Федеральное государственное автономное образовательное учреждение

высшего образования

«ЮЖНЫЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

ИНСТИТУТ КОМПЬЮТЕРНЫХ ТЕХНОЛОГИЙ И ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ

КАФЕДРА СИСТЕМНОГО АНАЛИЗА И ТЕЛЕКОММУНИКАЦИИ

**ПРОВЕРИЛ ВЫПОЛНИЛ**

Старший преподаватель кафедры САиТ Студент группы КТбо1-2

Лапшин В.С. Гудков М.Д

«\_\_» \_\_\_\_\_\_\_\_\_\_2023 г. «\_\_» \_\_\_\_\_\_\_\_\_\_2023 г.

**ЛАБОРАТОРНАЯ РАБОТА №3**

По дисциплине

«Основы алгоритмизации и программирования»

Вариант 8

На тему

«**Динамическое распределение памяти, списки**»

**Таганрог 2023**

**Цель**: освоение студентами способов представления и обработки данных в виде списков.

**Задачи**: Написать программы для работы с односвязным и двусвязным списками

**Ход работы**:

**Составление математической модели**:

Для хранения данных выделяется память. Ее выделение может осуществляться статическим или динамическим способом. Статическое выделение памяти. При статическом способе выделения памяти компилятор, встретив в исходном коде программы объявление объекта, определяет тип объекта и резервирует для него потребную память. Класс памяти задает место, где этот объект будет располагаться. В роли объектов могут выступать сегменты данных, стеки, списки, очереди и пр. Динамическое выделение памяти. В практике программирования часто возникают ситуации, когда заранее не известно число объектов (чисел, строк текста и прочих данных), которые будет хранить и обрабатывать программа, тогда применяется, т.н. динамическое выделение памяти. Под динамическим выделением понимается процесс выделения или освобождения памяти в процессе исполнения программы. Использование динамической памяти устраняет необходимость отслеживания программистом распределения памяти под объекты данных. При динамическом выделении памяти для хранения данных используется специальная область памяти, называемая «куча» (heap). Объем кучи и ее место расположения зависят от модели памяти, которая определяет логическую структуру памяти программы.

**Реализация.**

#include <iostream>

#include <fstream>

#include <cstdlib>

#include <ctime>

#include <algorithm>

#include <string>

#include <list>

#include <cmath>

using namespace std;

struct Recipe { // создаем структуру которую будем использовать. в структуре находятся 3 строчки

string name;

string ingredients;

string instructions;

};

list<Recipe> RecipeBook{

{"Шашлык по-грузински", "Свиная шея - 1 кг, лук - 0.5 кг, соль, перец", "Маринуем мясо, жарим на углях - PROFIT!"},

{"Борщ", "Говядина на косточке - 1 кг, картофель - 1 кг, лук - 1 шт, морковь - 1 шт, свекла - 1 шт", "Варим бульон, крошим овощи, варим 3 часа - PROFIT!"},

{"Яичница", "Яйцо куриное - 2 шт, масло сливочное - 10 г", "Растапливаем масло на сковороде, вбиваем яйца, жарим 3 минуты"} };

int menu() {

cout << "\n Выберите действие: \n";

cout << "0. Вывести список на экран \n";

cout << "1. Включить новый элемент в конец списка \n";

cout << "2. Включить новый элемент в заданное место \n";

cout << "3. Включить новый элемент ПОСЛЕ элемента с заданной информацией \n";

cout << "4. Включить новый элемент ПЕРЕД элементом с заданной информацией \n";

cout << "5. Включить новый элемент в середину списка \n";

cout << "6. Исключить элемент из середины списка \n";

cout << "7. Исключить элемент с заданной информацией \n";

cout << "8. Исключить элемент из конца списка \n";

cout << "9. Исключить элемент из заданного места \n";

cout << "10. Исключить элемент из головы списка \n";

cout << "11. Выход. \n";

int a; //ввести в a

cin >> a;

return a; // для работы меню все эти 3 строки(типа чел ввёл какую-то цифру)

}

int main() {

system("chcp 1251"); // настраиваем кодировку консоли

int a = menu(); //вывели меню

while (a != 11) {

if (a == 0) {

int i = 1;

for (Recipe rec : RecipeBook)

{

cout << "Номер рецепта: " << to\_string(i++) << endl;

cout << "Название блюда: " << rec.name << endl;

cout << "Ингридиенты: " << rec.ingredients << endl;

cout << "Инструкция по приготовлению: " << rec.instructions << endl;

}

}

if (a == 1) {

Recipe NewRecipe; // Для нового рецепта

cout << "\n Выбран режим добавления нового блюда в конец списка" << endl;

cout << "Введите название блюда: ";

cin.ignore();

getline(cin, NewRecipe.name);

cout << "Введите ингридиенты для блюда через запятую: ";

getline(cin, NewRecipe.ingredients);

cout << "Введите вариант приготовления блюда, в конце нажмите Enter: ";

getline(cin, NewRecipe.instructions);

RecipeBook.push\_back(NewRecipe);

cout << "Рецепт успешно добавлен в конец списка!" << endl;

}

else if (a == 2) {

Recipe NewRecipe; // Для нового рецепта

int pos = 0; // Указанная позиция

int i = 1; // Для перебора итератором

auto iter = RecipeBook.cbegin(); // Итератор списка устанавливаем на первый элемент списка

cout << "\n Выбран режим добавления нового блюда в произвольное место списка" << endl;

do {

cout << "Укажите номер в списке, на который требуется поместить новый рецепт: ";

cin.ignore();

cin >> pos;

if (pos > RecipeBook.size())

cout << "Позиция находится за пределами списка! Попробуйте ещё раз" << endl;

} while (pos > RecipeBook.size());

cout << "Введите название блюда: ";

cin.ignore();

getline(cin, NewRecipe.name);

cout << "Введите ингридиенты для блюда через запятую: ";

getline(cin, NewRecipe.ingredients);

cout << "Введите вариант приготовления блюда, в конце нажмите Enter: ";

getline(cin, NewRecipe.instructions);

// Ищем нужную позицию в списке

while (i < pos) { // Как только i станет равно pos - итератор стоит на нужном месте

iter++; // Побежали итератором

i++; // Наращиваем счётчик

}

// Вставляем новый рецепт на нужное место

RecipeBook.insert(iter, NewRecipe);

cout << "Рецепт успешно добавлен в заданное место списка!" << endl;

}

else if (a == 3) {

string str3 = "";

Recipe NewRecipe;

int i = 0;

auto iter = RecipeBook.cbegin();

cout << "\n Выбран режим добавления после указанного на блюдо" << endl;

cout << "Введите НАЗВАНИЕ рецепта для поиска, новый рецпт будет добавлен ПОСЛЕ него: ";

cin.ignore();

getline(cin, str3);

for (auto iter = RecipeBook.begin(); iter != RecipeBook.end(); iter++)

{

if (iter->name == str3) {

iter++;

cout << "Введите название блюда: ";

cin.ignore();

getline(cin, NewRecipe.name);

cout << "Введите ингридиенты для блюда через запятую: ";

getline(cin, NewRecipe.ingredients);

cout << "Введите вариант приготовления блюда, в конце нажмите Enter: ";

getline(cin, NewRecipe.instructions);

RecipeBook.insert(iter, NewRecipe);

cout << "Рецепт успешно добавлен в заданное место списка!" << endl;

break;

}

}

}

else if (a == 4) {

string str3 = "";

Recipe NewRecipe;

int i = 0;

auto iter = RecipeBook.cbegin();

cout << "\n Выбран режим добавления до указанного на блюдо" << endl;

cout << "Введите НАЗВАНИЕ рецепта для поиска, новый рецпт будет добавлен ПЕРЕД ним: ";

cin.ignore();

getline(cin, str3);

for (auto iter = RecipeBook.begin(); iter != RecipeBook.end(); iter++)

{

if (iter->name == str3) {

iter--;

iter++;

cout << "Введите название блюда: ";

cin.ignore();

getline(cin, NewRecipe.name);

cout << "Введите ингридиенты для блюда через запятую: ";

getline(cin, NewRecipe.ingredients);

cout << "Введите вариант приготовления блюда, в конце нажмите Enter: ";

getline(cin, NewRecipe.instructions);

RecipeBook.insert(iter, NewRecipe);

cout << "Рецепт успешно добавлен в заданное место списка!" << endl;

break;

}

}

}

else if (a == 5) {

Recipe NewRecipe;

auto iter = RecipeBook.cbegin();

cout << "\n Выбран режим добавления в середину списка" << endl;

cout << "Введите название блюда: ";

cin.ignore();

getline(cin, NewRecipe.name);

cout << "Введите ингридиенты для блюда через запятую: ";

getline(cin, NewRecipe.ingredients);

cout << "Введите вариант приготовления блюда, в конце нажмите Enter: ";

getline(cin, NewRecipe.instructions);

for (int i = 1; i < ceil(RecipeBook.size() / 2.0); i++)

{

iter++;

}

iter++;

RecipeBook.insert(iter, NewRecipe);

}

else if (a == 6) {

cout << "Рецепт успешно удален из середины списка!" << endl;

if (RecipeBook.size() >= 4) {

auto iter = RecipeBook.cbegin();

for (int i = 1; i < ceil(RecipeBook.size() / 2.0); i++)

{

iter++;

}

RecipeBook.erase(iter);

}

else if (RecipeBook.size() < 4) {

auto iter = RecipeBook.cbegin();

for (int i = 1; i < ceil(RecipeBook.size() / 2.0); i++)

{

iter++;

}

RecipeBook.erase(iter);

}

}

else if (a == 7) {

cout << "Введите НАЗВАНИЕ рецепта для удаления: ";

string str3 = "";

auto iter = RecipeBook.cbegin();

cin.ignore();

getline(cin, str3);

for (int i = 1; i < RecipeBook.size(); i++)

{

if (iter->name == str3) {

break;

}

iter++;

}

RecipeBook.erase(iter);

cout << "Рецепт успешно удален из списка!" << endl;

}

else if (a == 8) {

cout << "Рецепт успешно удален из конца списка!" << endl;

auto iter = RecipeBook.cbegin();

for (int i = 1; i < RecipeBook.size(); i++) {

iter++;

}

RecipeBook.erase(iter);

}

else if (a == 9) {

int pos = 0; // Указанная позиция

int i = 1; // Для перебора итератором

auto iter = RecipeBook.cbegin(); // Итератор списка устанавливаем на первый элемент списка

cout << "Укажите номер в списке, который требуется удалить: ";

cin.ignore();

cin >> pos;

if (pos > RecipeBook.size())

cout << "Позиция находится за пределами списка! Попробуйте ещё раз" << endl;

while (i < pos) {

iter++; // Побежали итератором

i++; // Наращиваем счётчик

}

// Вставляем новый рецепт на нужное место

RecipeBook.erase(iter);

cout << "Рецепт успешно удален из списка!" << endl;

}

else if (a == 10) {

cout << "Рецепт успешно удален из головы списка!" << endl;

auto iter = RecipeBook.cbegin();

RecipeBook.erase(iter);

}

a = menu();

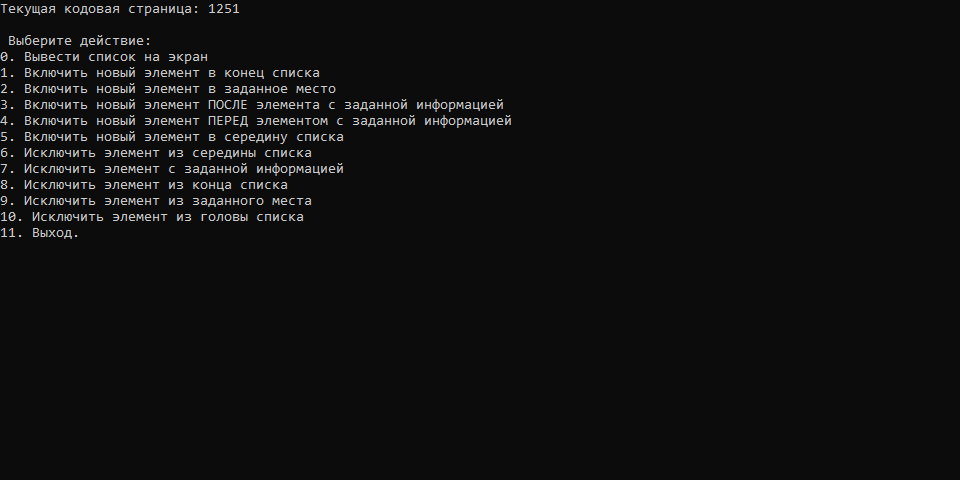
}

return 0;

}

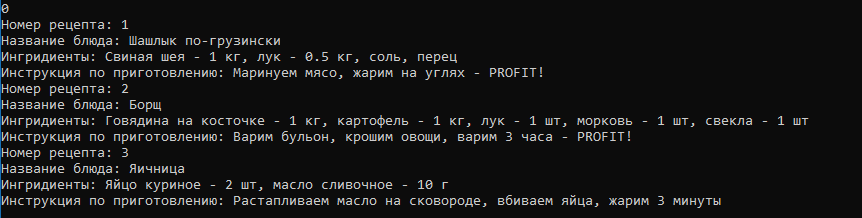
**Результат работы программы.**

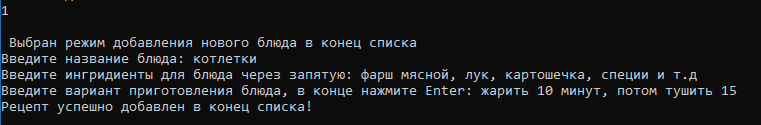
Входные данные:

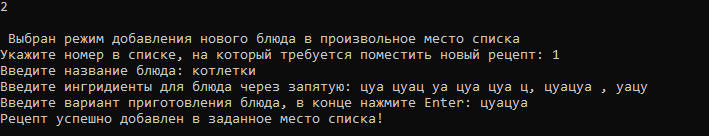


Выходные данные:

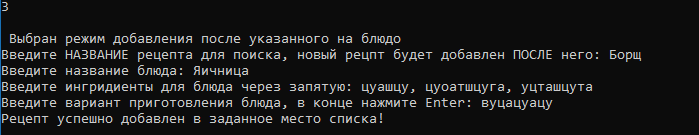
0:

1:

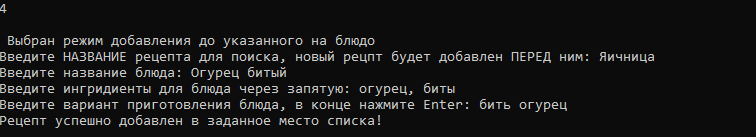


2:

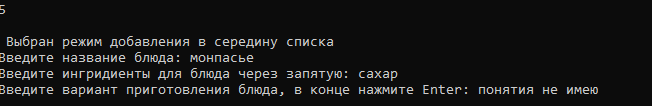
3:



4:



5:



6:

C:\Users\User\YandexDisk\Скриншоты\VsDebugConsole_Fdt3HUJqII.png

7:

C:\Users\User\YandexDisk\Скриншоты\VsDebugConsole_WbNwjYowdi.png

8:

C:\Users\User\YandexDisk\Скриншоты\VsDebugConsole_aZqqctuvZX.png

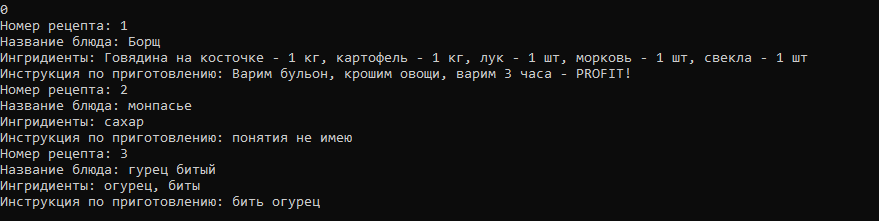
9:

C:\Users\User\YandexDisk\Скриншоты\VsDebugConsole_loH8Nm71Fh.png

10:

C:\Users\User\YandexDisk\Скриншоты\VsDebugConsole_i6cA65t2Hr.png

Итог:



**Вывод:**

В ходе работы была успешно создана программа, которая работает со списками. Были написаны различные функции, которые получают входные данные и уже потом обрабатывают их вместе со списками.