ПОЯСНЕНИЕ КОДА

**Мини-игра №1: «Угадай число» (Генерация случайных чисел)**

1. Описание программы

Данная программа представляет собой игру "Угадай число". Компьютер загадывает случайное число от 1 до 100, а пользователь должен угадать его. После каждой попытки программа сообщает, больше или меньше загаданное число. Игра продолжается до тех пор, пока пользователь не угадает число.

После завершения игры программа спрашивает, хочет ли пользователь сыграть ещё раз.

2. Разбор кода

Функция playGame():

setlocale(LC\_ALL, "RU"); //Установка кодировку на RU

//Генератор случайных чисел

std::random\_device rd;

std::mt19937 gen(rd());

std::uniform\_int\_distribution<> distrib(1, 100); //диапазон случайных чисел от 1 до 100

int secretNumber = distrib(gen);

int guess; //переменная для хранения числа, введенного пользователем

int attempts = 0; // счётчик попыток

std::cout << "Я загадал число от 1 до 100. Попробуйте угадать! " << "Подсказка: " << secretNumber << std::endl;

setlocale(LC\_ALL, "RU"); - устанавливает поддержку русского языка

std::random\_device rd;

std::mt19937 gen(rd()); - создание генератора случайных чисел

std::uniform\_int\_distribution<> distrib(1, 100); - задает диапазон случайных чисел от 1 до 100

int secretNumber = distrib(gen); - генерирует и сохраняет случайное число

int guess; - переменная для хранения числа, введенного пользователем

int attempts = 0; - счётчик попыток

while (true) { //цикл будет идти бесконечно пока число не будет угадано

//Проверка на корректность ввода числа

std::cout << "Введите число: ";

if (!(std::cin >> guess)) {

std::cin.clear(); //Сброс флага ошибки

std::cin.ignore(10000, '\n'); //Очистить буфер ввода

std::cout << "Некорректный ввод. Введите число!" << std::endl;

continue;

}

attempts++; //Увеличение счетчика попыток

//Проверка введеннего числа

if (guess < secretNumber) {

std::cout << "Загаданное число больше." << std::endl;

}

else if (guess > secretNumber) {

std::cout << "Загаданное число меньше." << std::endl;

}

else {

std::cout << "Поздравляю! Вы угадали число за " << attempts << " попыток." << std::endl;

break;

}

}

}

* Программа запрашивает у пользователя ввод числа.
* Если введено некорректное значение (например, буква), ввод очищается и программа просит ввести число снова.
* Каждая попытка увеличивает счётчик attempts.
* Если число меньше загаданного — программа говорит, что число больше.
* Если больше — программа говорит, что число меньше.
* Если угадано — выводится сообщение с числом попыток, и цикл завершается.

Функция main ():

int main() { // Проверяет хочет ли пользователь сыграть еще раз

char playAgain;

do {

playGame();

std::cout << "Хотите сыграть ещё раз? (y/n): ";

std::cin >> playAgain;

} while (playAgain == 'y' || playAgain == 'Y');

std::cout << "Спасибо за игру!" << std::endl; //Если не хочет, то игра заканчивается и выводит

return 0;

}

* Запускает игру с вызовом playGame().
* После завершения игры спрашивает, хочет ли пользователь сыграть ещё раз.
* Если пользователь вводит 'y' или 'Y', игра запускается заново.
* Если 'n' или другое значение — программа завершает работу.

**Мини-игра №2: «Генератор историй» (Работа с файлами)**

1. Описание программы

Данная программа генерирует случайные истории на основе заранее заданных списков и позволяет сохранять их в файл. Также она запоминает количество созданных историй и при следующем запуске показывает этот прогресс.

2. Разбор кода

Генератор случайных чисел

void initRandom() {

std::srand(static\_cast<unsigned int>(std::time(nullptr)));

}

Это чтобы случайные числа (или истории) были разными при каждом запуске

Генератор случайного элемента из вектора

// Генератор случайного элемента из вектора

std::string getRandomElement(const std::vector<std::string>& vec) {

int index = std::rand() % vec.size();

return vec[index];

}

Возвращает случайные элементы из списка (герой, место, действий и так далее)

// Генератор истории

std::string generateStory(const std::vector<std::string>& heroes,

const std::vector<std::string>& places,

const std::vector<std::string>& actions,

const std::vector<std::string>& details) {

return getRandomElement(heroes) + " " + getRandomElement(places) + ", " +

getRandomElement(actions) + " " + getRandomElement(details) + ".";

}

Выбирает случайные элементы из 4 списка и соединяет их в предложение

// Сохранение истории в файл

void saveStory(const std::string& story) {

std::ofstream outFile("stories.txt", std::ios::app);

if (outFile.is\_open()) {

outFile << story << "\n";

outFile.close();

std::cout << "История сохранена в файл stories.txt\n";

}

else {

std::cout << "Ошибка открытия файла!\n";

}

}

Открывает файл stories.txt в режиме дозаписи std::ios::app. Записывает в него новую историю. Если не удалось открыть файл, то сообщит об ошибке.

// Сохранение прогресса (количество историй)

void saveProgress(int storyCount) {

std::ofstream progressFile("progress.txt");

if (progressFile.is\_open()) {

progressFile << storyCount;

progressFile.close();

}

}

Записывает количество созданных историй в файл progress.txt.

// Загрузка прогресса (количество историй)

int loadProgress() {

std::ifstream progressFile("progress.txt");

int storyCount = 0;

if (progressFile.is\_open()) {

progressFile >> storyCount;

progressFile.close();

}

return storyCount;

}

Загружается количество созданных истории из progress.txt. Если файл пустой, то прогресс начинается с нуля

Главная функция main():

int main() {

setlocale(LC\_ALL, "RU");

initRandom();

std::vector<std::string> heroes = { "Смелый рыцарь", "Хитрый вор", "Волшебник", "Отважный пират", "Дерзкий исследователь" };

std::vector<std::string> places = { "в далёком королевстве", "на заброшенной фабрике", "в густом лесу", "на просторах космоса", "у подножия гор" };

std::vector<std::string> actions = { "победил дракона", "обнаружил сокровища", "выиграл битву", "устроил бал", "раскрыл древнюю тайну" };

std::vector<std::string> details = { "с волшебным мечом", "на летающем ковре", "под звуки волшебной музыки", "с удивительной силой", "в сопровождении магического существа" };

int storyCount = loadProgress();

std::cout << "Вы ранее создали " << storyCount << " историй.\n";

char playAgain;

do {

std::string story = generateStory(heroes, places, actions, details);

std::cout << "\nСгенерированная история:\n" << story << "\n";

std::cout << "\nСохранить историю в файл (Y/N)? ";

char saveChoice;

std::cin >> saveChoice;

if (saveChoice == 'Y' || saveChoice == 'y') {

saveStory(story);

storyCount++;

saveProgress(storyCount);

}

std::cout << "\nХотите создать ещё одну историю? (Y/N): ";

std::cin >> playAgain;

} while (playAgain == 'Y' || playAgain == 'y');

std::cout << "Спасибо за игру! Вы создали всего " << storyCount << " историй.\n";

return 0;

}

setlocale(LC\_ALL, "RU"); - устанавливает поддержку русского языка

int storyCount = loadProgress();

std::cout << "Вы ранее создали " << storyCount << " историй.\n"; - загружает ранее созданных историй из progress.txt

char playAgain;

do {

std::string story = generateStory(heroes, places, actions, details);

std::cout << "\nСгенерированная история:\n" << story << "\n";

std::cout << "\nСохранить историю в файл (Y/N)? ";

char saveChoice;

std::cin >> saveChoice;

if (saveChoice == 'Y' || saveChoice == 'y') {

saveStory(story);

storyCount++;

saveProgress(storyCount);

}

std::cout << "\nХотите создать ещё одну историю? (Y/N): ";

std::cin >> playAgain;

} while (playAgain == 'Y' || playAgain == 'y');

std::cout << "Спасибо за игру! Вы создали всего " << storyCount << " историй.\n";

return 0;

Генерирует новую случайную историю, далее предлагает сохранить историю в файл и создать еще одну историю. После окончания игру выводит всего созданных историй за игру.