адьос" << endl;

}

}

cout << "\nНажмите Enter, чтобы продолжить...";

cin.ignore();

cin.get();

} while (n != 10);

}

**Результаты запуска:**

**Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.**

**Первый запуск**

[**2 ЧАСТЬ**](../программы/бонго%20кот/бонго%20кот.sln)  **DEQUE И QUEUE**

**Вариант 15 (30)**

Создать std::queue для хранения строк, представляющих задачи. Выполнить следующие действия:

Добавить 5 задач в очередь (например, "Сделать уборку", "Купить продукты" использовать getline для ввода строки с пробелами).

Вывести и удалить первую задачу из очереди.

Проверить размер очереди и вывести его на экран после удаления задачи.

Добавить еще 3 задачи и вывести все оставшиеся задачи в очереди.

#include <iostream>

#include <queue>

#include <list>

#include <string>

#ifdef \_WIN32

#define CLEAR "cls"

#else

#define CLEAR "clear"

#endif

using namespace std;

int main()

{

setlocale(0, "");

int n;

string name;

queue<string>a;

cout << "Добавление 5 задач: " << endl;

for (int i = 0; i < 5; i++) {

getline(cin, name);

a.push(name);

}

cout << "\nВывод первой строки и удаление: " << a.front() << endl;

a.pop();

cout << "Размер очереди после удаления первой задачи: " << a.size() << endl;

cout << "\nВведите еще 3 задачи:" << endl;

for (int i = 0; i < 3; i++) {

getline(cin, name);

a.push(name);

}

cout << "\nОставшиеся задачи в очереди:" << endl;

while (!a.empty()) {

cout << a.front() << endl;

a.pop();

}

}

**Результаты запуска:**

**Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.**

**Первый запуск**

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

**Второй запуск**

**Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.**

**Третий запуск**

**3 ЧАСТЬ STACK**

[**3.1**](../программы/In%20the%20back%20of%20my%20mind%20you%20died)  **ОБЩЕЕ ЗАДАНИЕ (выполняют все)**

Общее для всех. Создание стека: Напишите программу, которая создает стек целых чисел и добавляет в него n элементов. Выведите элементы стека на экран, но при этом не очищая сам стек.

#include <iostream>

#include <stack>

using namespace std;

int main() {

setlocale(0, "");

int n,num;

cout << "Количество: ";

cin >> n;

stack<int> a;

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

cin >> num;

a.push(num);

}

cout << "\nСтек:" << endl;

stack<int> b = a;

while (!b.empty()) {

cout << b.top() << endl;

b.pop();

}

}

**Результаты запуска:**

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Первый запуск

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Второй запуск

Изображение выглядит как снимок экрана, текст, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Третий запуск

[**Вариант**](../программы/Господи%20боже/Господи%20боже.sln) **15 (30).** Реализуйте функцию, которая возвращает верхний элемент стека без его удаления, и выведите этот элемент на экран. Предусмотрите проверку на пустоту стека.

#include <iostream>

#include <stack>

using namespace std;

int voz(stack<int>& b) {

if (b.empty()) {

return -1;

}

return b.top();

}

int main() {

setlocale(0, "");

stack<int> a;

int n, num;

cout << "Количество: ";

cin >> n;

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

cin >> num;

a.push(num);

}

int c = voz(a);

if (c == -1) {

cout << "Нет" << endl;

}

else {

cout << "Верхний элемент: " << c << endl;

}

}

**Результаты запуска:**

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Первый запуск

Изображение выглядит как текст, Шрифт, снимок экрана, дизайн

Контент, сгенерированный ИИ, может содержать ошибки.

Второй запуск

Изображение выглядит как текст, Шрифт, снимок экрана, дизайн

Контент, сгенерированный ИИ, может содержать ошибки.

Третий запуск

**4 ЧАСТЬ SET UNORDERED\_SET MULTISET**

[**Вариант**](../программы/Дно/Дно.sln) **15 (30).** Создайте множество строк, добавьте несколько значений и проверьте, содержится ли определенная строка в множестве.

#include <iostream>

#include <set>

#include <string>

using namespace std;

int main() {

setlocale(0, "");

set<string> a;

int n;

string b, s;

cout << "Количество: ";

cin >> n;

cin.ignore();

cout << "Множество: " << endl;

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

getline(cin, b);

a.insert(b);

}

cout << "Элементы множества:" << endl;

for (const string& num : a) {

cout << num << endl;

}

cout << "\nПоиск: ";

getline(cin, s);

if (a.find(s) == a.end()) {

cout << "Строка \"" << s << "\" нет" << endl;

}

else {

cout << "Строка \"" << s << "\" есть" << endl;

}

}

**Результаты запуска:**

Изображение выглядит как текст, снимок экрана, Шрифт, дизайн

Контент, сгенерированный ИИ, может содержать ошибки.

Первый запуск

Изображение выглядит как текст, снимок экрана, Шрифт, дизайн

Контент, сгенерированный ИИ, может содержать ошибки.

Второй запуск

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Третий запуск

[**5.1**](../программы/Ситком/Ситком.sln) **ОБЩЕЕ ЗАДАНИЕ (выполняют все)**

Напишите программу, которая позволяет хранить имена людей и их номера телефонов в телефоном справочнике, используя std::map.

Реализуйте следующие функции:

* Добавление нового контакта: Пользователь вводит имя и номер телефона. Добавьте его в телефонную книгу.
* Поиск контакта: Пользователь вводит имя, и программа выводит номер телефона, если контакт есть в телефонной книге.
* Удаление контакта: Пользователь вводит имя, и программа удаляет контакт из телефонной книги, если он существует.
* Просмотр справочника: Выведите все имена и номера телефонов в телефонном справочнике.

Не забудьте обрабатывать случаи, когда пользователь пытается добавить контакт с уже существующим именем или удалить контакт, которого нет в справочнике. Используйте функцию count или проверку на существование ключа в std::map.

#include <iostream>

#include <map>

#include <string>

#ifdef \_WIN32

#define CLEAR "cls"

#else

#define CLEAR "clear"

#endif

using namespace std;

int main() {

setlocale(0, "");

map<string, string> a;

int cho;

string name, phone;

do {

system(CLEAR);

cout << "Телефонный справочник:" << endl;

for (const auto& b: a) {

cout << b.first << ": " << b.second << endl;

}

cout << "\nТелефонный справочник" << endl;

cout << "1. Добавить новый контакт" << endl;

cout << "2. Найти контакт" << endl;

cout << "3. Удалить контакт" << endl;

cout << "4. Просмотреть справочник" << endl;

cout << "0. Выход" << endl;

cout << "Выберите действие: ";

cin >> cho;

cin.ignore();

switch (cho) {

case 1: {

cout << "Введите имя: ";

getline(cin, name);

cout << "Введите номер телефона: ";

getline(cin, phone);

if (a.count(name) > 0) {

cout << "Контакт с именем \"" << name << "\" уже существует." << endl;

}

else {

a[name] = phone;

cout << "Контакт \"" << name << "\" успешно добавлен." << endl;

}

break;

}

case 2: {

cout << "Введите имя для поиска: ";

getline(cin, name);

if (a.count(name) > 0) {

cout << "Номер телефона для \"" << name << "\": " << a[name] << endl;

}

else {

cout << "Контакт с именем \"" << name << "\" не найден." << endl;

}

break;

}

case 3: {

cout << "Введите имя для удаления: ";

getline(cin, name);

if (a.count(name) > 0) {

a.erase(name);

cout << "Контакт \"" << name << "\" успешно удален." << endl;

}

else {

cout << "Контакт с именем \"" << name << "\" не найден." << endl;

}

break;

}

case 4: {

if (a.empty()) {

cout << "Справочник пуст." << endl;

}

else {

cout << "Телефонный справочник:" << endl;

for (const auto& pair : a) {

cout << pair.first << ": " << pair.second << endl;

}

}

break;

}

case 0: {

cout << "Выход из программы." << endl;

break;

}

default: {

cout << "Нет" << endl;

}

}

cout << "\nНажмите Enter два раза, чтобы продолжить...";

cin.ignore();

cin.get();

} while (cho != 0);

}

**Результаты запуска:**

Изображение выглядит как текст, Шрифт, снимок экрана

Контент, сгенерированный ИИ, может содержать ошибки.

Первый запуск

[**5.2**](../программы/Рандомыш/Рандомыш.sln)  **ЗАДАНИЕ**

**Вариант 15 (30).** Напишите программу, которая создает std::map для хранения цен на товары и позволяет изменять цену товара по ключу.

#include <iostream>

#include <map>

#include <string>

using namespace std;

int main() {

setlocale(0, "");

map<string, double> zen;

zen["Apple"] = 1.50;

zen["Banana"] = 0.75;

zen["Orenge"] = 1.20;

zen["Milk"] = 2.00;

cout << "Текущие цены на товары:" << endl;

for (const auto& item : zen) {

cout << item.first << ": " << " " << item.second << " $" << endl;

}

string name;

cout << "\nВведите название товара: ";

getline(cin, name);

if (zen.count(name) > 0) {

double newz;

cout << "Введите новую цену для товара \"" << name << "\": ";

cin >> newz;

zen[name] = newz;

cout << "Цена товара \"" << name << "\" успешно изменена на " << " " << newz << " $." << endl;

}

else {

cout << "Товар с названием \"" << name << "\" не найден" << endl;

}

cout << "\nОбновленные цены на товары:" << endl;

for (const auto& item : zen) {

cout << item.first << ": " << " " << item.second << " $" << endl;

}

}

**Результаты запуска:**

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Первый запуск

Изображение выглядит как текст, снимок экрана, Шрифт, дизайн

Контент, сгенерированный ИИ, может содержать ошибки.

Второй запуск

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Третий запуск