МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ

Кафедра інформатики

ПОЯСНЮВАЛЬНА ЗАПИСКА

до курсової роботи

на тему: Розробка тестової програми «Орфографія»

з дисципліни «Програмування»

спеціальності 123 «Комп’ютерна інженерія»

Керівники \_\_\_\_\_\_\_\_\_\_\_ Бєлова Н.В.

(підпис) (прізвище, ініціали)

\_\_\_\_\_\_\_\_\_\_\_ Руденко Д.О.

(підпис) (прізвище, ініціали)

\_\_\_\_\_\_\_\_\_\_\_ Сінельнікова Т.Ф.

(підпис) (прізвище, ініціали)

Студент гр. \_\_\_\_\_\_\_\_\_\_\_\_ Анікаєв Р.О.

КІУКІу21-2  (підпис) (прізвище, ініціали)

2021 рік

РЕФЕРАТ

Пояснювальна записка курсового проєкту має: 22 с., 6 рис., 5 джерел, 2 додатки.

Об’єктом дослідження є аналіз функцій і методів роботи із зовнішніми файлами, для тестування користувача та виведення результатів у файл. Зокрема створення консольної програми проходження диктанту, та запис результатів в зовнішній файл, на мові програмування С++, в середовищі VS.

Метою курсового проєктування э розробка тестової програми для перевірки студентів на орфографію.

У результаті виконання курсової роботи була розроблена консольна програма проходження диктанту для студентів, та перевірки правильності введення слів, з виведенням результатів у зовнішній файл.

АЛГОРИТМ, ПРОГРАМУВАННЯ, PROCEDURE, PROJECT, ТЕСТУВАННЯ, ДИКТАНТ, ФАЙЛ, ОРФОГРАФІЯ

ЗМІСТ

[ВСТУП 4](#_Toc90239514)

[1 ОСНОВНІ ТЕОРЕТИЧНІ ВІДОМОСТІ 7](#_Toc90239515)

[1.1 Мова програмування С++ 7](#_Toc90239516)

[1.2 Використання функцій в С++ 9](#_Toc90239517)

[2 ПРОЄКТУВАННЯ ПРОГРАМНОЇ СИСТЕМИ 12](#_Toc90239518)

[2.1 Алгоритм побудови програми 12](#_Toc90239519)

[2.2 Тестування програми 13](#_Toc90239520)

[ВИСНОВКИ 16](#_Toc90239521)

[ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ 17](#_Toc90239522)

[ДОДАТОК A 18](#_Toc90239523)

[ДОДАТОК Б 20](#_Toc90239524)

ВСТУП

Орфографія - система загальноприйнятих правил, що визначають способи передачі мови в писемній формі.

Як синонім орфографії вживається термін правопис, який, проте, має ширше значення, оскільки включає в себе й пунктуацію.

Орфографія і графіка тісно пов'язані між собою, але не тотожні. Графіка є засобом, інвентарем орфографії, а орфографія — це правила вживання графічних засобів.

Як, за уявленнями давніх людей, Земля лежить на трьох китах, так українська орфографія базується на трьох принципах: фонетичному, морфологічному та історичному, або традиційному.

Орфографічна грамотність учнів – це одна з актуальних проблем, що стоїть перед школою протягом усього її історичного поступу. Але попри її вік, проблема орфографічною грамотності досі не вирішена. Так, орфографічно правильне письмо передбачає вміння знаходити, впізнавати явища мови з урахуванням, так званого орфографічного досвіду, який допомагає зупинитися, замислитися, перевірити себе, коли це необхідно.

Проблема формування орфографічної грамотності має високий рівень загальної дослідженості. Так, важливу роль у становленні методики орфографії відіграли праці Н. М. Зорька, М. Р. Львова, Г. М. Приступи, М. М. Разумовської, та інших. На жаль, не в повному обсязі вчителі та викладачі розуміють правильно зміст цього. Тому орфографічна грамотність учнів і студентів низька через недостачу сформованості орфографічної навички.

Диктанти – активна форма тренувальних вправ, яка вимагає уваги, зосередженості, активної розумової діяльності. Ми використовуємо їх для роботи на уроці під час вивчення теми, а також для контролю за засвоєнням навчального матеріалу з граматики та правопису.

Диктант – письмова робота, вправа для перевірки грамотності учнів і засвоєння ними правил правопису (орфографії і пунктуації).

Якщо розглядати диктант з історичного погляду, то побачимо поступову диференціацію, урізноманітнення його видів. Проте цей процес не був рівномірним. Різним було ставлення до диктантів – від надто частого їх використання до цілковитого заперечення, змінювалась оцінка їх ролі у навчальному процесі.

Оцінюючи навчальну ефективність диктантів, потрібно мати на увазі конкретні їх форми та способи їх реалізації. Людині надзвичайно часто доводиться, окрім списування з готового тексту, формулювання власних думок, записувати сприйняте на слух, почуте від когось. Практичні потреби життя вимагають відповідних навичок передавати на письмі текстуальне, вибірково чи стисло усне мовлення, сприйняте на слух. Оскільки між усною мовою і писемною, між звуками в їх системі, в живому потоці мовлення і засобами їх передачі на письмі існують складні, неоднозначні відношення, необхідно створити зв'язок між слуховим сприйманням і графічним образом слова, між вимовою і його написанням. Цього можна досягти лише відповідними вправами.

Диктант, якщо вміло проводити і застосовувати різні їх види, є одним із засобів боротьби за культуру писемної мови, вони виконують організуючу і виховну роль, допомагають сформувати професійні навики регламентованої роботи, вміння дотримуватись встановленого порядку і ритму. У той же час сама передача на письмі вимагає напруження і зосередження уваги. Тобто, це один із засобів виховання уваги, дисциплінованості, організованості, точності та охайності в навчальній діяльності, вміння включатись в розумову роботу і виконувати завдання в даний момент.

У методиці й практиці навчання існує багато різних видів диктантів, кількість яких поступово збільшується. Різні методисти користуються неоднаковою класифікацією диктантів. Так, В. Т. Горбачук класифікує диктанти, враховуючи такі три критерії: мета проведення, характер запису диктованого тексту, час пояснення правопису орфограм у тексті.

Таким чином, існує чимало підходів щодо класифікації й систематизації диктантів, однак у процесі роботи для себе визначила 4 аспекти, які можна взяти за основу:

* за призначенням: навчальні диктанти, навчально-перевірні, перевірні (контрольні);
* за часом запобігання помилок і закріплення граматичного та правописного матеріалу: попереджувальні, пояснювальні;
* за характером відтворення на письмі: без зміни тексту (диктанти-завдання, диктанти-переклади), зі зміною тексту (вибіркові, словникові, творчі, вільні);
* за сприйманням тексту: зорово-слухові, слухово-зорові, слухові, зорові (диктант з пам'яті).

В цій програмі використано саме диктант за сприйманням тексту. По проходженню нього буде перевірено зорову пам’ять користувача, перевіривши правильність запам’ятованих слів із словника.

Так як програма потребує різнобічної доробки, то на теперішній час її можна використовувати, наприклад:

* проводити диктанти у навчальних закладах, на заняттях, щоб дізнатися як добре засвоївся новий матеріал;
* при доробці програми , можна і використовувати на контрольних опитуваннях та заліках (змінивши при цьому вид диктанту).

1 ОСНОВНІ ТЕОРЕТИЧНІ ВІДОМОСТІ

1.1 Мова програмування С++

С++ є результатом еволюції мови С, створеної Денісом Рітчі та Брайаном Керніганом. Ще на початку 70-х років минулого сторіччя С набула широкої популярності як мова розробляння операційної системи UNIX. Сьогодні більшість операційних систем написано мовою С та/чи С++. С є доступною для більшості систем, до того ж вона є машинонезалежною.

Широке використання С на комп’ютерах різних апаратних платформ призвело до появи різноманітних варіацій мови. Це зумовило потребу стандартизації мови і 1990 року відповідний документ стандарту ANSI/ISO 9899:1990 було розроблено.

Мову С++ розробив Бьєрн Страуструп на початку 80-х років минулого сторіччя. Порівняно з С, мова С++ набула багато “прикрас”, головною з яких є можливість об’єктно-орієнтованого програмування. Об’єкти є за своєю суттю програмними компонентами, які моделюють предмети реального світу. Модульний, об’єктно-орієнтований підхід до проєктування та реалізації програмного забезпечення дозволили полегшити написання складних потужних програмних проєктів.

Об’єктно-орієнтовані програми є набагато простіші для розуміння, виправлення та модифікації.1991 року на основі С і С++ фірмою Sun, а саме Джеймсом Гослінгом, було створено мову Java. Метою Java на початку було керування інтелектуальною побутовою технікою, а згодом, завдячуючи шаленому вибухові популярності Word Wide Web, її швидко було перекваліфіковано і представлено 1995 року як потужний засіб розробляння програм для розв’язування різноманітних сучасних задач.

Сьогодні Java використовується для створювання динамічного й інтерактивного вмісту Web-сторінок, розширювання можливостей Webсерверів, створювання програм для мобільних телефонів та побутових пристроїв, розробляння широкомасштабних програмних додатків на підприємствах тощо. Подібні можливості мають сучасні версії С++, як-от Microsoft® Visual C++®.

NET та Borland® C++ BuilderТМ. На основі С++ і Java для платформи .NET було розроблено мову С#, яка, подібно до Microsoft® Visual C++®.NET та Visual Basic.NET, дозволяє інтегрувати в програмні додатки можливості Internet та Web, писати компоненти програмного забезпечення мовою, яку знає розроблювач, а потім будувати додатки, комбінуючи ці компоненти з компонентами, написаними будь-якою іншою .NET мовою.

Еволюція C++ Builder розпочинається з 1997 року, і майже кожного року випускаються оновлені й оптимізовані версії. Приміром, версія Borland C++ Builder Enterprise 6.0 for Windows NT\2000\XP + Update є потужним і якісним інструментом реалізації RAD-стилю (Rapid Application Development – швидке розробляння додатків) стосовно самих різних класів задач, включаючи роботу з Internet, ActiveX, з проєктами рівня підприємств, і все це – з використанням стандартного C++.

Borland C++ Builder тісно інтегрований з обома базовими технологіями створення розподілених систем – CORBA та COM. Borland  C++ Builder сумісний з Visual C++. Підтримує більшість технологій сучасної розробки. 2005 року Borland випустила Borland Developer Studio 2006, який включає у себе Borland C++ Builder 2006 з оптимізованим керуванням конфігурацією та налагодження. Borland Developer Studio 2006 – єдиний повноцінний комплект, який містить Delphi, C++Builder та C#Builder. 2007 року CodeGear випустила C++ Builder 2007, в якому реалізувала повну підтримку API Microsoft Windows Vista, збільшила повноту відповідності стандартові ANSI C++, прискорила розробку до 500 %, долучила підтримку MSBuild, архітектур баз даних DBX4 та “VCL для Web”, яка підтримує AJAX.

Підтримка API Microsoft Windows Vista розмітила в собі додатки, оформлені у стилі Vista, і підтримку VCL для Aero та Vista Desktop. CodeGear RAD Studio 2007 містить C++ Builder 2007 та Delphi. Також 2007 року CodeGear відродила марку “Turbo” і випустила дві “Turbo” версії C++ Builder: Turbo C++ Professional та Turbo C++ Explorer (безкоштовна), основаних на Borland C++ Builder 2006. Наприкінці 2008 року компанія CodeGear випустила нову версію RAD Studio, до якої увійшли Delphi 2009 та С++ Builder 2009.

С++ – потужна, лаконічна, гнучка, мобільна і жива мова, яка розвивається і в різних своїх версіях набуває нових сучасних можливостей.

1.2 Використання функцій в С++

Функція (function) – це підпрограма, яка може отримати дані та повертати деяке значення. Кожна функція має своє власне ім'я. Коли це ім'я зі списком даних (параметрів) зустрічається будь-де у програмі, виконання програми переходить до тіла цієї функції. Такий перехід має назву виклику функції (function call). Після того, як були виконані інструкції в тілі функції, процес виконання повертається до попередньої функції (з якої було здійснено виклик) та виконуються її наступні інструкції. Складні програми повинні бути розділені на декілька функцій, виклик яких здійснюється по черзі.

Виклик функції main() здійснюється операційною системою.

Під час виклику функцій усі їх локальні змінні, адреси функцій, а також значення параметрів і інші локальні дані розміщуються в спеціальній області пам'яті, що зветься програмним стеком, або стеком викликів (call stack). Під час повернення з функції область, зайнята її локальними даними, звільняється.

Визначення функції включає тіло функції - блок з програмним кодом, який виконується під час виклику функції.

Оголошення функції без визначення має назву прототипу (prototype). Прототип функції складається з типу результату функції, імені та списку параметрів. Список параметрів (parameter list) – це список всіх аргументів та їх типів, розділених комами.

int sum(int a, int b); // прототип

Прототип функції та визначення функції повинні збігатися щодо типу результату, імені та списку параметрів. Прототип функції не обов'язково містить імена параметрів.

int sum(int, int); // прототип

Визначення функції (function definition) складається з заголовка функції і тіла. Тіло функції (function body) – це набір інструкцій, укладених у фігурні дужки.

int sum(int a, int b)

{

return a + b;

}

Усі функції мають тип результату. Інструкція return завершує виконання функції та повертає управління до попередньої функції, з якої було здійснено виклик. Інструкція return всередині функції main() передає управління операційній системі, а результат, який повертає функція main(), може бути використаний як код помилки (0 - помилок немає).

Значення виразу в інструкції return повертається до функції, з якої було здійснено виклик. Функції усіх типів (за винятком void) повинні містити інструкцію return з таким виразом. Інструкція return завжди завершує виконання функції.

Функцію не можна визначити всередині іншої функції. У C++ немає локальних функцій.

Незважаючи на те, що немає обмеження на розмір функції, добре спроєктовані функції зазвичай містять не дуже багато коду. Функцію з меншим розміром легше зрозуміти та підтримувати її роботу.

Аргументи передаються у функцію в тому порядку, в якому вони були оголошені і визначені. Будь-який вираз C++ може бути аргументом функції, в тому числі константа, математичні та логічні вирази, виклик інших функцій, які повертають значення.

2 ПРОЄКТУВАННЯ ПРОГРАМНОЇ СИСТЕМИ

2.1 Алгоритм побудови програми

При виконанні курсової роботи була розроблена поетапна схема виконання проєкту. Для початку створювався сам проєкт “ Coursework\_Anikaev”, програма “Орфографія”, його вигляд, меню, головна функція, та функції обробки. На рисунку 2.1 побудовані етапи створення проєкту.

Визначення проблеми завдання

Написання головної функції та інтерфейсу програми

Написання функції для виведення словника

Створення програми для диктанту

Написання функції для тестування

Написання функції для введення результатів у файл

Написання функції для виведення результатів

Тестування програми

Удосконалення програми

Рисунок 2.1 – Алгоритм побудови програми

Реалізація програми, показана в додатку А, та додатку Б.

2.2 Тестування програми

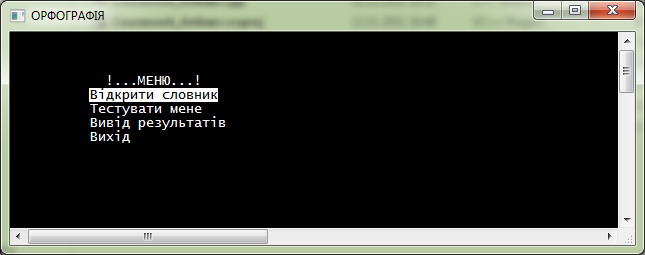
Запускається програма за допомогою .exe файла – Coursework\_Anikaev.exe. Запущену програми можна побачити на рисунку 2.2.

Рисунок 2.2 – Запущена програма

При запуску, на монітор виводиться меню програми “Орфографія”. Першим пунктом меню є “словник”. Для запуску, вибираємо пункт і нажимаємо ‘Enter’. Показано на рисунку 2.3.

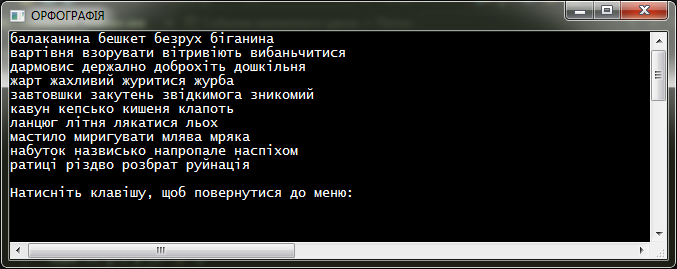


Рисунок 2.3 – Пункт 1 “Словник”

Далі, щоб пройти тестування (написати диктант), вибираємо пункт два “Тестувати мене”. Вводимо ім’я користувача. Пишемо по пам’яті 10 слів, нажимаємо Enter. Вікно тестування можна побачити на рисунку 2.4.

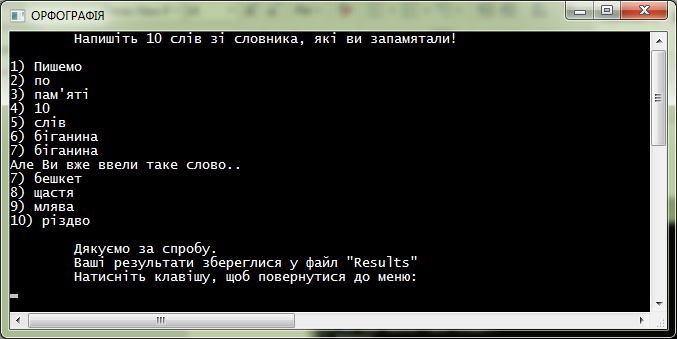


Рисунок 2.4 – Пункт 2 “Тестувати мене”

Далі, результати тестування виводяться у зовнішній файл “Results”. Показано на рисунку 2.5. Щоб отримати результати тестування, треба вибрати третій пункт меню “Вивід результатів”. Подивитися можна на рисунку 2.6.

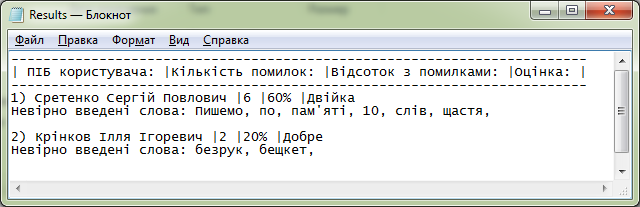


Рисунок 2.5 – Зовнішній файл “Results”

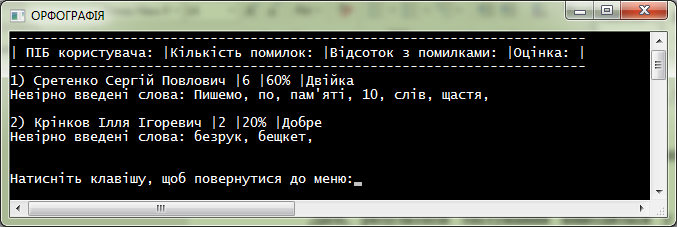


Рисунок 2.6 – Пункт меню “Вивід результатів”

Останнім пунктом меню являється кнопка “Вихід”, для виходу з програми.

ВИСНОВКИ

Для розробки проєкту були проаналізовані наступні різновиди диктантів:

* за призначенням: навчальні диктанти, навчально-перевірні, перевірні (контрольні);
* за часом запобігання помилок і закріплення граматичного та правописного матеріалу: попереджувальні, пояснювальні;
* за характером відтворення на письмі: без зміни тексту (диктанти-завдання, диктанти-переклади), зі зміною тексту (вибіркові, словникові, творчі, вільні);
* за сприйманням тексту: зорово-слухові, слухово-зорові, слухові, зорові (диктант з пам'яті).

З виконанням курсової роботи, був створений проєкт “Coursework\_Anikaev”, програма “Орфографія” із функцією “checks()”, що вмикається вибором пункту меню програми – “Тестувати мене”, яка проводить диктант за сприйманням тексту, та виводить результати написання у файл “Results”.

Програма має деякі недоліки. Програма тестує лише 10 введених слів, та має обмежену кількість слів у словнику.

Ця тестова програма може бути корисною лише на незначних ‘зрізі знань’, на деяких парах, для визначенні правильності вивчених нових слів. Але при удосконаленні програми, її можна використовувати в різних закладах, для проведення орфографічних диктантів.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Страуструп Б. Язык программирования C++, 3-е изд./ Пер. с англ. СПб.; М.: ”Невский Диалект” – “Издательство БИНОМ”,1999 г. – 991 с.
2. Х.М.Дейтел, П.Дж. Дейтел. Как программировать на С++.- М.:ЗАО «Издательство БИНОМ», 2000 г. – 1024 с.
3. Язык программирования С ++. Б'ярн Страуструп, 1997 г.
4. https://filologukraine.ucoz.ua/publ/mova/sulm/orfografija\_ukrajinskoji\_movi/3-1-0-22
5. http://iwanoff.inf.ua/programming\_1\_ua/LabTraining03.html

ДОДАТОК A

Код з файлу ‘Coursework\_Anikaev.cpp’

#include "Header.h"

int main() {

system("title ОРФОГРАФІЯ");//назва программи

//Українська мова

SetConsoleCP(1251); //Встановити в консолі шрифт Lucida Console!!!

SetConsoleOutputCP(1251);

vuvod\_1();//Заголовок файлу з результатами

HANDLE hStdOut = GetStdHandle(STD\_OUTPUT\_HANDLE);//щоб отримати //дескриптор стандартного пристрою введення/виводу

int activeColor = BACKGROUND\_RED | BACKGROUND\_GREEN | BACKGROUND\_BLUE | BACKGROUND\_INTENSITY;

int defaultColor = FOREGROUND\_BLUE | FOREGROUND\_GREEN | FOREGROUND\_RED | FOREGROUND\_INTENSITY;

const int stringsSize = 5;

const char\* strings[stringsSize] = { " !...МЕНЮ...! ", "Вiдкрити словник", "Тестувати мене", "Вивід результатів", "Вихiд" };

COORD pos;//Структура. Визначає координати символьної комірки у буфері //екрана консолі.

COORD bufferSize = { 180, 50 };

SetConsoleScreenBufferSize(hStdOut, bufferSize);

int activeItem = 1;//активний перший пункт меню

int parti = 0;//кількість участників тестування

while (1) {//зациклювання

for (int i = 0; i < stringsSize; i++) {//виведення меню

if (i == activeItem)

SetConsoleTextAttribute(hStdOut, activeColor);//колір //активного пункту меню

else

SetConsoleTextAttribute(hStdOut, defaultColor);//колір //неактивного пункту меню

pos.X = 10;

pos.Y = 3 + i;

SetConsoleCursorPosition(hStdOut, pos);//встановлює положення

cout << strings[i];

}

int code = \_getch();

switch (code) {

case 72: {//up

activeItem--;

if (activeItem < 0) activeItem = 0;

}break;

case 80: {//down

activeItem++;

if (activeItem > stringsSize - 1) activeItem = stringsSize - 1;

}break;

case 13: {//Enter

if (1 == activeItem) {

vuvod\_words();//Виведення словника на консоль

}

if (2 == activeItem) {

checks();//тестування

parti += 1;//+1 участник

}

if (3 == activeItem) {

vuvod\_result(parti);//Виведення результатів, якщо

}

if (4 == activeItem) {

return 0;

}

}break;

}

}

system("pause");

return 0;

}

ДОДАТОК Б

Код з файлу ‘Header.h’

#pragma once//директива препроцесора, включення лише один раз

#include <iostream>

#include <string>

#include <windows.h>

#include <conio.h>//\_getch();

#include <fstream>

using namespace std;

int candidats = 0;

void vuvod\_1()//Заголовок файлу з результатами

{

char odin[120] = "------------------------------------------------------------------------";

char dva[120] = "| ПІБ користувача: |Кількість помилок: |Відсоток з помилками: |Оцінка: |";

char tri[120] = "------------------------------------------------------------------------";

ofstream ptr\_f("Results.txt"); // створюємо об'єкт класу ofstream для запису та //пов'язуємо його з файлом Results.txt

if(ptr\_f.is\_open())

ptr\_f << odin << endl << dva << endl << tri << endl;// запис рядка у файл

ptr\_f.close(); // закрываем файл\*/

}

void vuvod\_words()//Виведення словника на консоль

{

system("cls");//очистити вікно

ifstream ptr\_f("Dictionary.txt", ios::in);

cout << ptr\_f.rdbuf();

ptr\_f.close();

cout << "\n\nНатисніть клавішу, щоб повернутися до меню:";

\_getch();

system("cls");

}

void checks()

{

candidats += 1;//Плюс один учасник

//=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=

//зчитення словника

system("cls");

char vuv0[40][100];

string vuv1;//для зчитування рядків із файлу

int i = 0;

ifstream ptr\_f("Dictionary.txt", ios::in);

while (!ptr\_f.eof()) {

ptr\_f >> vuv1;

strncpy\_s(vuv0[i++], vuv1.c\_str(), 100);

}

//=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=

//введення данних

char name[100];//повне ім'я

cout << "\tВведіть ПІБ для тестування:\n\t";

cin.getline(name, 100);

cout << "\tХай щастить!)";

Sleep(1000);//затримка на 1с.

system("cls");

char text1[10][20];//введені слова

cout << "\tНапишіть 10 слів зі словника, які ви запамятали!\n\n1) ";

cin >> text1[0];

int lent = 0;

for (int j = 1; j < 10; j++) {

cout << j + 1 << ") ";

cin >> text1[j];

//dyplic:

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

lent = sizeof(text1[i]);//довжина слова

if (strncmp(text1[j], text1[i], lent) == 0) {

cout << "Але Ви вже ввели таке слово..\n";

cout << j + 1 << ") ";

cin >> text1[j];

i = -1;

}

}

}

//=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=

float max = 0;

int ri1 = 0;// кількість правильних слів /якщо правилне -> ri1 = 1

char text2[255]="";// Рядок з неправильними словами

int len;

//підрахунок кількості правильних слів

for (int j = 0; j < 10; j++) {//кожен з 10 введенних слів

for (int k = 0; k < 40; k++) {//кожен з 40 слів словника

len = sizeof(vuv0[k]);//довжина слова

if (strncmp(text1[j], vuv0[k], len) == 0) {

//cout << j;

max += 1;

ri1 = 1;

break;

}

}

if(ri1==0) {//якщо слово не співпало

//додаємо неправильне слово в рядок

strcat\_s(text2, text1[j]);

strcat\_s(text2, ", ");

}

ri1 = 0;

}

//cout << endl << max << endl << text2;

cout << "\n\tДякуємо за спробу.\n\tВаші результати збереглися у файл \"Results\"";

cout << "\n\tНатисніть клавішу, щоб повернутися до меню:";

//=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-

//запис у файл результата

float mist, pers;

mist = 10 - max;//неправильних слів

pers = (mist \* 100)/10;//відсоток непрв слів

string gld;

if (pers <= 10) gld = "Відмінно";

else if (pers <= 20) gld = "Добре";

else if (pers <= 30) gld = "Задовільно";

else gld = "Двійка";

if (max == 10) strcat\_s(text2, " - ");

ofstream ptr\_f1("Results.txt", ios\_base::app);//відкрити файл для запису в кінець //файлу

if (ptr\_f1.is\_open()) {

ptr\_f1 << candidats << ") " << name << " |" << mist << " |" << pers << "% |" << gld << endl;

ptr\_f1 << "Невірно введені слова: " << text2 << endl << endl;

}

ptr\_f1.close();

cin.get();//щоб наступному рядку не присвоювалося '\n'

cout << endl;

\_getch();

system("cls");

}

void vuvod\_result(int parti) {//вивід результатів на єкран

system("cls");

if (parti == 0) {

cout << "Вибачте, але Ви ще не проходили тестування!\n";

cout << "Поверніться до головного меню, пройдіть тест, та дивіться свої результати.";

cout << "\nРезультати тестування Вы можете побачити в цьому пункті,\n";

cout << "або в файлі \"Results\".";

}

else {

ifstream ptr\_f("Results.txt", ios::in);

cout << ptr\_f.rdbuf();

ptr\_f.close();

}

cout << "\nНатисніть клавішу, щоб повернутися до меню:";

\_getch();

system("cls");

}