**Домашнее задание к дисциплине «Синтаксис С++»**

**Задание №1**

В уроках №2 и №3 Вы познакомились с переменными, в том числе узнали как объявить переменную, задать ей значение через консоль и вывести это значение обратно в консоль. Для отработки данного навыка создайте несколько переменных основных типов, введите их значения через консоль и после выведите все сохранённые значения обратно в консоль.

Протестируйте программу. Сделайте скриншот консоли после выполнения всей

программы и отправьте его преподавателю.

**// https://github.com/A-l-E-v/CPP\_Synergy/blob/main/U-379/vars.cpp**

#include <iostream>

using namespace std;

int main()

{

int a;

double b;

char c;

bool d;

string e;

cout << "Введите int: ";

cin >> a;

cout << "Введите double: ";

cin >> b;

cout << "Введите char: ";

cin >> c;

cout << "Введите bool (1|0): ";

cin >> d;

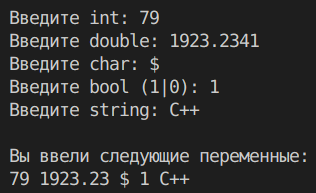
cout << "Введите string: ";

cin >> e;

cout << "\nВы ввели следующие переменные:\n";

cout << a << " " << b << " " << c << " " << d << " " << e << "\n";

return 0;



}

**Задание №2**

В уроке №4 Вы узнали как работает условный оператор. Для отработки данного навыка напишите программу, которая определяет чётность числа.

Протестируйте программу несколько раз. Отправьте преподавателю 2 скриншота, один для чётного числа, а другой для нечётного.

**#include <iostream>**

using namespace std;

int main()

{

int x;

cout << "\nВведите целое число: ";

cin >> x;

if (x % 2 == 0) {

cout << "\nЧисло: " << x << " чётное.\n";

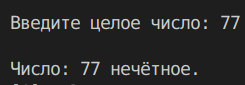
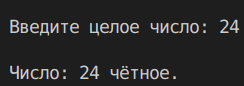
} else {

cout << "\nЧисло: " << x << " нечётное.\n";

}

return 0;

}

****

**Задание №3**

Написать программу, которая создаёт текстовый файл, записывает в него строку, введённую пользователем в консоль. Затем программа должна закрыть файл для записи, открыть его для чтения, считать записанную строку и вывести её в консоль.

Отправьте преподавателю код программы, скриншот консоли и скриншот открытого

текстового файла.

**#include <iostream>**

#include <fstream>

using namespace std;

int main()

{

string str;

fstream file\_w("str\_file.txt", ios\_base::out);

// файл открыт?

if (file\_w.is\_open())

{

cout << "Файл str\_file.txt открыт для записи!\n";

cout << "Введите строку для записи в файл: ";

getline(cin, str);

cout << "Вы ввели: " << str << "\n";

file\_w << str << endl;

file\_w.close();

cout << "Файл закрыт после записи.\n\n";

}

else

{

cout << "Ошибка открытия файла на запись.\n";

return 1;

}

fstream file\_r("str\_file.txt", ios\_base::in);

// файл открыт?

if (file\_r.is\_open())

{

cout << "Файл str\_file.txt открыт для чтения!\n";

cout << "Содержимое файла:\n";

// читаем файл с самого начала

file\_r.seekg(0, ios\_base::beg);

// сохраняем в строку

getline(file\_r, str);

cout << str << endl;

file\_r.close();

cout << "Файл закрыт после чтения.\n";

}

else

{

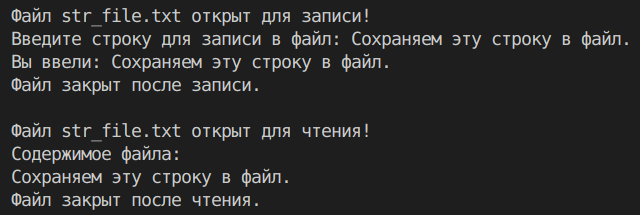
cout << "Ошибка открытия файла на чтение.\n";

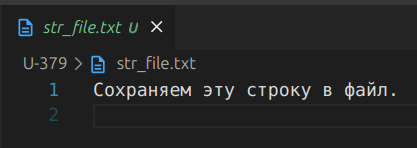
return 1;

}

return 0;

}





**Задание №4\***

Создать целочисленную переменную, а также указатель ну эту целочисленную переменную. Вывести в консоль адрес переменной двумя способами (используя саму переменную и указатель на переменную). Также вывести в консоль значение переменной, используя указатель на неё.

Отправьте преподавателю код программы и скриншот консоли.

**#include <iostream>**

using namespace std;

int main()

{

int x = 72;

int\* ptr\_x = &x;

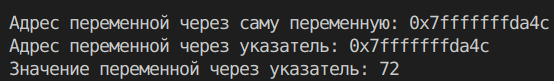
cout << "\nАдрес переменной через саму переменную: " << &x << "\n";

cout << "Адрес переменной через указатель: " << ptr\_x << "\n";

cout << "Значение переменной через указатель: " << \*ptr\_x << "\n";

return 0;

}

****

**Задание №5\***

Пользователь вводит N и M – количество строк и столбцов в матрице. Программа должна заполнить матрицу случайными двузначными числами и корректно вывести полученную матрицу в консоль. После этого пользователь вводит номер строки или столбца, а программа выводит сумму числе в соответствующей строке или соответствующем столбце.

Отправьте преподавателю код программы и скриншот консоли.

Эта задача была сделана, описана и отправлена на проверку в уроке 3.6.5. Ссылка на код программы:

https://github.com/A-l-E-v/CPP\_Synergy/blob/main/U-365/nm\_array.cpp

**Задание №6\***

Напишите рекурсивную функцию, которая переводит число из десятичной системы в двоичную. Допускается использовать тип string для хранения двоичного числа.

Отправьте преподавателю код программы и скриншот консоли.

Эта задача была сделана, описана и отправлена на проверку в уроке 3.6.5. Ссылка на код программы:

https://github.com/A-l-E-v/CPP\_Synergy/blob/main/U-365/recbin.cpp

**Задание №7\***

Реализуйте программу «Заметки». В начале программы должен выводиться список доступных заметок – список текстовых файлов в директории программы. Пользователь может выбрать заметку и считать с неё информацию или создать новую заметку.

Отправьте преподавателю код программы, скриншот консоли в процессе выполнения, скриншот директории с заметками и скриншот открытой заметки.

**#include <iostream>**

#include <filesystem>

#include <string>

#include <map>

#include <fstream>

#include <iomanip>

#include <ctime>

#include <sstream>

namespace fs = std::filesystem;

using namespace std;

string path = "notes\_dir";

string full\_path;

string note\_text = "";

map<int, string> notes\_map;

int files = 1;

int total\_files;

bool quit = false;

bool new\_note = false;

int menu\_entry = 1;

void create\_note()

{

cout << "\nВведите текст новой заметки: ";

cin.ignore();

getline(cin, note\_text);

auto t = time(nullptr);

auto tm = \*localtime(&t);

ostringstream oss;

oss << put\_time(&tm, "%d-%m-%Y@%H:%M:%S");

auto date\_time = oss.str();

full\_path = path + "/" + "Note\_on\_" + date\_time;

fstream file\_w(full\_path, ios\_base::out);

file\_w << note\_text << endl;

file\_w.close();

return;

}

void show\_note(int menu\_entry)

{

fstream note\_file(notes\_map[menu\_entry], ios\_base::in);

if (note\_file.is\_open())

{

note\_file.seekg(0, ios\_base::beg);

getline(note\_file, note\_text);

cout << "\n\n~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~\n\n";

cout << note\_text << endl;

cout << "\n\n~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~\n\n";

note\_file.close();

}

return;

}

int main()

{

while (!quit)

{

cout << "\n\nВаши заметки:\n";

for (const auto &entry : fs::directory\_iterator(path))

{

notes\_map[files] = entry.path();

files++;

}

total\_files = files;

files = 1;

for (auto it : notes\_map)

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

cout << "\n 0 - новая заметка\n-1 - выход\n";

cout << "\nВведите действие или номер заметки: ";

cin >> menu\_entry;

switch (menu\_entry)

{

case 0:

{

create\_note();

break;

}

case -1:

cout << "\nУдачного дня!\n";

return 0;

break;

default:

if (menu\_entry <= total\_files)

{

show\_note(menu\_entry);

break;

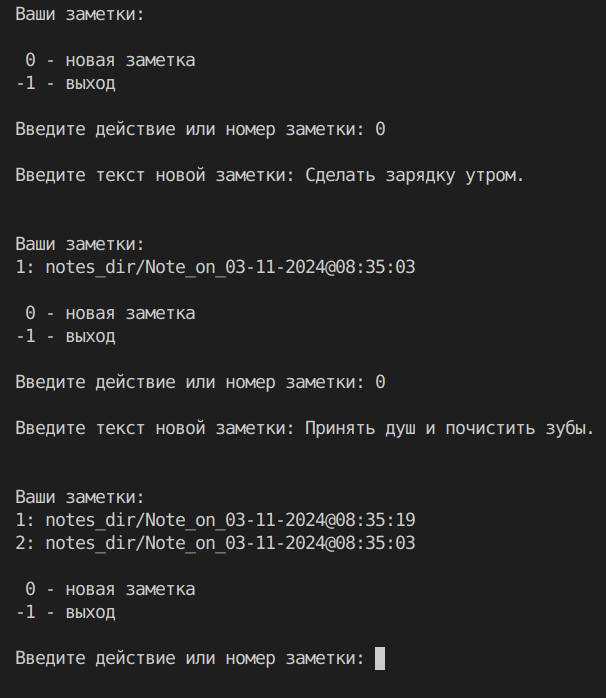
}

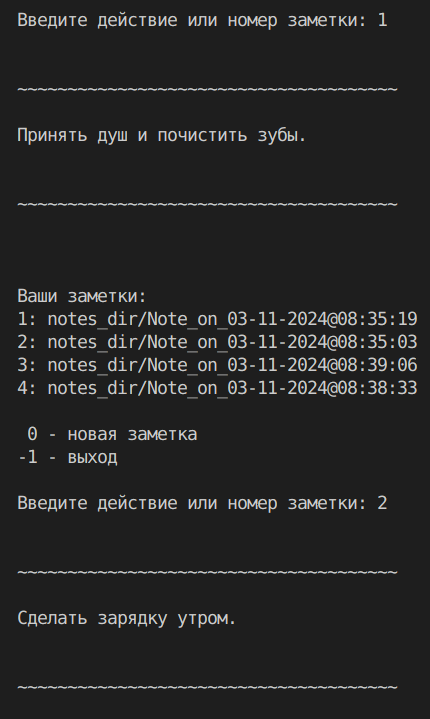
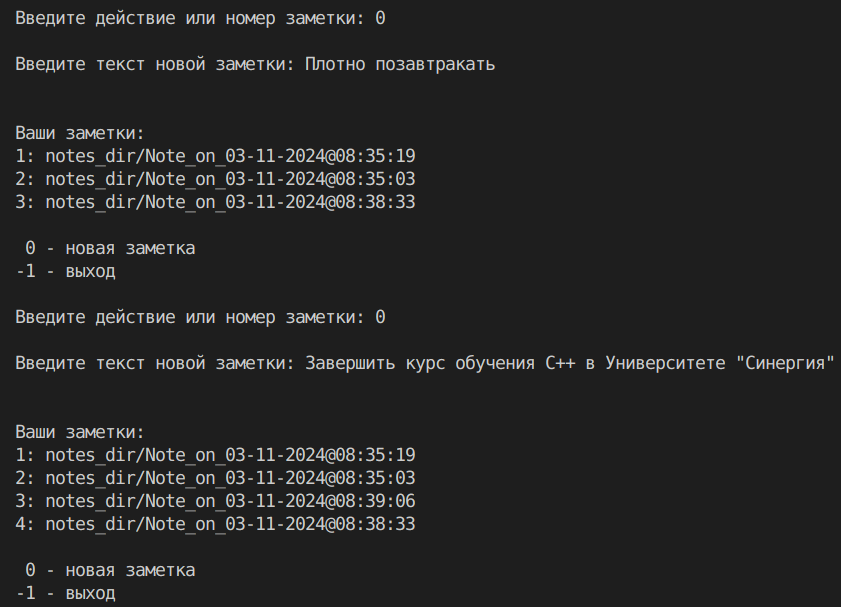
}

}

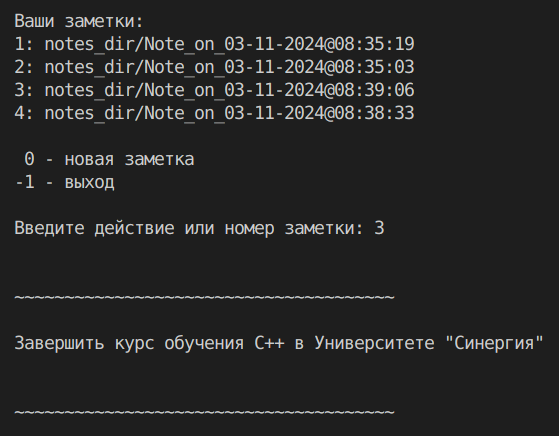
}

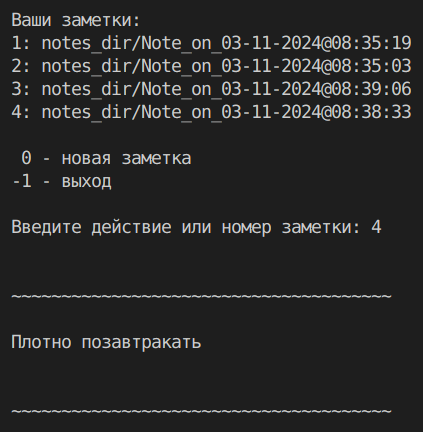
При первом запуске программы список заметок пуст и предлагается создать новую заметку или выйти из программы. Начинаем создавать заметки:



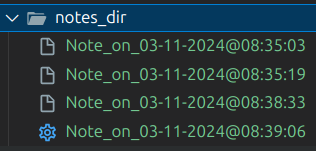


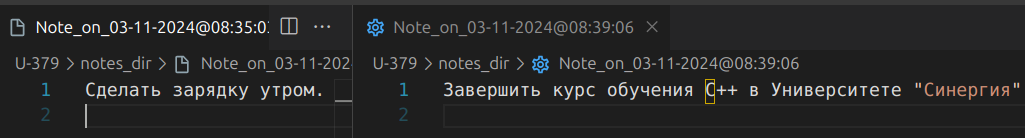
Итого было создано 4 заметки. Можем их просмотреть:

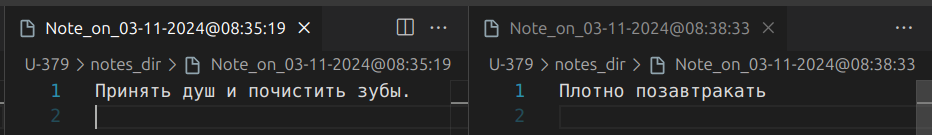




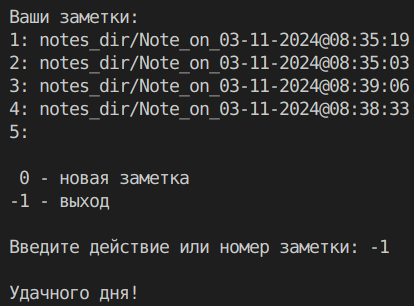
Содержимое директории notes\_dir — содержит все сохранённые заметки:







Выходим из программы, введя «-1»:



**Задание №8\***

Реализовать упрощённый вариант стека – структуры, где всегда доступен только последний добавленный элемент. Элементом стека должна быть структура, хранящая целое число и указатель на элемент. Количество элементов и сами элементы пользователь вводит с консоли, после чего они должны быть выведены в корректно порядке. Не забудьте в конце корректно удалить все элементы стека.

Отправьте преподавателю код программы и скриншот консоли.

**#include <iostream>**

using namespace std;

// кол-во элементов стека

int stack\_size;

class Stack

{

int \*value;

int position;

public:

Stack(int size); // конструктор

~Stack(); // деструктор

void push(int); // добавить элемент

int pop(); // извлечь элемент

};

// Конструктор и инициализации стека

Stack::Stack(int size)

{

value = new int[size];

position = -1;

}

// Деструктор и удаление стека

Stack::~Stack()

{

delete[] value;

}

// Добавляем элемент

void Stack::push(int x)

{

value[++position] = x;

}

// Извлекаем верхний элемент

int Stack::pop()

{

return value[position--];

}

int main()

{

int x;

// ввод элементов стека

cout << "Введите количество элементов стека: ";

cin >> stack\_size;

Stack stack(stack\_size);

cout << "\n";

for (size\_t i = 0; i < stack\_size; i++)

{

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

cin >> x;

stack.push(x);

}

cout << "\nВывод элементов стека:\n";

for (size\_t i = 0; i < stack\_size; i++)

{

cout << stack.pop() << " ";

}

cout << "\nВывод стека завершён!\n\n";

}

