#include <iostream>

#include <string>

#include <vector>

#include <algorithm>

#include <random>

#include <limits>

#include <cstdlib>

#include <locale>

#include <fstream>

#include <sstream>

#include <Windows.h>

#include <cctype>

#include <codecvt>

using namespace std;

vector<string> round\_Info;

vector<string> words\_List;

HANDLE handle = GetStdHandle(STD\_OUTPUT\_HANDLE);

void readWordsFromFile(const string& filename, vector<string>& words) {

ifstream file(filename); // Открываем файл для чтения

if (!file.is\_open()) { // Проверяем, открылся ли файл

SetConsoleTextAttribute(handle, 4);

cerr << "Не удалось открыть файл " << filename << endl;

SetConsoleTextAttribute(handle, 7);

exit(1);

}

string word;

while (getline(file, word)) { // Читаем файл построчно (каждая строка — одно слово)

words.push\_back(word); // Добавляем слово в вектор

}

file.close(); // Закрываем файл

}

// Игрок

struct Player {

int score = 0;

int id = 0;

string name;

string givenWord;

string questions[3][3];

};

void showRound() {

for (auto res : round\_Info)

{

cout << res << "\n\n";

}

};

void takeRound(Player otgad, Player\* zagad, int points)

{

switch (points)

{

case 0: round\_Info[otgad.id - 1] = "Игрок " + otgad.name + " (" + to\_string(otgad.id) + ") не смог отгадать слово " + zagad->givenWord + " игрока " + zagad->name + "(" + to\_string(zagad->id) + ") из-за чего загадывающий теряет 1 очк.!"; break;

case 1: round\_Info[otgad.id - 1] = "Игрок " + otgad.name + " (" + to\_string(otgad.id) + ") смог отгадать слово " + zagad->givenWord + " игрока " + zagad->name + "(" + to\_string(zagad->id) + ") за кол-во попыток: 3. 1 очк. получает и загадывающий и отгадывающий!"; break;

default: round\_Info[otgad.id - 1] = "Игрок " + otgad.name + " (" + to\_string(otgad.id) + ") смог отгадать слово " + zagad->givenWord + " игрока " + zagad->name + "(" + to\_string(zagad->id) + ") за кол-во попыток: " + to\_string(4 - points) + ". Отгадывающий получает: " + to\_string(points) + " очк."; break;

}

}

void showFinalResults(const vector<Player>& vctrPlayers) {

SetConsoleTextAttribute(handle, 3);

cout << "\n==================== ИТОГОВЫЕ РЕЗУЛЬТАТЫ ====================" << endl;

SetConsoleTextAttribute(handle, 7);

for (int i = 0; i < vctrPlayers.size(); i++) {

cout << "Игрок " << i + 1 << " (" << vctrPlayers[i].name << ") - Баллы: " << vctrPlayers[i].score << endl;

}

SetConsoleTextAttribute(handle, 3);

cout << "=============================================================\n" << endl;

SetConsoleTextAttribute(handle, 7);

}

// Функция для очистки консоли

void clearConsole() {

#ifdef \_WIN32

system("cls"); // Очистка консоли для Windows

#else

system("clear"); // Очистка консоли для Linux/Unix/MacOS

#endif

}

// Функция для ожидания передачи хода следующему игроку

void waitForNextPlayer(int code = 0, string name = "empty") {

cout << "" << endl;

// Считываем ввод пользователя

string input;

cout << "Нажмите Enter, чтобы продолжить..." << endl;

getline(cin, input);

// Если пользователь ввел что-то кроме Enter

while (input != "") {

SetConsoleTextAttribute(handle, 4);

cout << "Некорректный ввод. Пожалуйста, нажмите Enter, чтобы продолжить..." << endl;

SetConsoleTextAttribute(handle, 7);

getline(cin, input);

}

// Очищаем консоль

clearConsole();

if (code)

{

SetConsoleTextAttribute(handle, 3);

cout << "Переход хода игроку " << name << " (" << to\_string(code) << ")" << endl;

SetConsoleTextAttribute(handle, 7);

}

// Второе ожидание нажатия Enter

cout << "Нажмите Enter, чтобы продолжить..." << endl;

getline(cin, input);

while (input != "") {

SetConsoleTextAttribute(handle, 4);

cout << "Некорректный ввод. Пожалуйста, нажмите Enter, чтобы продолжить..." << endl;

SetConsoleTextAttribute(handle, 7);

getline(cin, input);

}

clearConsole();

}

string processString(std::string input) {

// Установка локали для работы с UTF-8

locale loc("ru\_RU.UTF-8");

wstring\_convert<codecvt\_utf8<wchar\_t>, wchar\_t> converter;

// Преобразуем строку в wide string

wstring wideInput = converter.from\_bytes(input);

wstring result;

for (wchar\_t c : wideInput) {

// Проверяем диапазон символов

if (c >= L'А' && c <= L'Я') {

result += c + (L'а' - L'А'); // Преобразуем заглавную букву в строчную

}

else if (c == L'Ё') {

result += L'ё'; // Специальный случай для буквы Ё

}

else {

result += c; // Остальные символы остаются неизменными

}

}

// Преобразуем обратно в UTF-8 и выводим

string output = converter.to\_bytes(result);

return output;

}

void checkForMinus(vector<Player>& vctrPlayers)

{

for (int i = 0; i < vctrPlayers.size(); i++)

{

if (vctrPlayers[i].score < 0)

{

vctrPlayers[i].score = 0;

}

}

}

// Функция для проверки ввода числа

int getValidNumberInput(int min, int max) {

int num;

while (true) {

cin >> num;

if (cin.fail() || num < min || num > max) {

SetConsoleTextAttribute(handle, 4);

cout << "Ошибка! Пожалуйста, введите число от " << min << " до " << max << "." << endl;

SetConsoleTextAttribute(handle, 7);

cin.clear(); // Сброс флага ошибки

cin.ignore(100000, '\n'); // Очищаем входной поток

}

else {

return num;

}

}

}

string getValidStringInput() {

string input;

while (input.empty()) {

getline(cin, input);

if (input.empty()) {

SetConsoleTextAttribute(handle, 4);

cout << "Ошибка! Пожалуйста, введите непустое значение." << endl;

SetConsoleTextAttribute(handle, 7);

}

else {

return input;

}

}

return input;

}

//

string AssistMQ() {

string input;

getline(cin, input);

return input;

}

void makeQuestions(Player& player) {

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

switch (j) {

case 0:SetConsoleTextAttribute(handle, 3);

cout << endl << "Введите 3 прилагательных: " << endl; break;

case 1:SetConsoleTextAttribute(handle, 3);

cout << endl << "Введите 3 глагола: " << endl; break;

case 2:SetConsoleTextAttribute(handle, 3);

cout << endl << "Введите 3 существительных: " << endl; break;

SetConsoleTextAttribute(handle, 7);

}

bool validInput = false;

while (!validInput) {

// Ввод одной строки с тремя словами

SetConsoleTextAttribute(handle, 7);

cout << "Введите 3 слова через пробел: ";

string input = AssistMQ();

// Используем stringstream для разделения строки на слова

stringstream ss(input);

string word;

vector<string> words;

// Читаем все слова из строки

while (ss >> word) {

words.push\_back(word);

}

// Проверяем, что введено ровно 3 слова

if (words.size() == 3) {

// Заполняем массив player.questions[j]

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

player.questions[j][i] = words[i];

}

validInput = true; // Ввод правильный, выходим из цикла

}

else {

SetConsoleTextAttribute(handle, 4);

cout << "Ошибка: введите ровно 3 слова!" << endl;

SetConsoleTextAttribute(handle, 7);

}

}

}

}

//

// Функция для передачи слов игрокам

void giveWord(vector<Player>& vctrPlayers) {

for (int i = 0; i < vctrPlayers.size(); i++) {

if (words\_List.empty()) {

SetConsoleTextAttribute(handle, 4);

cout << endl << "Произошла ошибка. Слова для игры закончились! Программа будет завершена!";

SetConsoleTextAttribute(handle, 7);

showFinalResults(vctrPlayers);

exit(1);

}

vctrPlayers[i].givenWord = words\_List.back();

words\_List.pop\_back();

}

}

// Этап 1: Загадать слово и ввести вопросы

void stageOne(vector<Player>& vctrPlayers) {

giveWord(vctrPlayers);

SetConsoleTextAttribute(handle, 3);

cout << "Этап 1: Загадывание слов." << endl;

SetConsoleTextAttribute(handle, 7);

for (size\_t i = 0; i < vctrPlayers.size(); i++) {

waitForNextPlayer(vctrPlayers[i].id, vctrPlayers[i].name);

SetConsoleTextAttribute(handle, 3);

cout << "\n========================================" << endl;

SetConsoleTextAttribute(handle, 7);

cout << "\nХод игрока номер " << i + 1 << " (" << vctrPlayers[i].name << "):" << endl;

cout << "\nВаше слово для загадки: ";

SetConsoleTextAttribute(handle, 3);

cout << vctrPlayers[i].givenWord << endl;

SetConsoleTextAttribute(handle, 7);

makeQuestions(vctrPlayers[i]);

SetConsoleTextAttribute(handle, 3);

cout << "========================================\n" << endl;

SetConsoleTextAttribute(handle, 7);

}

}

// Этап 2: Угадывание слов

void stageTwo(vector<Player>& vctrPlayers) {

random\_device rd;

mt19937 gen(rd());

vector<Player\*> randomUkaz(vctrPlayers.size());

for (int i = 0; i < vctrPlayers.size(); i++) {

randomUkaz[i] = &(vctrPlayers[i]);

}

shuffle(randomUkaz.begin(), randomUkaz.end(), gen);

SetConsoleTextAttribute(handle, 3);

cout << "Этап 2: Отгадывание слов." << endl;

SetConsoleTextAttribute(handle, 7);

for (size\_t i = 0; i < vctrPlayers.size(); i++) {

waitForNextPlayer(vctrPlayers[i].id, vctrPlayers[i].name);

SetConsoleTextAttribute(handle, 3);

cout << "\n========================================" << endl;

SetConsoleTextAttribute(handle, 7);

cout << "\nХод игрока номер " << i + 1 << " (" << vctrPlayers[i].name << "):" << endl;

for (size\_t j = 0; j < vctrPlayers.size(); j++) {

if (&(vctrPlayers[i]) == randomUkaz[j]) {

continue;

}

else {

int k = 3;

for (; k > 0; k--) {

switch (k) {

case 3:

SetConsoleTextAttribute(handle, 3);

cout << "3 прилагательных: ";

SetConsoleTextAttribute(handle, 7);

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

cout << randomUkaz[j]->questions[0][m] << " ";

}

cout << endl;

break;

case 2:

cout << "\nХод игрока номер " << i + 1 << " (" << vctrPlayers[i].name << "):" << endl;

SetConsoleTextAttribute(handle, 3);

cout << "3 глагола: ";

SetConsoleTextAttribute(handle, 7);

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

cout << randomUkaz[j]->questions[1][m] << " ";

}

cout << endl;

SetConsoleTextAttribute(handle, 3);

cout << "3 прилагательных: ";

SetConsoleTextAttribute(handle, 7);

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

cout << randomUkaz[j]->questions[0][m] << " ";

}

cout << endl;

break;

case 1:

cout << "\nХод игрока номер " << i + 1 << " (" << vctrPlayers[i].name << "):" << endl;

SetConsoleTextAttribute(handle, 3);

cout << "3 существительных: ";

SetConsoleTextAttribute(handle, 7);

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

cout << randomUkaz[j]->questions[2][m] << " ";

}

cout << endl;

SetConsoleTextAttribute(handle, 3);

cout << "3 глагола: ";

SetConsoleTextAttribute(handle, 7);

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

cout << randomUkaz[j]->questions[1][m] << " ";

}

cout << endl;

SetConsoleTextAttribute(handle, 3);

cout << "3 прилагательных: ";

SetConsoleTextAttribute(handle, 7);

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

cout << randomUkaz[j]->questions[0][m] << " ";

}

cout << endl;

break;

}

cout << "\nВведите слово: ";

string answer;

answer = getValidStringInput();

// Удаление лишних пробелов

answer.erase(remove(answer.begin(), answer.end(), ' '), answer.end());

string givenWordLower = randomUkaz[j]->givenWord;

givenWordLower.erase(remove(givenWordLower.begin(), givenWordLower.end(), ' '), givenWordLower.end());

// Преобразование к нижнему регистру

locale loc;

for (char& c : answer) c = std::tolower(c, loc);

for (char& c : givenWordLower) c = std::tolower(c, loc);

if (processString(answer) != processString(randomUkaz[j]->givenWord)) {

clearConsole();

SetConsoleTextAttribute(handle, 7);

SetConsoleTextAttribute(handle, 3);

cout << "\n========================================" << endl;

SetConsoleTextAttribute(handle, 7);

continue;

}

else {

switch (k) {

case 1: {

SetConsoleTextAttribute(handle, 3);

cout << "Поздравляю! Вы отгадали слово!" << endl;

cout << "Вы и загадывающий получаете по 1 очку!" << endl;

vctrPlayers[i].score++;

(randomUkaz[j]->score)++;

SetConsoleTextAttribute(handle, 7);

break;

}

default: {

SetConsoleTextAttribute(handle, 3);

cout << "Поздравляю! Вы отгадали слово!" << endl;

cout << "Вы получаете " << k << " очков!" << endl;

vctrPlayers[i].score += k;

SetConsoleTextAttribute(handle, 7);

break;

}

}

break;

}

}

if (k == 0) {

cout << "Увы, вы не смогли отгадать слово. Загадывающий теряет 1 очко!" << endl;

(randomUkaz[j]->score)--;

}

takeRound(vctrPlayers[i], randomUkaz[j], k);

randomUkaz.erase(randomUkaz.begin() + j);

break;

}

}

cin.get();

}

}

// Функция для отображения статистики игроков

void showStats(vector<Player>& vctrPlayers) {

SetConsoleTextAttribute(handle, 3);

cout << "\n==================== Статистика ====================" << endl;

SetConsoleTextAttribute(handle, 7);

for (int i = 0; i < vctrPlayers.size(); i++) {

cout << "Игрок " << i + 1 << " (" << vctrPlayers[i].name << ") - Баллы: " << vctrPlayers[i].score << endl;

}

SetConsoleTextAttribute(handle, 3);

cout << "====================================================\n" << endl;

SetConsoleTextAttribute(handle, 7);

}

// Проверка победителей

int checkWinners(vector<Player> vctrPlayers) {

int result = 0;

for (int i = 0; i < vctrPlayers.size(); i++) {

if (vctrPlayers[i].score >= 15) {

SetConsoleTextAttribute(handle, 3);

cout << "\nИгрок " << i + 1 << " (" << vctrPlayers[i].name << ") одержал победу, набрав " << vctrPlayers[i].score << " очков!" << endl;

SetConsoleTextAttribute(handle, 7);

result++;

}

}

return result;

}

int main() {

SetConsoleCP(1251);

setlocale(LC\_ALL, "Russian");

keybd\_event(VK\_MENU, 0x38, 0, 0);

keybd\_event(VK\_RETURN, 0x1c, 0, 0);

keybd\_event(VK\_RETURN, 0x1c, KEYEVENTF\_KEYUP, 0);

keybd\_event(VK\_MENU, 0x38, KEYEVENTF\_KEYUP, 0);

string filename = "dictionary.txt"; // Укажите имя вашего файла

readWordsFromFile(filename, words\_List);

random\_device rd;

mt19937 gen(rd());

shuffle(words\_List.begin(), words\_List.end(), gen);

int numberOfPlayers;

SetConsoleTextAttribute(handle, 3);

cout << "=================== Добро пожаловать в игру! ===================" << endl;

SetConsoleTextAttribute(handle, 7);

cout << "Введите число игроков (от 2 до 5): ";

numberOfPlayers = getValidNumberInput(2, 5);

SetConsoleTextAttribute(handle, 3);

cout << "================================================================\n" << endl;

SetConsoleTextAttribute(handle, 7);

vector<string> temp(numberOfPlayers);

round\_Info = temp;

vector<Player> players(numberOfPlayers);

cin.clear(); // Сброс флага ошибки

cin.ignore(100000, '\n'); // Очищаем входной поток

for (size\_t i = 0; i < numberOfPlayers; i++) {

players[i].id = i + 1;

cout << "Введите имя игрока номер " << i + 1 << ": ";

players[i].name = getValidStringInput();

cout << endl;

}

while (true) {

stageOne(players);

stageTwo(players);

checkForMinus(players);

waitForNextPlayer();

showStats(players);

cout << endl;

showRound();

waitForNextPlayer();

if (checkWinners(players)) {

showFinalResults(players);

cout << "\nИгра завершена! Спасибо за участие!" << endl;

return 0;

}

}

SetConsoleCP(866);

return 0;

}