Министерство образования и науки Кыргызской Республики

Кыргызский государственный технический университет

им.И.Раззакова

Факультет информационных технологий

Кафедра «Программное обеспечение компьютерных систем»

Направление:710400 «Программная инженерия»

ОТЧЕТ

По дисциплине: «Объектно-ориентированное программирование»

Лабораторная работа №2

Тема: «Классы и объекты»

Выполнил: студент группы

ПИ(б)-2-19 Улан уулу Нурдин

Проверила: Каткова С. Н.

Бишкек – 2021

**Задание 1**

1. **Постановка задачи:**

В здании аэропорта на экранах отображается информация о самолетах, а именно: информация о пункте отправления, пункте назначения, номере рейса, времени прибытия, времени отправления, номере секции для регистрации. Экраны – это средство, которое помогает своевременно зарегистрировать и отправить пассажиров.  Важно, чтобы информация на экранах была понятной и верной.

Создайте необходимую информацию в виде таблицы для такого экрана, с помощью класса Aeroflot, содержащего в описании следующие поля: номер рейса; название пункта отправления; название пункта назначения рейса; время прибытия; время отправления; место регистрации.

Напишите код программы, выполняющей следующие действия: ввод с клавиатуры значений полей объектов; сортировку записей в таблице в алфавитном порядке по названию пунктов назначения; вывод на консоль значений полей объектов класса в виде таблицы рейсов; если таких рейсов нет, выдать соответствующее сообщение.

1. **Описание исходных данных:**
2. Структура info описывает информацию о самолетах и состоит из следующих полей:

* str departure – место отправление;
* str destination – место прибытия;
* int number\_flight - номер рейса;
* str date\_departure – дата отправления;
* str date\_destination – дата прибытия;
* int number\_registration – номер регистрации.

1. Класс Aeroflot описывает информацию о самолетах и состоит из следующих полей и методов:

Закрытые поля класса Aeroflot c модификатором private:

* int count – количество самолетов;
* info\* Arr – динамический массив структуры.

Открытые public методы класса Aeroflot:

* void setInfo(int n, info\* arr) – Сетер с параметрами массива объектов структуры;
* void setInfo(int n) – Сетер без параметров;
* void add() – добавление нового самолета;
* void output() – вывод самолетов;
* int SearchElem(str item) – метод для поиска;
* void search() – поиск рейса;
* void Sort() – сортировка самолетов по названию пунктов назначения;
* void Initial\_Sort() – сортировка в изначальное состояние;
* ~Aeroflot() – деструктор класса.

1. **Блок-схемы класса и методов:**









1. **Код программы:**

#include <iostream>

#include <ctime>

#include <fstream>

#include <string>

#include "Windows.h"

#include <time.h>

#include <iomanip>

#define ll long long

#define str string

using namespace std;

bool check\_number(str s) { //функция для проверки введенного символа

if (atoi(s.c\_str()) || s == "0")

return true;

else return false;

}

struct info {

str departure; //отправление

str destination;//место назначения

int number\_flight = 0;//номер рейса

str date\_departure;//дата отправления

str date\_destination;//дата посадки

int number\_registration = 0;//номер регистрации

};

class Aeroflot {

int count;

info\* Arr;

private:

int SearchElem(str item) {

int c = 0, s = -1, choice;

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

if (Arr[i].destination == item) {

c++;

s = i;

}

}

if (c == 0)

return s;

else if (c == 1)

return s;

else {

while (true) {

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

cout << item << "\t" << i + 1 << endl;

cout << "\nChoice one elem: ";

cin >> choice; system("cls");

if (choice > c || choice < 0)

cout << "Error!!!\n" << endl;

else {

s = choice - 1;

return s;

}

}

}

}

public:

void setInfo(int n, info\* arr) {

count = n;

Arr = new info[count]; //Выделение дин. памяти

for (int i = 0; i < count; i++)

Arr[i] = arr[i];//Copy всей информации из main в класс

}

void setInfo() {

count = 0;

Arr = new info[count]; //Выделение дин. памяти

}

void add() {//метод добавления

info\* temp = new info[count]; //создание третьего стакана

for (int i = 0; i < count; i++)

temp[i] = Arr[i]; //copy для добавления

count++; str s;

Arr = new info[count]; //увеличение выделяемой памяти для записи нового элемента

cout << "Введите место отправления:";

cin.ignore(); getline(cin, s);

Arr[count - 1].departure = s;

cout << "Введите место назначения:";

getline(cin, s);

Arr[count - 1].destination = s;

Arr[count - 1].number\_flight = count;

cout << "Введите дату вылета:";

getline(cin, s);

Arr[count - 1].date\_departure = s;

cout << "Введите дату прилета:";

getline(cin, s);

Arr[count - 1].date\_destination = s;

cout << "Введите номер регистрационной секции:";

cin >> s; Arr[count - 1].number\_registration = atoi(s.c\_str());

for (int i = 0; i < count - 1; i++)

Arr[i] = temp[i];

delete[]temp;

}

void output() { //метод вывода

cout << "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_" << endl;

cout << "| № fligth | Departure | Destination | Date departure | Date destination | № reg |" << endl;

cout << "|----------+-----------+-------------+------------------+------------------+-------+" << endl;

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

cout << "| " << setw(8) << Arr[i].number\_flight << " | " << setw(9) << Arr[i].departure;

cout << " | " << setw(11) << Arr[i].destination << " | ";

cout << setw(16) << Arr[i].date\_departure << " | " << setw(16) << Arr[i].date\_destination << " | ";

cout << setw(5) << Arr[i].number\_registration << " |" << endl;

cout << "|----------+-----------+-------------+------------------+------------------+-------+" << endl;

}

}

void sort() { //метод сортировки

for (int j = 1; j < count; j++)

for (int i = 0; i < count - j; i++)

if (Arr[i].destination > Arr[i + 1].destination)

swap(Arr[i], Arr[i + 1]);

}

void initial\_Sort() { //метод для возвращения обратного вида

for (int j = 1; j < count; j++)

for (int i = 0; i < count - j; i++)

if (Arr[i].number\_flight > Arr[i + 1].number\_flight)

swap(Arr[i], Arr[i + 1]);

}

void search() { //метод для поиска рейса по месту назначения

str s; int x;

cout << "Введите место назначение искоемого рейса: ";

cin >> s; system("cls");

x = SearchElem(s); // вызов метода из private секции

if (x != -1) {

cout << "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_" << endl;

cout << "| № fligth | Departure | Destination | Date departure | Date destination | № reg |" << endl;

cout << "|----------+-----------+-------------+------------------+------------------+-------+" << endl;

cout << "| " << setw(8) << Arr[x].number\_flight << " | " << setw(9) << Arr[x].departure;

cout << " | " << setw(11) << Arr[x].destination << " | ";

cout << setw(16) << Arr[x].date\_departure << " | " << setw(16) << Arr[x].date\_destination << " | ";

cout << setw(5) << Arr[x].number\_registration << " |" << endl;

cout << "|----------+-----------+-------------+------------------+------------------+-------+" << endl;

} else

cout << "Такого рейса нет!!!" << endl << endl;

}

~Aeroflot() { //деструктор класса

count = NULL;

delete[]Arr;//освобождение памяти

}

};

void main\_menu() {

cout << " ,---. ,---. " << endl;

cout << " / /^`.\\.--''''-./,'^\\ \\ " << endl;

cout << " \\ \\ \_ \_ / / " << "\t\t\t\t\t\tГлавное Меню:\t\t" << endl;

cout << " `./ / \_\_ \_\_ \\ \\,' " << endl;

cout << " / /\_O)\_(\_O\\ \\ " << "\t\t\t\t\t1. Добавить новую запись\t" << endl;

cout << " | .-' \_\_\_ `-. | " << endl;

cout << " .--| \\\_/ |--. " << "\t\t\t\t\t2. Вывести записи полетов\t" << endl;

cout << " ,' \\ \\ | / / `." << endl;

cout << " / `. `--^--' ,' \\" << "\t\t\t\t\t\t3. Отсортировать записи полетов\t\t" << endl;

cout << " .-^^^^^-. `--.\_\_\_.--' .-^^^^^-." << endl;

cout << ".-----------/ \\------------------/ \\--------------." << "\t\t\t4. Вернуть в первоначальный вид\t\t\t" << endl;

cout << "| .---------\\ /----------------- \\ /------------. |" << endl;

cout << "| | `-`--`--' `--'--'-' | |" << "\t\t\t5. Поиск записи полета\t\t\t" << endl;

cout << "| |\_\_ \_\_ \_\_\_ | | \_\_\_ \_\_\_ \_ \_\_ \_\_\_ \_\_\_ | |" << endl;

cout << "| |\\ \\ /\\ / / / \_ \\ | | / \_\_| / \_ \\ | '\_ ` \_ \\ / \_ \\ | |" << "\t\t\t6. Выход" << endl;

cout << "| | \\ V V / | \_\_/ | | | (\_\_ | (\_) | | | | | | | | \_\_/ | |" << endl;

cout << "| | \\\_/\\\_/ \\\_\_\_| |\_| \\\_\_\_| \\\_\_\_/ |\_| |\_| |\_| \\\_\_\_| | |" << endl;

cout << "| |\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_| |" << endl;

cout << "|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|" << endl;

cout << " )\_\_\_\_\_\_\_\_\_\_|\_\_|\_\_\_\_\_\_\_\_\_\_(" << endl;

cout << " | || |" << endl;

cout << " |\_\_\_\_\_\_\_\_\_\_\_\_||\_\_\_\_\_\_\_\_\_\_\_\_|" << endl;

cout << " ),-----.( ),-----.(" << endl;

cout << " ,' ==. \\ / .== `." << endl;

cout << " / ) ( \\" << endl;

cout << " `===========' `===========' " << endl;

}

int main() {

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

time\_t start, end;

double dif;

str s; bool f = true;

Aeroflot A;

start = clock();

cout << "Читать данные из файла?\n1 - Yes\t2 - No\n\n";

cin >> s;

system("cls");

while (f) {

if (!check\_number(s))

cout << "Вы ввели не число!!!" << endl;

else {

f = false;

if (atoi(s.c\_str()) == 1) {

fstream sin; int n;

sin.open("1\\air.txt"); //открытие файла

sin >> n;

info\* arr = new info[n]; //создание и выделение динамического массива объекта структуры

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

sin >> arr[i].number\_flight;

sin.ignore(); getline(sin, arr[i].departure); getline(sin, arr[i].destination);

sin >> arr[i].number\_registration;

sin.ignore(); getline(sin, arr[i].date\_departure); getline(sin, arr[i].date\_destination);

}

sin.close(); //закрытие файла

A.setInfo(n, arr); //передача данных в класс

delete[]arr; //освобождение памяти

} else if (atoi(s.c\_str()) == 2)

A.setInfo(); //передача данных в класс

}

}

f = true;

while (f) {

main\_menu();

cin >> s;

system("cls");

if (!check\_number(s))

cout << "Вы ввели не число!!!" << endl;

else {

switch (atoi(s.c\_str())) {

case 1: A.add(); break;

case 2: A.output(); break;

case 3: A.sort(); break;

case 4: A.initial\_Sort(); break;

case 5: A.search(); break;

case 6: f = false; break;

default: cout << "Такой операции не существует!!!" << endl << "Введите заново." << endl; break;

}

}

}

system("cls"); end = clock();

dif = difftime(end, start);

cout << "Программа работала " << (dif / 1000.0) << " секунд\n";

return 0;

}

1. **Результаты работы программы:**

Вывод:



Добавление:



Сортировка по названию пунктов назначения (Destination):



Вернуть в первоначальный вид:



**Задание 2**

1. **Постановка задачи:**

Предусмотрите возможность: работы с переменным числом студентов; поиска студента по какому-либо признаку (например, по фамилии, дате рождения или номеру телефона); добавления или удаления записей; сортировки по разным полям.

Разработайте программу, демонстрирующую работу с этим классом.

Программа должна содержать меню, позволяющее осуществить проверку всех методов класса.

1. **Описание исходных данных:**
2. Структура info описывает информацию о студентах и состоит из следующих полей:

* str full\_name – ФИО студента;

Так же внутри info есть структура date которая состоит из следующих полей:

* + str day – день рождение;
  + str month – месяц рождение;
  + str year – год рождение;
* date date\_b – дата рождения студента;
* str number\_stud – номер студента;

1. Класс Stud описывает информацию о студентах и состоит из следующих полей и методов:

Закрытые поля и методы класса Stud c модификатором private:

* int n – количество студентов;
* info\* variable\_class – динамический массив структуры.
* void pop(int x) – метод для удаление студента;
* int SearchElem(str item) – метод для поиска студента;
* void outputSearch(int x) – метод для вывода искомого студента;
* void BubbleSort(int check) – метод для сортировки студентов;
* void Choice(str s) – метод для движения в меню выбора.

Открытые public методы класса Stud:

* Stud(int count, info\* x) – конструктор с параметрами;
* void add() – метод добавления;
* void output() – метод вывода;
* void del() – метод удаления;
* void sort() – метод сортировки;
* void search() – метод поиска;
* ~Stud() – деструктор класса.

1. **Блок-схемы:**











1. **Код программы:**

#include <iostream>

#include <ctime>

#include <fstream>

#include <string>

#include "Windows.h"

#include <time.h>

#include <iomanip>

#define ll long long

#define str string

using namespace std;

struct info {

str full\_name;

struct date { //вложенная структура для реализации дня:месяца:года

str day;

str month;

str year;

} date\_b; //объект структуры

str number\_stud;

};

bool check\_number(str s) { //функция для проверки ввода числа

if (atoi(s.c\_str()) || s == "0")

return true;

else return false;

}

class Stud {

info\* variable\_class; //динамический массив объекта структуры, как поле класса

int n; // кол - во элементов объекта структуры

private:

void pop(int x) { //метод для удаления

variable\_class[x].full\_name = ' '; // удаление имени для дальнейшего перетаскивания в конец

for (int j = 1; j < n; j++)

for (int i = 0; i < n - j; i++)

if (variable\_class[i].full\_name == " ")

swap(variable\_class[i], variable\_class[i + 1]); //перетаскивание в конец

n--;

ofstream sout;

sout.open("1\\Stud.txt");

sout << n << endl;

for (int i = 0; i < n; i++) { //перезаписывание в файл новых данных

sout << variable\_class[i].full\_name << endl;

sout << variable\_class[i].date\_b.day << ' ' << variable\_class[i].date\_b.month << ' ' << variable\_class[i].date\_b.year << endl;

sout << variable\_class[i].number\_stud << endl;

}

sout.close();

}

int SearchElem(str item) { //метод для поиска

int count = 0, s = -1, choice;

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

if (variable\_class[i].full\_name == item) { //поиск по ФИО

count++;

s = i;

}

else if ((variable\_class[i].date\_b.day[0] == item[0] && variable\_class[i].date\_b.day[1] == item[1])

&& (variable\_class[i].date\_b.month[0] == item[3] && variable\_class[i].date\_b.month[1] == item[4])

&& (variable\_class[i].date\_b.year[0] == item[6] && variable\_class[i].date\_b.year[1] == item[7]

&& (variable\_class[i].date\_b.year[2] == item[8] && variable\_class[i].date\_b.year[3] == item[9]))) {//поиск по дате рождения

count++;

s = i;

}

else if (variable\_class[i].number\_stud == item) { //поиск по номеру телефона

count++;

s = i;

}

}

if (count == 0) //проверка есть ли искоемый студент

return s; //если нет, возвращается -1

else if (count == 1)

return s;//если да, возвращается индекс

else { //если их больше 1, предлагается выбор

while (true) {

for (int i = 0; i < count; i++)

cout << item << "\t" << i + 1 << endl;

cout << "\nChoice one elem: ";

cin >> choice; //выбор нужного

system("cls");

if (choice > count || choice < 0) //проверка на ввод

cout << "Error!!!\n" << endl;

else {

s = choice - 1;

return s;

}

}

}

}

void outputSearch(int x) { //вывод искоемого студента

cout << "+ № student + Full Name + Date of Birthday + Number Student +" << endl;

cout << "|-----------|----------------------|------------------|----------------|" << endl;

cout << "| " << setw(9) << x + 1 << " | " << setw(20) << variable\_class[x].full\_name << " | ";

cout << setw(8) << variable\_class[x].date\_b.day << '-' << variable\_class[x].date\_b.month << '-' << variable\_class[x].date\_b.year;

cout << " | " << setw(14) << variable\_class[x].number\_stud << " |" << endl;

}

void BubbleSort(int check) { //метод сортировки

if (check == 1) {

fstream sin;

sin.open("1\\stud.txt");

sin >> n;

for (int i = 0; i < n; i++) { //возврат в первоначальный вид

sin.ignore(); getline(sin, variable\_class[i].full\_name);

sin >> variable\_class[i].date\_b.day >> variable\_class[i].date\_b.month >> variable\_class[i].date\_b.year;

sin >> variable\_class[i].number\_stud;

}

sin.close();

}

else if (check == 2) { //сортировка по ФИО

for (int j = 1; j < n; j++)

for (int i = 0; i < n - j; i++)

if (variable\_class[i].full\_name > variable\_class[i + 1].full\_name)

swap(variable\_class[i], variable\_class[i + 1]);

}

else if (check == 3) { //сортировка по дате рождения

for (int j = 0; j < n - 1; j++)

for (int i = 0; i < n - 1; i++)

if ((atoi(variable\_class[i].date\_b.day.c\_str()) + (atoi(variable\_class[i].date\_b.month.c\_str()) \* 100)

+ (atoi(variable\_class[i].date\_b.year.c\_str()) \* 10000)) > (atoi(variable\_class[i + 1].date\_b.day.c\_str())

+ (atoi(variable\_class[i + 1].date\_b.month.c\_str()) \* 100) + (atoi(variable\_class[i + 1].date\_b.year.c\_str()) \* 10000))) //здесь я умножаю дни месяцы и годы и сравниваю кто больше живет

swap(variable\_class[i], variable\_class[i + 1]);

}

else //сортировка по номеру телефона

for (int j = 1; j < n; j++)

for (int i = 0; i < n - j; i++)

if (atoi(variable\_class[i].number\_stud.c\_str()) > atoi(variable\_class[i + 1].number\_stud.c\_str()))

swap(variable\_class[i], variable\_class[i + 1]);

}

void Choice(str s) { //меню выбора

cout << " \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_" << endl;

cout << " / \\ " << setw(10) << s << " студентов \\." << endl;

cout << "| | |." << endl;

cout << " \\\_ | 1. По номеру в списке |." << endl;

cout << " | 2. По ФИО |." << endl;

cout << " | 3. По дате рождения |." << endl;

cout << " | 4. По номеру телефона |." << endl;

cout << " | 5. Назад |." << endl;

cout << " | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|\_\_\_" << endl;

cout << " | / /." << endl;

cout << " \\\_/\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/." << endl;

}

public:

Stud(int count, info\* x) { //конструктор с параметрами

this->n = count;

this->variable\_class = new info[n]; //выделение дин. памяти

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

this->variable\_class[i] = x[i]; //передача данных в переменную класса

}

}

void add() {

info\* temp = new info[n]; //создание резеруара для временного хранения

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

temp[i] = variable\_class[i]; //передача данных в резеруар

}

n++;

variable\_class = new info[n]; //переопределения занимаего места

for (int i = 0; i < n - 1; i++) {

variable\_class[i] = temp[i];//передача данных обратно

}

cout << "Введите ФИО нового студента:";

cin.ignore(); getline(cin, variable\_class[n - 1].full\_name);

cout << "Введите дату рождения нового студента:";

cin >> variable\_class[n - 1].date\_b.day >> variable\_class[n - 1].date\_b.month >> variable\_class[n - 1].date\_b.year;

cout << "Введите номер нового студента:";

cin >> variable\_class[n - 1].number\_stud;

ofstream sout;

sout.open("1\\Stud.txt");

sout << n << endl;

for (int i = 0; i < n; i++) { //запись обратно в файл

sout << variable\_class[i].full\_name << endl;

sout << variable\_class[i].date\_b.day << ' ' << variable\_class[i].date\_b.month << ' ' << variable\_class[i].date\_b.year << endl;

sout << variable\_class[i].number\_stud << endl;

}

sout.close();

delete[]temp; //освобождения памяти

}

void output() { //метод вывода

cout << "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_" << endl;

cout << "| № student | Full Name | Date of Birthday | Number Student |" << endl;

cout << "|-----------|----------------------|------------------|----------------|" << endl;

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

cout << "| " << setw(9) << i + 1 << " | " << setw(20) << variable\_class[i].full\_name << " | ";

cout << setw(8) << variable\_class[i].date\_b.day << '-' << variable\_class[i].date\_b.month << '-' << variable\_class[i].date\_b.year;

cout << " | " << setw(14) << variable\_class[i].number\_stud << " |" << endl;

cout << "|-----------|----------------------|------------------|----------------|" << endl;

}

}

void del() { //метод удаления

int x = 0;

str s; bool f = true;

while (f) {

Choice("Удаление");

cout << "\nВведите признак удаления:";

cin >> s; system("cls");

if (!check\_number(s)) //проверка на ввод числа

cout << "Вы ввели не число!!!" << endl;

else {

switch (atoi(s.c\_str())) { //конверт в int

case 1: f = false;

cout << "Введите номер студента из списка для удаления:";

cin >> s; system("cls");

if (!check\_number(s))

cout << "Вы ввели не число!!!" << endl;

else {

if ((atoi(s.c\_str()) - 1) >= n || (atoi(s.c\_str()) - 1) < 0) //проверка на коректность ввода

cout << "Вы ввели неправильно, повторите ввод!!!" << endl;

else pop(atoi(s.c\_str()) - 1); //вызов метода из private секции

}; break;

case 2: f = false;

cout << "Введите ФИО студента из списка:";

cin.ignore(); getline(cin, s); system("cls"); x = SearchElem(s);//вызов метода из private секции

if (x == -1)

cout << "Такого студента нет в базе данных!!!" << endl << endl;

else pop(x);//вызов метода из private секции break;

case 3: f = false;

cout << "Введите дату рождения студента из списка:";

cin >> s; system("cls"); x = SearchElem(s); //вызов метода из private секции

if (x == -1)

cout << "Такого студента нет в базе данных!!!" << endl << endl;

else pop(x); //вызов метода из private секции break;

case 4: f = false;

cout << "Введите номер телефона студента из списка:";

cin >> s; system("cls"); x = SearchElem(s);//вызов метода из private секции

if (x == -1)

cout << "Такого студента нет в базе данных!!!" << endl << endl;

else pop(x); //вызов метода из private секции break;

case 5: f = false; break;

default: cout << "Такой операции не существует!!!" << endl << "Введите заново." << endl; break;

}

}

}

}

void sort() {

str s;

bool f = true;

while (f) {

Choice("Сортировка");

cout << endl;

cout << "Введите признак сортировки:";

cin >> s;

system("cls");

if (!check\_number(s)) //проверка на ввод

cout << "Вы ввели не число!!!" << endl;

else {

switch (atoi(s.c\_str())) { //конверт в int

case 1: f = false; BubbleSort(1); break;

case 2: f = false; BubbleSort(2); break;

case 3: f = false; BubbleSort(3); break;

case 4: f = false; BubbleSort(4); break;

case 5: f = false; break;

default: cout << "Такой операции не существует!!!" << endl << "Введите заново." << endl; break;

}

}

}

}

void search() {

int x = -1; str s;

bool f = true;

while (f) {

Choice("Поиск");

cout << endl;

cout << "Введите признак поиска:";

cin >> s;

system("cls");

if (!check\_number(s)) //проверка на ввод

cout << "Вы ввели не число!!!" << endl;

else {

switch (atoi(s.c\_str())) {

case 1: f = false;

cout << "Введите номер студента из списка:";

cin >> s; system("cls");

if (!check\_number(s))//проверка на ввод

cout << "Вы ввели не число!!!" << endl;

else {

if ((atoi(s.c\_str()) - 1) >= n || (atoi(s.c\_str()) - 1) < 0)//проверка на ввод

cout << "Такого студента нет в базе данных!!!" << endl << endl;

else outputSearch(atoi(s.c\_str()) - 1); //вызов метода из private секции

}; break;

case 2: f = false;

cout << "Введите ФИО студента из списка:";

cin.ignore(); getline(cin, s); system("cls"); x = SearchElem(s);

if (x != -1)

outputSearch(x);//вызов метода из private секции

else cout << "Такого студента нет в базе данных!!!" << endl << endl; break;

case 3: f = false;

cout << "Введите дату рождения студента из списка:";

cin >> s; system("cls"); x = SearchElem(s);

if (x != -1)

outputSearch(x);//вызов метода из private секции

else cout << "Такого студента нет в базе данных!!!" << endl << endl; break;

case 4: f = false;

cout << "Введите номер телефона студента из списка:";

cin >> s; system("cls"); x = SearchElem(s);

if (x != -1)

outputSearch(x);//вызов метода из private секции

else cout << "Такого студента нет в базе данных!!!" << endl << endl; break;

case 5: f = false; break;

default: cout << "Такой операции не существует!!!" << endl << "Введите заново." << endl; break;

}

}

}

}

~Stud() {

n = NULL;

delete[]variable\_class; //освобождение памяти

}

};

void main\_menu() {

cout << " ,---. ,---. " << endl;

cout << " / /^`.\\.--''''-./,'^\\ \\ " << endl;

cout << " \\ \\ \_ \_ / / " << "\t\t\t\t\t\tГлавное Меню:" << endl;

cout << " `./ / \_\_ \_\_ \\ \\,' " << endl;

cout << " / /\_O)\_(\_O\\ \\ " << "\t\t\t\t\t1. Добавить нового студента" << endl;

cout << " | .-' \_\_\_ `-. | " << endl;

cout << " .--| \\\_/ |--. " << "\t\t\t\t\t2. Вывести список студентов" << endl;

cout << " ,' \\ \\ | / / `." << endl;

cout << " / `. `--^--' ,' \\" << "\t\t\t\t\t\t3. Удалить студента" << endl;

cout << " .-^^^^^-. `--.\_\_\_.--' .-^^^^^-." << endl;

cout << ".-----------/ \\------------------/ \\--------------." << "\t\t\t4. Отсортировать список студентов" << endl;

cout << "| .---------\\ /----------------- \\ /------------. |" << endl;

cout << "| | `-`--`--' `--'--'-' | |" << "\t\t\t5. Поиск студента" << endl;

cout << "| |\_\_ \_\_ \_\_\_ | | \_\_\_ \_\_\_ \_ \_\_ \_\_\_ \_\_\_ | |" << endl;

cout << "| |\\ \\ /\\ / / / \_ \\ | | / \_\_| / \_ \\ | '\_ ` \_ \\ / \_ \\ | |" << "\t\t\t6. Выход" << endl;

cout << "| | \\ V V / | \_\_/ | | | (\_\_ | (\_) | | | | | | | | \_\_/ | |" << endl;

cout << "| | \\\_/\\\_/ \\\_\_\_| |\_| \\\_\_\_| \\\_\_\_/ |\_| |\_| |\_| \\\_\_\_| | |" << endl;

cout << "| |\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_| |" << endl;

cout << "|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|" << endl;

cout << " )\_\_\_\_\_\_\_\_\_\_|\_\_|\_\_\_\_\_\_\_\_\_\_(" << endl;

cout << " | || |" << endl;

cout << " |\_\_\_\_\_\_\_\_\_\_\_\_||\_\_\_\_\_\_\_\_\_\_\_\_|" << endl;

cout << " ),-----.( ),-----.(" << endl;

cout << " ,' ==. \\ / .== `." << endl;

cout << " / ) ( \\" << endl;

cout << " `===========' `===========' " << endl;

}

int main() {

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

int n, x; str s;

fstream sin; bool f = true;

sin.open("1\\Stud.txt");

sin >> n;

info\* variable\_main = new info[n]; //выделение памяти для объекта структуры

for (int i = 0; i < n; i++) { //заполнение полей объекта структуры

sin.ignore(); getline(sin, variable\_main[i].full\_name);

sin >> variable\_main[i].date\_b.day >> variable\_main[i].date\_b.month >> variable\_main[i].date\_b.year;

sin >> variable\_main[i].number\_stud;

}

sin.close();

Stud S(n, variable\_main); //создание объекта класса и передача данных через конструктор

delete[]variable\_main; //освобождение памяти

while (f) {

main\_menu();

cin >> s;

system("cls");

if (!check\_number(s))

cout << "Вы ввели не число!!!" << endl;

else {

x = atoi(s.c\_str());

switch (x) {

case 1: S.add(); break;

case 2: S.output(); break;

case 3: S.del(); break;

case 4: S.sort(); break;

case 5: S.search(); break;

case 6: f = false; break;

default: cout << "Такой операции не существует!!!" << endl << "Введите заново." << endl; break;

}

}

}

return 0;

}

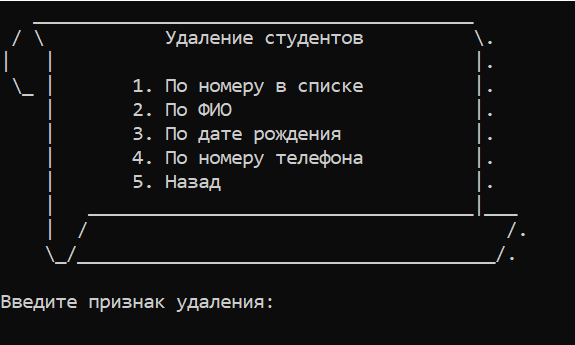
1. **Результаты работы программы:**

Вывод:

****

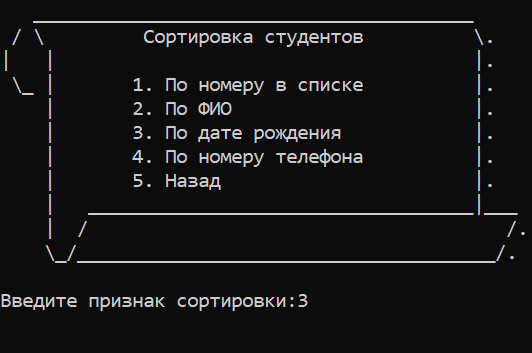
Добавление:  

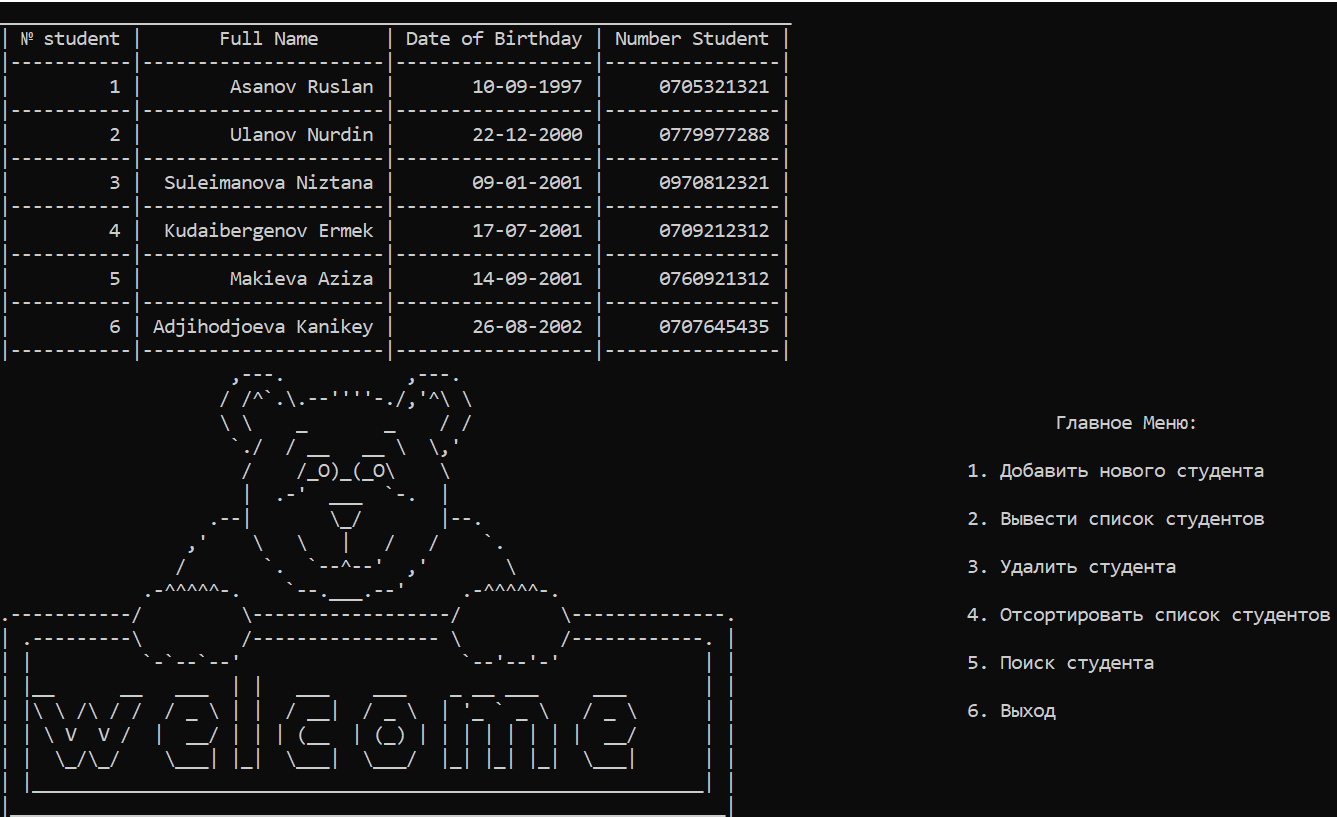

Удаление:



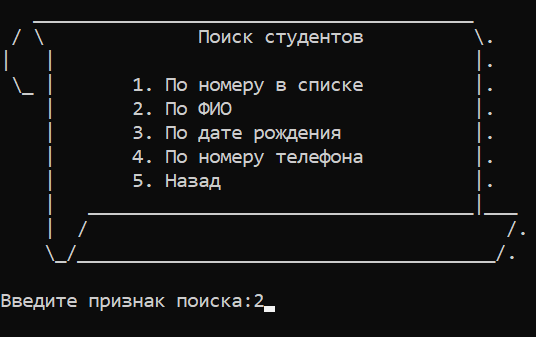


Сортировка:

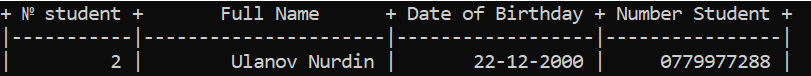


****

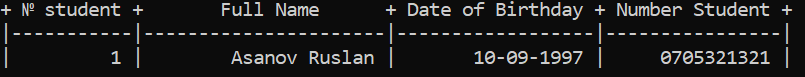
Поиск:







****

****

**Задание 3**

1. **Постановка задачи:**

Разработать программу, использующую этот класс для моделирования работы Т-образного сортировочного узла на железной дороге. Ваша программа должна имитировать работу диспетчера. Она должна делить главный состав, состоящий из вагонов двух типов, на два состава, и отправлять их по разным направлениям. Каждый состав должен состоять из вагонов одного типа. Рассмотрите случаи заполнения полей класса из файла или с клавиатуры. Вы можете использовать структуру данных «контейнер» и стандартные функции **push()** and **pop()** из библиотеки шаблонов STL.

1. **Описание исходных данных:**
2. Структура stack\_info описывает информацию о односвязном списке и состоит из следующих полей:

* int item – элемент стека;
* int id – уникальный ключ каждого вагона;
* stack\_info\* next – указатель на следующий элемент;

1. Класс Stack\_class описывает информацию о стеке и состоит из следующих полей и методов:

Закрытые поля класса Stack\_class c модификатором private:

* int count – количество вагонов главном составе;
* stack\_info\* S – объект структуры.
* void d(int& k1, int& k2) – метод подсчета типов вагона;

Открытые public методы класса Stack\_class:

* Stack\_class() – Конструктор по умолчанию;
* void push(int s) – добавление нового вагона;
* void output() – вывод вагонов;
* void pop() – удаление вагона;
* int getCount() – получение количества вагонов;
* void divide() – разделение вагонов;
* ~Stack\_class() – деструктор класса.

1. **Блок-схемы:**



1. **Код программы:**

#include <iostream>

#include <ctime>

#include <fstream>

#include <string>

#include "Windows.h"

#include <time.h>

#include <iomanip>

#define ll long long

#define str string

using namespace std;

bool check\_number(str s) { //функция для проверки ввода

if (atoi(s.c\_str()) || s == "0")

return true;

else return false;

}

struct stack\_info { //структура

int item;

stack\_info\* next; //указатель на следующий элемент

};

class Stack\_class {

stack\_info\* S; //создание объекта структуры

int count;

private:

void d(int& k1, int& k2) { //закрытый метод для подсчета кол - во 1-го и 2-го типа

stack\_info\* pv = S;

while (pv != NULL) {

if (pv->item % 2 == 0)

k1++;

else k2++;

pv = pv->next;

}

}

public:

Stack\_class() { //конструктор без параметров

S = NULL;

count = 0;

}

void push(int s) { //метод для добавления

stack\_info\* pv = new stack\_info; //создание следующего элемента списка

pv->item = s;

pv->next = S;

S = pv;

count++;

}

void pop() { //метод удаления

if (count == 0)

cout << "Стек пуст!!!" << endl << "Нечего удалять!!!" << endl;

else {

stack\_info\* pv = S;

S = S->next;

cout << "Элемент " << pv->item << " удален!!!" << endl;

delete pv;

count--;

}

}

void output() { //метод вывода

if (count == 0)

cout << "Стек пуст!!!" << endl << "Нечего показывать!!!" << endl;

else {

stack\_info\* pv = S;

int i = count;

cout << "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_" << endl;

cout << "| Элементы | Индекс |" << endl;

while (pv != NULL) {

cout << "| " << setw(8) << pv->item << " | " << setw(6) << i << " |" << endl;

pv = pv->next;

i--;

}

}

}

int getCount() { //Гетер для получения числа вагонов

return count;

}

void divide() { //разделение вагонов

if (count == 0)

cout << "Стек пуст!!!" << endl << "Нечего разделять!!!" << endl;

else {

int countE = 0, countO = 0;

int i = 0, j = 0, min, max;

d(countE, countO);

int\* arrEven = new int[countE + 1];

int\* arrOdd = new int[countO + 1];

stack\_info\* pv = S;

while (pv != NULL) {

if (pv->item % 2 == 0) {

arrEven[i] = pv->item;

i++;

}

else {

arrOdd[j] = pv->item;

j++;

}

pv = pv->next;

}

cout << "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_" << endl;

cout << "| Четные | Нечетные |" << endl;

if (countE > countO) {

max = countE;

min = countO;

}

else {

min = countE;

max = countO;

}

for (int i = 0; i < min; i++)

cout << "| " << setw(6) << arrEven[i] << " | " << setw(8) << arrOdd[i] << " |" << endl;

if (max == countE)

for (int i = min; i < max; i++)

cout << "| " << setw(6) << arrEven[i] << endl;

else

for (int i = min; i < max; i++)

cout << "| " << setw(9) << " | " << setw(8) << arrOdd[i] << " |" << endl;

delete[]arrEven;

delete[]arrOdd;

}

}

~Stack\_class() {

delete S; //освобождение памяти

count = NULL;

}

};

void main\_menu() {

cout << " ,---. ,---. " << endl;

cout << " / /^`.\\.--''''-./,'^\\ \\ " << endl;

cout << " \\ \\ \_ \_ / / " << "\t\t\t\t\t\tГлавное Меню:\t\t" << endl;

cout << " `./ / \_\_ \_\_ \\ \\,' " << endl;

cout << " / /\_O)\_(\_O\\ \\ " << "\t\t\t\t\t1. Добавить новый элемент в stack\t" << endl;

cout << " | .-' \_\_\_ `-. | " << endl;

cout << " .--| \\\_/ |--. " << "\t\t\t\t\t2. Вывести элементы stack'a\t" << endl;

cout << " ,' \\ \\ | / / `." << endl;

cout << " / `. `--^--' ,' \\" << "\t\t\t\t\t\t3. Удалить элемент stack'a\t\t" << endl;

cout << " .-^^^^^-. `--.\_\_\_.--' .-^^^^^-." << endl;

cout << ".-----------/ \\------------------/ \\--------------." << "\t\t\t4. Вывести кол - во элементов stack'a\t\t\t" << endl;

cout << "| .---------\\ /----------------- \\ /------------. |" << endl;

cout << "| | `-`--`--' `--'--'-' | |" << "\t\t\t5. Разделить и вывести stack\t\t\t" << endl;

cout << "| |\_\_ \_\_ \_\_\_ | | \_\_\_ \_\_\_ \_ \_\_ \_\_\_ \_\_\_ | |" << endl;

cout << "| |\\ \\ /\\ / / / \_ \\ | | / \_\_| / \_ \\ | '\_ ` \_ \\ / \_ \\ | |" << "\t\t\t6. Выход" << endl;

cout << "| | \\ V V / | \_\_/ | | | (\_\_ | (\_) | | | | | | | | \_\_/ | |" << endl;

cout << "| | \\\_/\\\_/ \\\_\_\_| |\_| \\\_\_\_| \\\_\_\_/ |\_| |\_| |\_| \\\_\_\_| | |" << endl;

cout << "| |\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_| |" << endl;

cout << "|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|" << endl;

cout << " )\_\_\_\_\_\_\_\_\_\_|\_\_|\_\_\_\_\_\_\_\_\_\_(" << endl;

cout << " | || |" << endl;

cout << " |\_\_\_\_\_\_\_\_\_\_\_\_||\_\_\_\_\_\_\_\_\_\_\_\_|" << endl;

cout << " ),-----.( ),-----.(" << endl;

cout << " ,' ==. \\ / .== `." << endl;

cout << " / ) ( \\" << endl;

cout << " `===========' `===========' " << endl;

}

int main() {

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

Stack\_class S;

str s; int x, n; bool f = true;

while (f) {

cout << "Хотите прочитать с файла данные?\n 1 - yes\t 2 - no" << endl;

cin >> s; system("cls");

if (!check\_number(s))//проверка на ввод числа

cout << "Вы ввели не число!!!" << endl;

else {

x = atoi(s.c\_str()); // конверт из динамического массива char в int

if (x == 1) {

f = false; fstream sin;

sin.open("1\\Stack.txt"); //открытие файла

sin >> n;

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

sin >> x; S.push(x); //добавление элементов в класс

}

sin.close(); //закрытие файла

}

else if (x == 2)

f = false;

}

}

f = true;

while (f) {

main\_menu();

cin >> s; system("cls");

if (!check\_number(s))//проверка на ввод числа

cout << "Вы ввели не число!!!" << endl;

else {

x = atoi(s.c\_str());// конверт из динамического массива char в int

switch (x) {

case 1:

cout << "Введите новый элемент:";

cin >> s;

system("cls");

if (!check\_number(s))//проверка на ввод числа

cout << "Вы ввели не число!!!" << endl;

else

S.push(atoi(s.c\_str())); //вызов метода

break;

case 2: S.output(); break;//вызов метода

case 3: S.pop(); break;//вызов метода

case 4: cout << "В стеке: " << S.getCount() << " элементов" << endl; break;//вызов метода

case 5: S.divide(); break;//вызов метода

case 6: f = false; break;

default: cout << "Такой операции не существует!!!" << endl << "Введите заново." << endl; break;

}

}

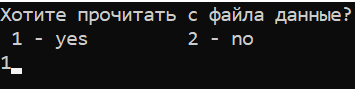
}

return 0;

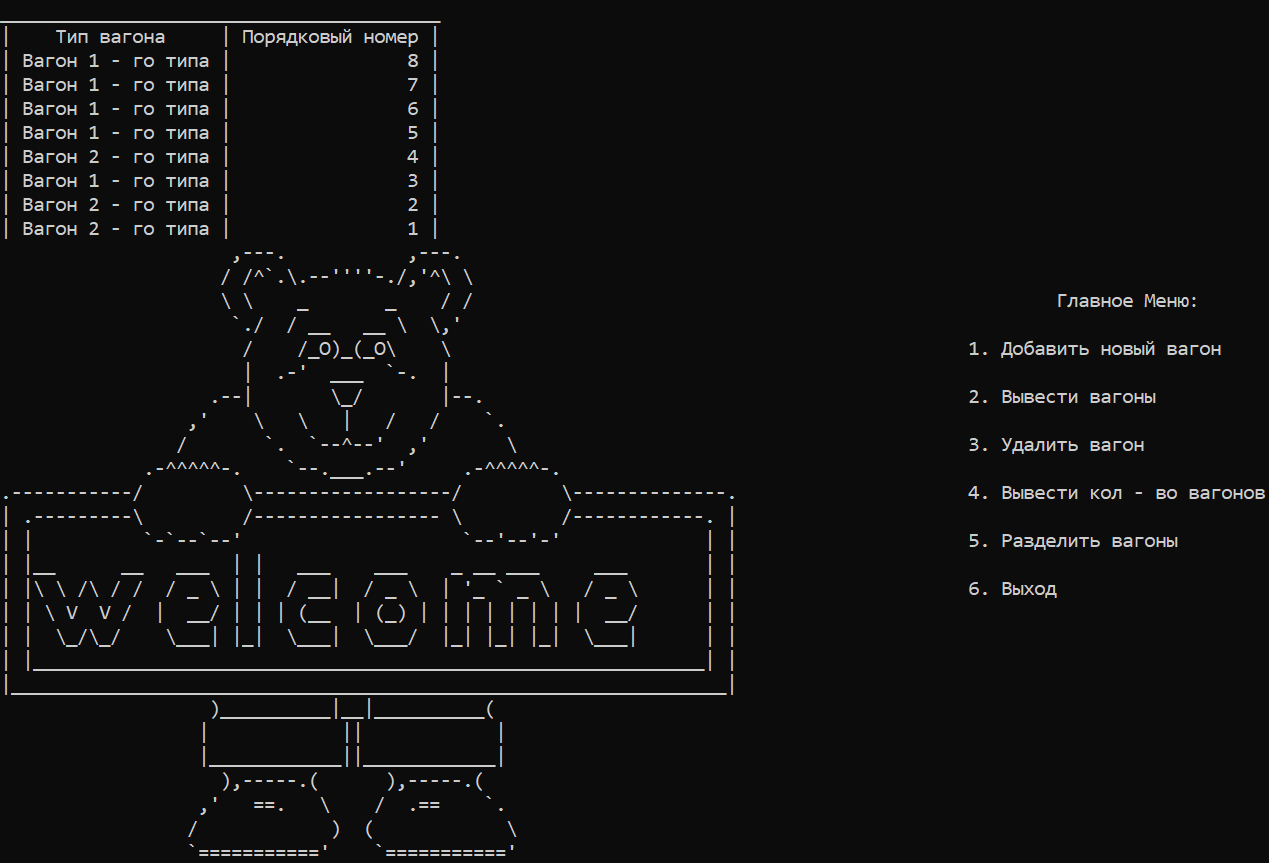
}

1. **Результаты работы программы:**

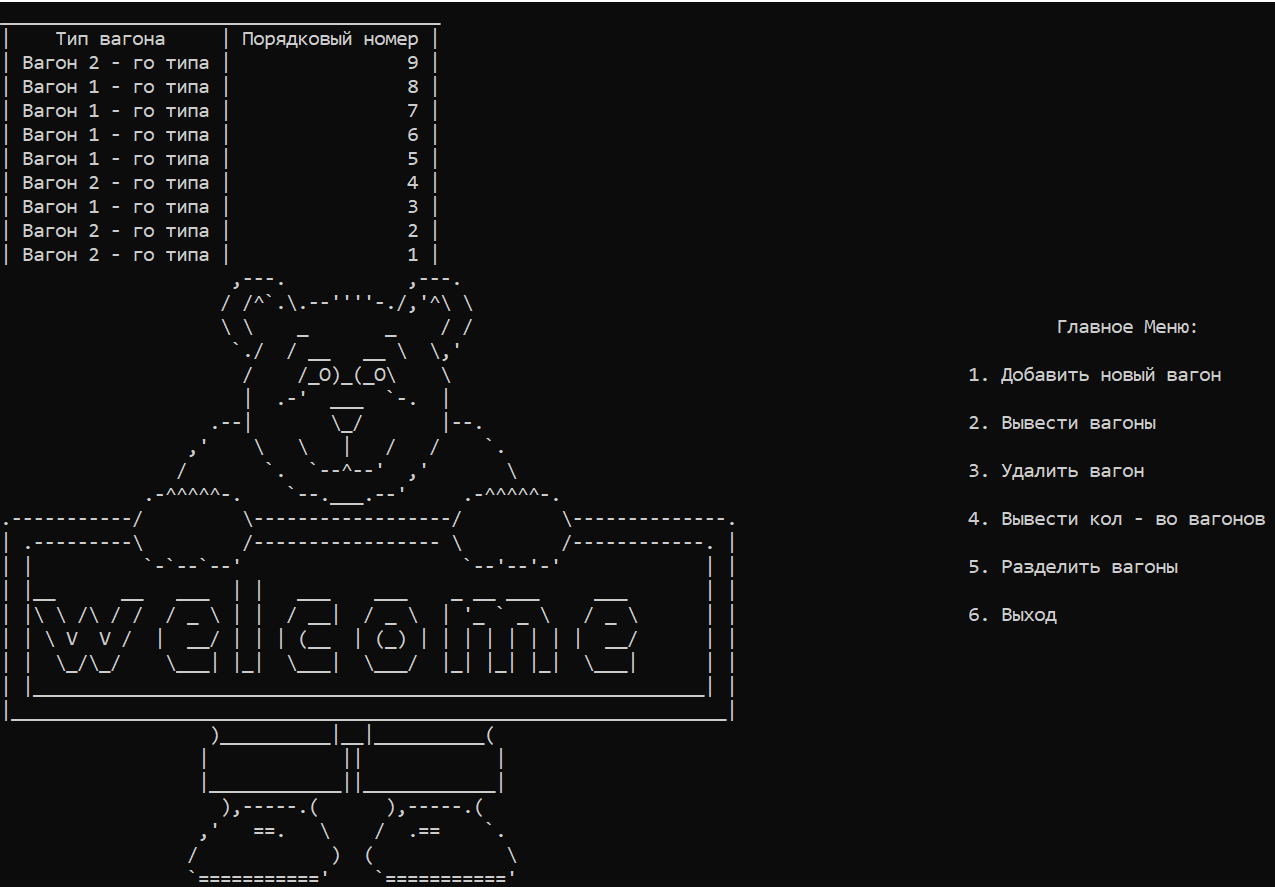
Выбор:



Вывод:

****

Добавление:



Удаление:



Вывод количества вагонов:



Разделение:



**Ответы на вопросы:**

1. **Как вы понимаете, что такое класс и объект? Приведите примеры.**

Класс — это абстракция того, что мы хотим, а объект класса — это то, над чем мы воздействуем. Например:

1. **Каков синтаксис и смысл определения класса? Как называются составные части класса?**

class название\_класса { тело\_класса; };

В классе есть поля (атрибуты) класса и методы (возможности) класса.

1. **Что такое идентификаторы доступа к элементам класса? Дайте определения.**

Идентификаторы доступа к элементам класса это к чему может программист из места вызова объекта обратится, есть public, private и protected доступы. К public секции программист может свободно обращаться, к protected только наследуемые классы, к private только внутри класса.

1. **Что такое инкапсуляция?**

Сокрытие данных, то есть те данные, которые лежат в private секции. Те переменные или методы, которые не должен увидеть другой программист, работающий в команде.

1. **Где в программе с классами можно создавать объекты?**

Где угодно.

1. **В какой момент метод готов для вызова объектом?**

Когда угодно, если нет синтаксических ошибок при компиляции, и он находится в public секции.

1. **Как называются методы, встроенные в структуру класса?**

Методы класса.

1. **Как определяется метод, если внутри класса записан только его заголовок, сам метод определен вне класса?**

Прототип метода.

1. **Тиражируются ли методы класса?**

Методы класса не тиражируются

1. **Как в программе написать доступ к открытым и закрытым полям класса?**

По умолчанию в c++ у класса private секция, чтобы сделать public секцию нужно написать public:, а для того чтобы сделать private секцию нужно написать private: