#include <iostream>

#include <string>

#include <stdlib.h>

//#include <unistd.h>

using namespace std;

int multiLevel = 1;

int GlobalLevel = 1;

//Словарь слов для генерации строки

string dictionary[6][20] = {

{ "And","Fix","Own","Egg","Nod","Who","Far","Net","Way","Fat", "Joy", "Sun",

"Sky", "Toy", "Top", "Ant", "Bat", "Dog", "Cat", "Ear"},

{ "Bake","Word","List","Race","Rice","Lace","Beam","Game","Keep","Tree", "Fish",

"Lamb", "Bird", "Bear", "Deer", "Wolf", "Duck", "Frog", "Goat", "Lion"},

{ "About","Alert","Beach","Bread","Cause","Cross","Break","Chain","Crowd","Breed",

"Apple", "Books", "Chair", "Dance", "Earth", "Flute", "Grave", "Happy", "Impel", "Juice"},

{ "Beacon","Chisel","Garden","Helmet","Jungle","Kitten","Ladder","Mellow","Noodle","Rocket",

"Animal", "Bridge", "Candle", "Dragon", "Flower", "Grapes", "Hammer", "Island", "Jacket", "Kitchen"},

{ "Believe","Control","Delight","Example","Fiction","Journey","Laundry","Mystery","Respect","Teacher",

"Bicycle", "Cartoon", "Diamond", "Elephant", "Factory", "Galaxy", "Harvest", "Iceberg", "Justice", "Kingdom"},

{ "Absolute","Computer","Dinosaur","Elephant","Festival","Guardian","Hospital","Jealousy","Mediator","Notebook",

"Airplane", "Basement", "Champion", "Delicate", "Elephant", "Favorite", "Gigantic", "Heavenly", "Intrepid", "Jealousy"}

};

//Приветственное окно

void starting() {

setlocale(LC\_ALL, "Russian");

cout << "-------------------" << endl

<< " печать вслепую" << endl

<< "-------------------" << endl;

}

//Генерация строки

string decompose(int number) {

int startLength = number;

string result = "";

while (number > 8) {

int part = rand() % 6 + 3;

if (number - part - 1 >= 3 || number - part - 1 == 0) {

result += dictionary[part - 3][rand() % 20];

result += " ";

number -= part + 1;

}

}

if (number >= 3 && number <= 8) {

result += dictionary[number - 3][rand() % 20];

}

if (number < 3) {

return decompose(startLength);

}

else {

return result;

}

}

//создание метода split()

void split(const string& str, string result[100], int& wordCount) {

wordCount = 0;

string word;

for (char ch : str) {

if (ch == ' ') {

if (!word.empty()) {

result[wordCount] = word;

wordCount++;

word.clear();

}

}

else {

word += ch;

}

}

if (!word.empty()) {

result[wordCount] = word;

wordCount++;

}

}

//Добавляет в слова буквы при ошибке

string repeat(char inp, int amount) {

string res = "";

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

res += inp;

}

return res;

}

//Сравнение строк, добавление букв в случае ошибки(применяем функцию repeat())

string compareStrs(string str1, string str2) {

string result = "";

string word = "";

bool flag = true;

string w1[100];

string w2[100];

int l1, l2, i, j;

split(str1, w1, l1);

split(str2, w2, l2);

for (i = 0; i < l1; i++) {

flag = true;

word = "";

if (GlobalLevel < 3) {

for (j = 0; j < size(w1[i]); j++) {

if (w1[i][j] != w2[i][j]) {

word += repeat(w1[i][j], multiLevel \* 2);

flag = false;

}

else {

word += w1[i][j];

}

}

}

//начиная с 3 уровня

else {

for (j = 0; j < size(w1[i]); j++) {

if (w1[i][j] != w2[i][j]) {

if (j == size(w1[i]) - 1) {

/\*word += repeat(w1[i][j - 1], multiLevel);\*/

word += repeat(w1[i][j], multiLevel \* 2);

flag = false;

}

else {

word += repeat(w1[i][j], multiLevel);

word += repeat(w1[i][j + 1], multiLevel);

flag = false;

}

}

else {

word += w1[i][j];

}

}

}

if (flag) {

if (size(w1[i]) > 8) {

result += decompose(size(word));

}

else {

result += dictionary[size(w1[i]) - 3][rand() % 20];

}

}

else {

result += word;

}

if (i < l1 - 1) {

result += " ";

}

}

return result;

}

//Проверка строки согласно условию

int sentenceChecker(string FirstVal) {

bool flag = true;

string UserVal;

int number = size(FirstVal);

//когда длина заполнила экран игра заканчивается

if (number > 100) {

cout << FirstVal << endl << endl;

cout << "Вы проиграли :(" << endl;

exit(0);

}

cout << endl << FirstVal << endl;

getline(cin, UserVal);

//заверешние программы при вводе 13

if (UserVal == "13") {

cout << "Завершение тренировки..." << endl;

return 0;

}

//проверка на длину(если длина не совпала, вывести строку заново)

if (!(size(UserVal) == size(FirstVal))) {

cout << "Неверная длина" << endl;

sentenceChecker(FirstVal);

}

//поиск ошибок в пользовательской строке

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

if (FirstVal[i] == UserVal[i]) {

continue;

}

else {

flag = false;

}

//проверка на пробелы(в случае ввода фигни вызывать функцию с той же строкой)

if ((FirstVal[i] == ' ' && UserVal[i] != ' ') || (FirstVal[i] != ' ' && UserVal[i] == ' ')) {

sentenceChecker(FirstVal);

}

}

//переход на новый уровень

if (number - 2 < 3 && flag == true) {

system("cls");

multiLevel \*= 2;

GlobalLevel += 1;

cout << "Новый уровень!" << '\n' << GlobalLevel << endl;

sentenceChecker(decompose(20));

return 0;

}

//все верно, уменьшить длину на 2

if (flag) {

sentenceChecker(decompose(number - 2));

}

//есть ошибки, генерируем новую строку

else {

sentenceChecker(compareStrs(FirstVal, UserVal));

}

}

// Главная функция

int main() {

srand(time(0));

starting();

setlocale(LC\_ALL, "Russian");

cout << "Чтобы начать нажмите клавишу Enter...";

cin.ignore();

sentenceChecker(decompose(20));

return 0;

}