**МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ**

**НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ “ЛЬВІВСЬКА ПОЛІТЕХНІКА”**

**КАФЕДРА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ**

# **КУРСОВА РОБОТА**

з дисципліни «Об'єктно орієнтоване програмування»

на тему: «Аналіз тексту»

Студента ПЗ-24

спеціальності 6.121

«Інженерія програмного забезпечення»

Бовчалюк О.В.

Науковий керівник

доцент кафедри ПЗ, доцент, к.т.н.

Коротєєва Т.О.

Національна шкала \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Кількість балів: \_\_\_\_\_\_\_\_\_\_Оцінка: ECTS \_\_\_\_\_

Члени комісії \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис) (прізвище та ініціали)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис) (прізвище та ініціали)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис) (прізвище та ініціали

Львів — 2018 рік

**Зміст**

1) Технічне завдання стор.3

2) UML діаграма класів стор.4

3) Діаграма прецедентів стор.5

4) Діаграма послідовностей стор.6

5) Алгоритм виконання стор.7

5) Протокол роботи стор.8

6) Опис виняткових ситуацій стор.12

7) Інструкція користувача стор.13

8) Код програми стор.14

**Технічне завдання для курсової роботи**

**з дисципліни «Об‘єктно-орієнтоване програмування»**

Розробити програму засобами ООП на мові С++ згідно вказаного варіанту.

Продемонструвати викладачу роботу розробленої програми.

Сформувати звіт з курсової роботи (не менше 20 стор.). Звіт має містити: 1) завдання; 2) алгоритм розв‘язку задачі; 3) діаграми UML [клас](http://vns.lpnu.ua/mod/glossary/showentry.php?eid=54317&displayformat=dictionary)ів, послідовності виконання, взаємозв‘язків; 4) текст розробленої програми з коментарями; 5) протокол роботи програми для кожного пункту завдання; 6) інструкція користувача. Інтерфейс програми  українською мовою; 7) компакт-диск з записаною інформацією (звіт і програма).

Програма має передбачати ввід даних в двох режимах ( з клавіатури та з файлу).

Передбачити виняткові ситуації.

**Варіант 7**

Написати програму для аналізу текстів. Тексти можуть вводитися користувачем через клавіатуру або ж зчитуватися з файлу. Програма повинна мати наступну функціональність:

1) Виводити відсортовану по кількості таблицю з літерами латинського алфавіту у форматі :

Назва літери | Кількість у тексті

Літери, які не зустрічаються у тексті вносити у таблицю не потрібно.

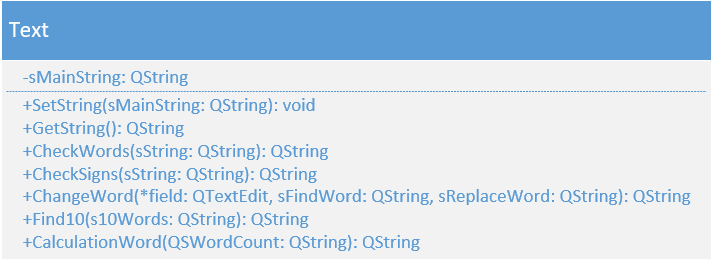
2) Реалізувати функціонал описаний в першому пункті, але для всіх символів, які зустрічаються в тексті

3) Реалізувати можливість калькуляції слів у тексті. Словом вважати будь-яку послідовність символів, відділену знаком пробілу

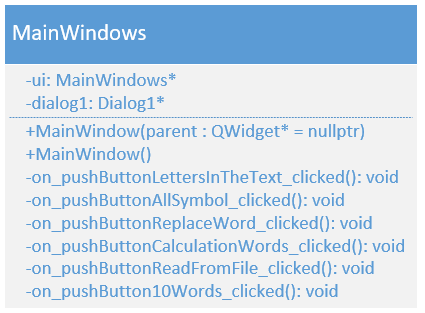
4) Відобразити десять найбільш вживаних у тексті слів

5) Реалізувати функціонал заміни одного слова на інше у тексті, з можливістю збереження модифікованого тексту у новий файл

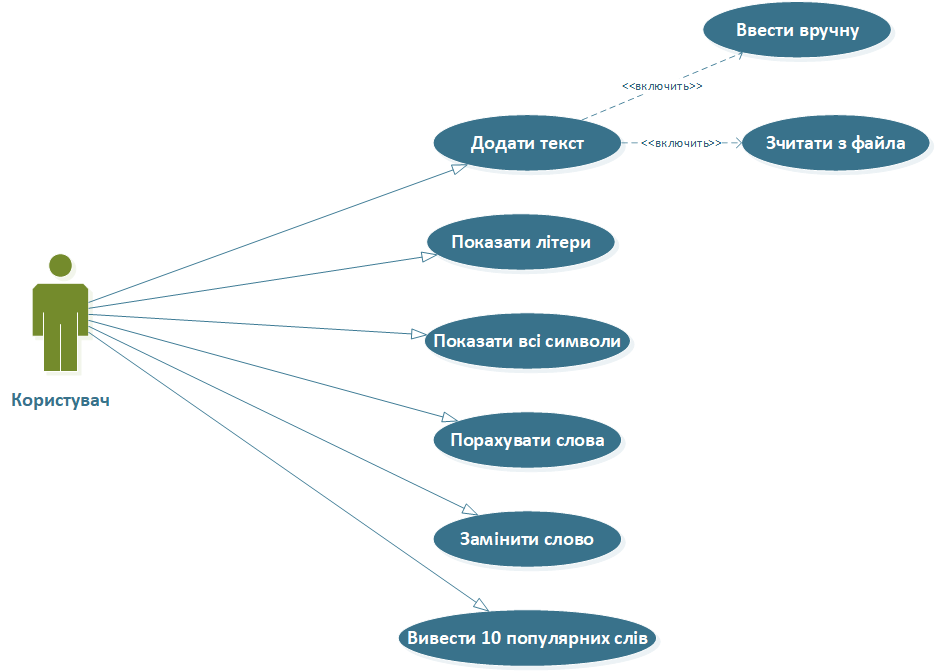
**UML Діаграми класів**



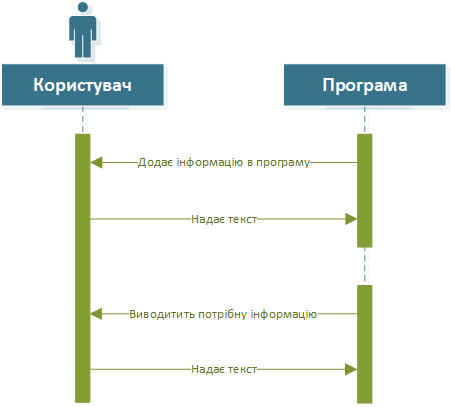




**Діаграма прецедентів**



**Діаграма послідовностей**



**Алгоритм виконання**

**Алгоритм А:**

**А1:** Відкрити файл або ввести вручну і відобразити дані.

**А2:** Літери які зустрічаються в тексті.

**А3:** Всі символи в тексті.

**А4:** Калькуляція слів.

**А5:** Заміна тексту.

**А6:** 10 популярних слів.

**Алгоритм функції А1**

**А1.1:** Зчитати шлях до файлу.

**А1.2:** Відкрити файл.

**A1.3:** Виконати додавання даних з файлу.

**Алгоритм функції А2**

**A2.1:** Зчитати текст.

**A2.2:** Порахувати літери які зустрічаються.

**A2.3:** Посортувати по кількості.

**A2.4:** Відобразити дані.

**Алгоритм функції А3**

**A3.1:** Зчитати текст.

**A3.2:** Порахувати всі символи які зустрічаються.

**A3.3:** Посортувати по кількості.

**A3.4:** Відобразити дані.

**Алгоритм функції А4**

**A4.1:** Зчитати текст.

**A4.2:** Порахувати кількість слів відділених пробілами.

**A4.3:** Відобразити дані.

**Алгоритм функції А5**

**A5.1:** Відкрити діалогове вікно.

**A5.2:** Зчитати слово яке найти та яке замінити.

**A5.3:** Провести заміну слів.

**A5.4:** Зберегти текст в новий файл.

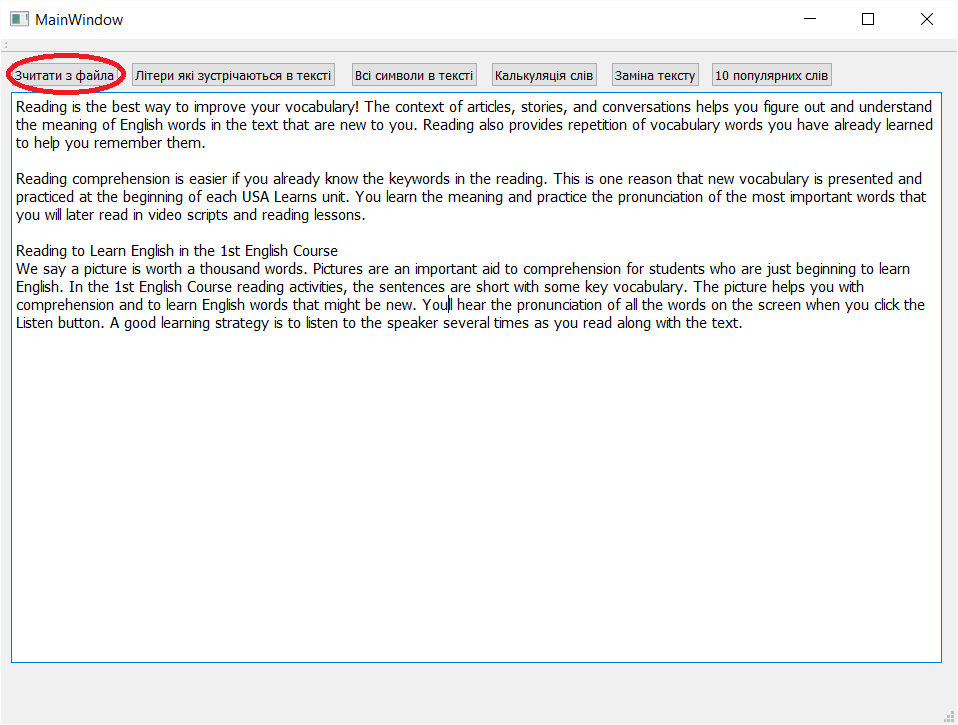
**Алгоритм функції А6**

**А1.1:** Зчитати текст.

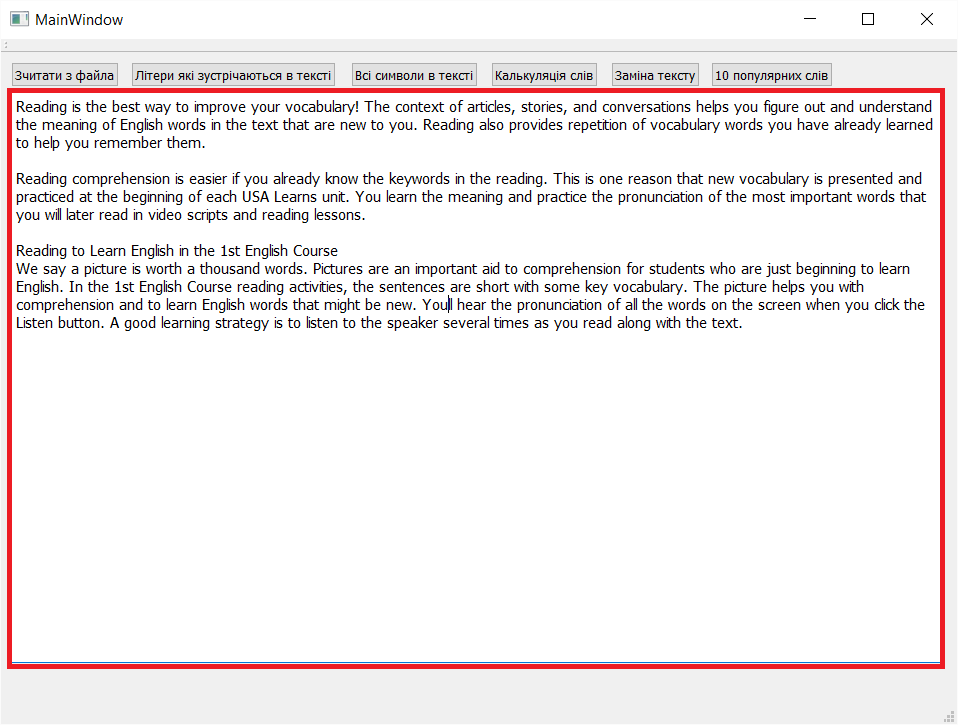
**А1.2:** Порахувати кожне слово в тексті.

**A1.3:** Вивести 10 популярних слів.

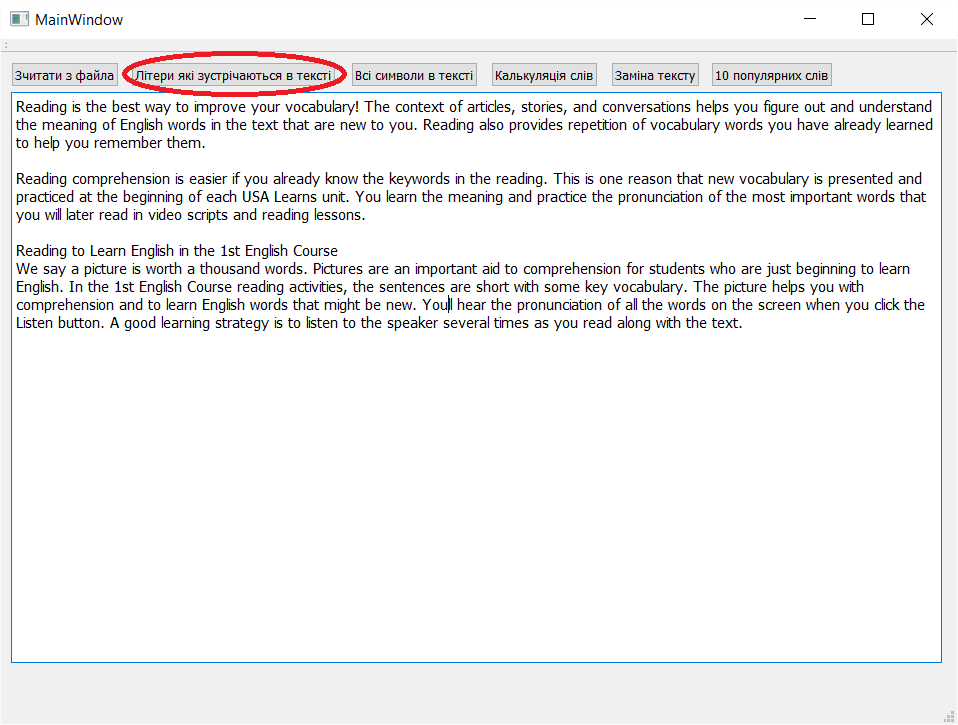
**Протокол роботи**



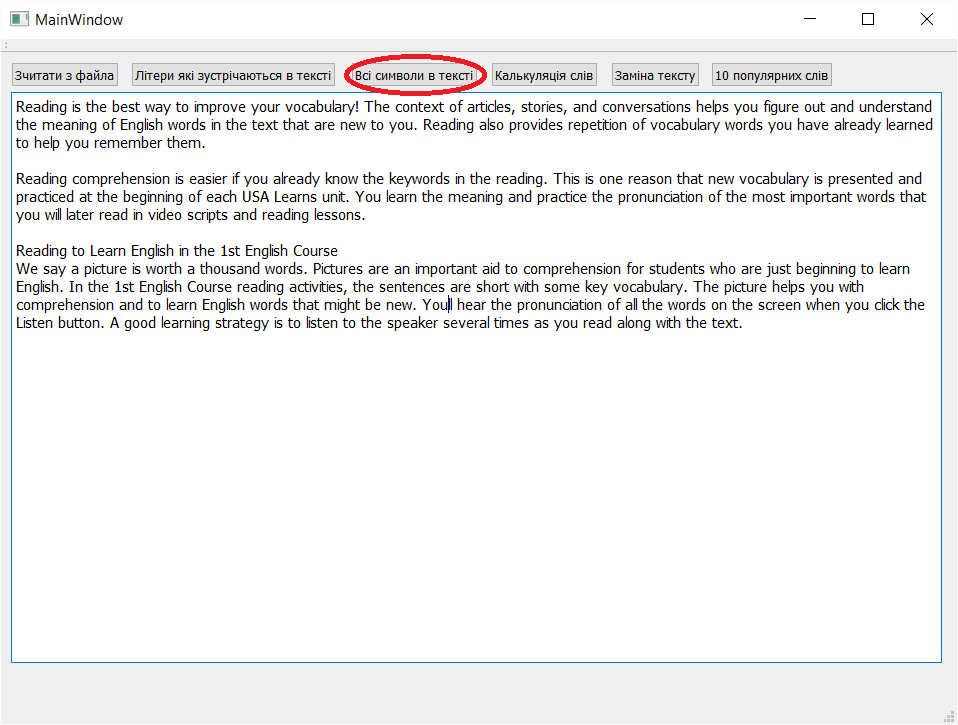
Введення з файлу

****

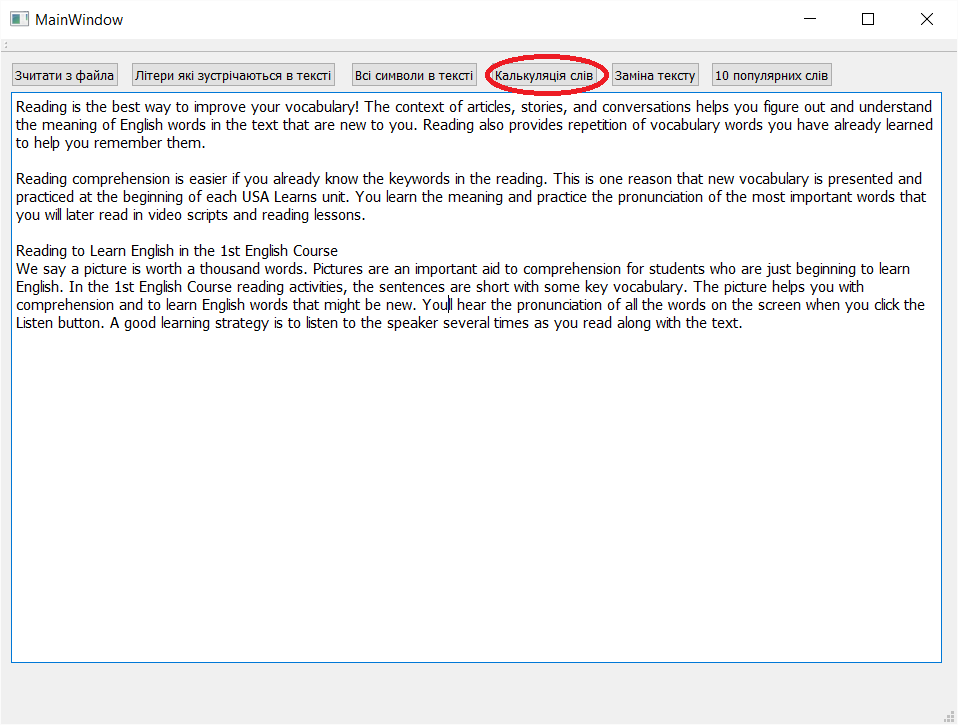
Введення з клавіатури

****

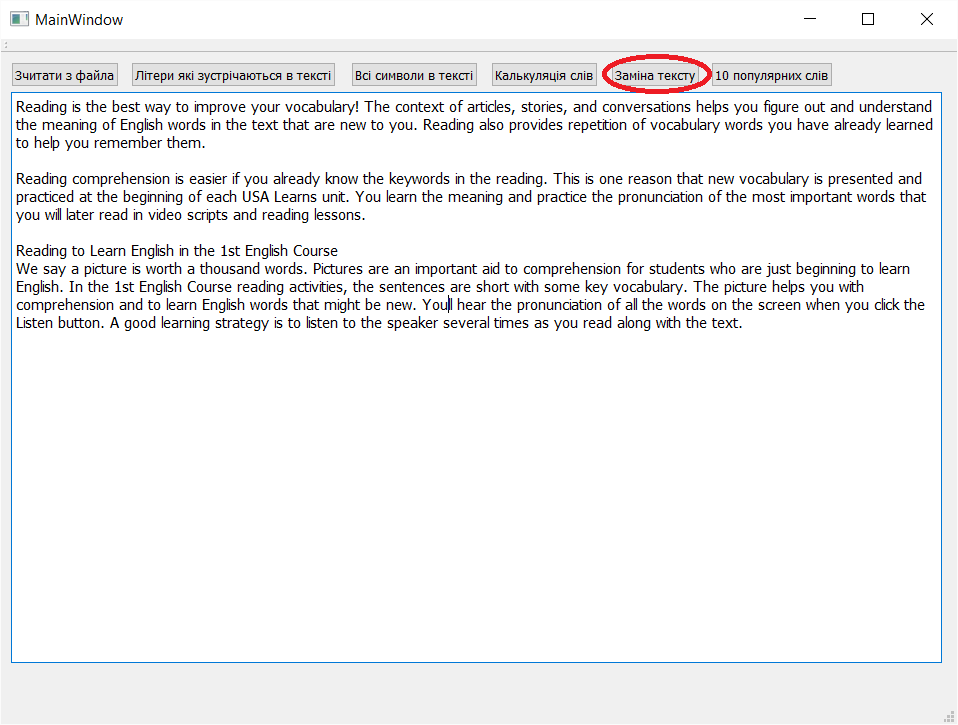
Літери які зустрічаються в тексті

****

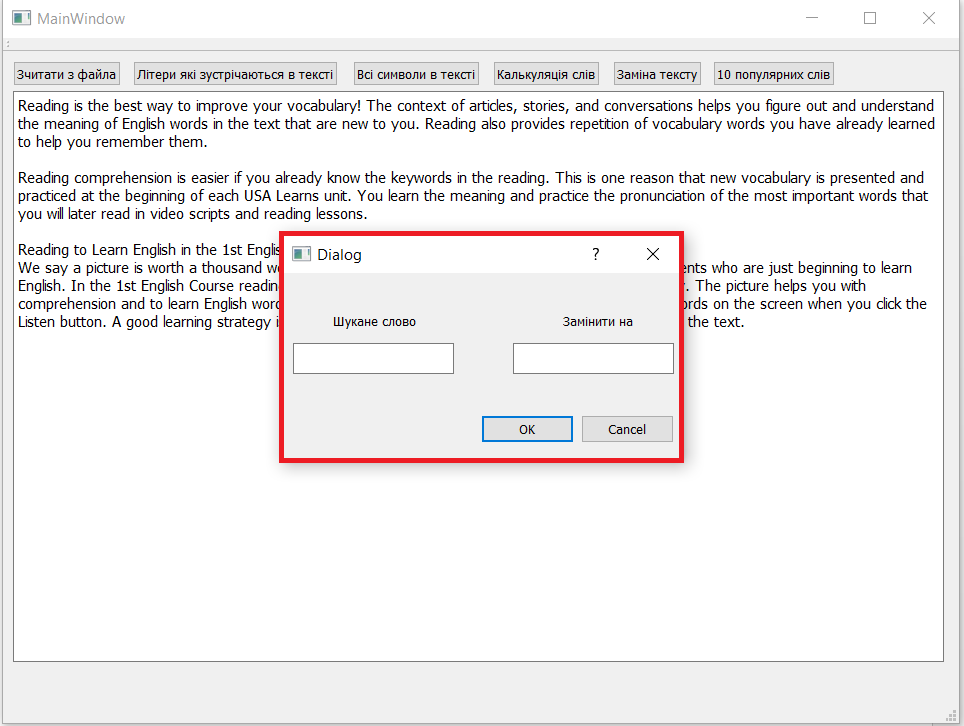
Всі символи які зустрічаються в тексті



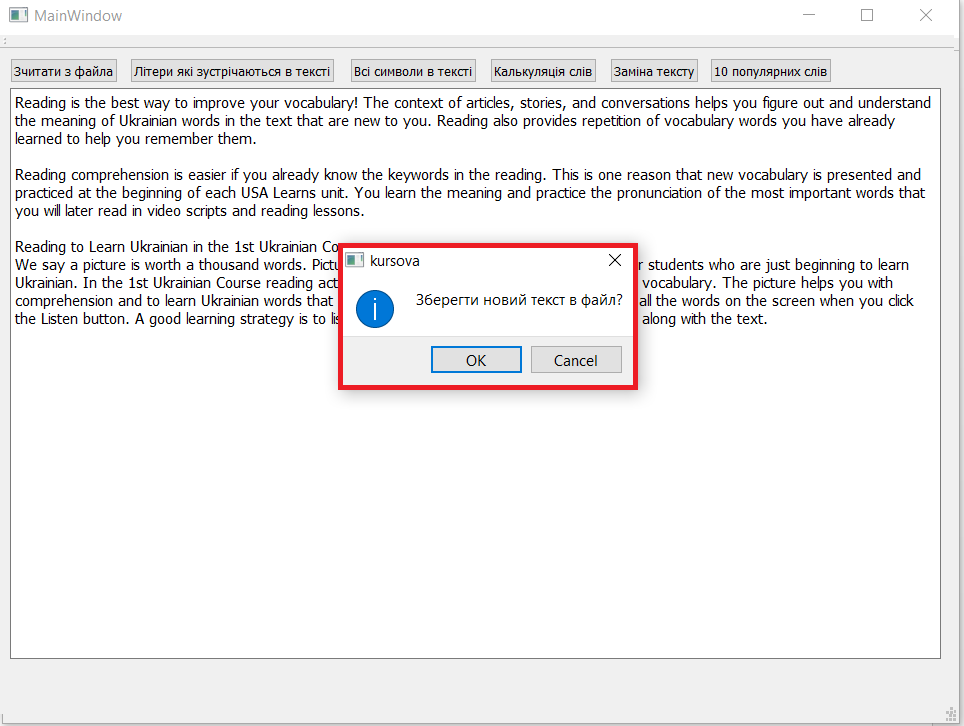
Калькуляція слів в тексті

****

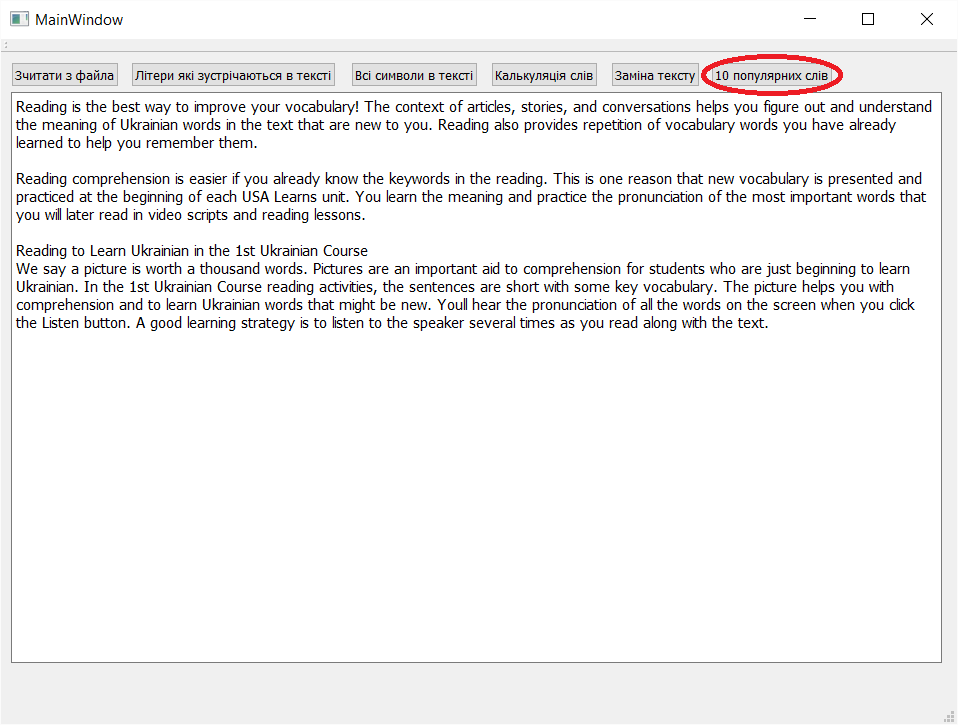
Заміна тексту



Діалогове вікно заміни тексту

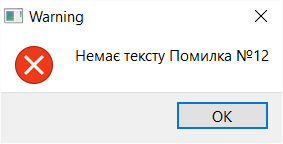


Збереження тексту в новий файл після заміни

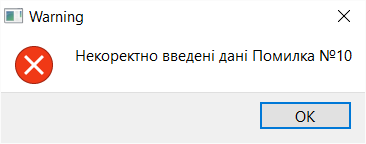


Виведення 10 популярних слів які зустрічаються в тексті

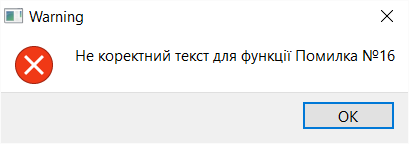
**Опис виняткових ситуацій**

****

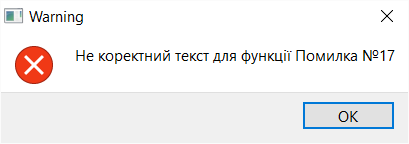
Не введено тексту для його опрацювання

****

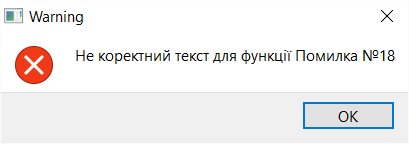
Некоректно введені дані при заміні слова

****

Не має тексту для підрахунку 10 популярних слів. Помилка №16

****

В тексті менше 10 слів. Помилка №17

****

Не має 10 різних слів в тексті для підрахунку популярних слів. Помилка №18

**Інструкція користувача**

Назва виконавчого файлу: CText.exe

Мінімальні системні вимоги для нормальної роботи:

процесор із частотою 800Мгц

128 мб оперативної пам'яті;

10 мб вільного місця на жорсткому диску;

32 Мб графічної пам'яті.

Приклад запису інформації "Hello world!"

-Зчитування тексту з файла: Зчитати з файла;

-Показати список літер які зустрічаються в тексті: Літери які зустрічаються в тексті;

-Показати список всіх символів які зустрічаються в тексті: Всі символи в тексті;

-Підрахувати кількість слів у тексті: Калькуляція слів;

-Заміна одного слова на інше: Заміна тексту;

-Показати 10 найпопулярніших слів в тексті: 10 популярних слів.

**Код програми**

**Text.h**

#ifndef TEXT\_H

#define TEXT\_H

#include <QString>

#include <QTextEdit>

#include <QFile>

#include <QDialog>

#include <QFileDialog>

#include <qdialog.h>

#include <QMainWindow>

#include "mainwindow.h"

class text {

private:

QString sMainString;

public:

text();

void **SetString**(QString sMainString);

QString **GetString**();

QString **CheckWords**(QString sString);

QString **CheckSigns**(QString sString);

QString **ChangeWord**(QTextEdit \*field, QString sFindWord, QString sReplaceWord, QString sCountOfReplace);

QString **Find10**(QString s10Words);

QString **CalculationWord**(QString QSwordCount);

};

#endif // TEXT\_H

**Dialog1.h**

#ifndef DIALOG1\_H

#define DIALOG1\_H

#include <QDialog>

#include <QString>

namespace Ui {

class Dialog1;

}

class Dialog1 : public QDialog

{

Q\_OBJECT

public:

explicit Dialog1(QWidget \*parent = 0);

~***Dialog1***();

QString **sFindWord**();

QString **sReplaceWord**();

private:

Ui::Dialog1 \*ui;

};

#endif // DIALOG1\_H

**Mainwindow.h**

#ifndef MAINWINDOW\_H

#define MAINWINDOW\_H

#include <mainwindow.h>

#include <QMainWindow>

#include "dialog1.h"

#include <QMessageBox>

#include <QTextStream>

#include <algorithm>

namespace Ui {

class MainWindow;

}

class MainWindow : public QMainWindow

{

Q\_OBJECT

public:

explicit MainWindow(QWidget \*parent = nullptr);

~***MainWindow***();

private slots:

void **on\_pushButtonLettersInTheText\_clicked**();

void **on\_pushButtonAllSymbol\_clicked**();

void **on\_pushButtonReplaceWord\_clicked**();

void **on\_pushButtonCalculationWords\_clicked**();

void **on\_pushButtonReadFromFile\_clicked**();

void **on\_pushButton10Words\_clicked**();

private:

Ui::MainWindow \*ui;

Dialog1 \*dialog1;

};

#endif // MAINWINDOW\_H

**Text.cpp**

#include "text.h"

#include <QMessageBox>

text::**text**() {

}

QString text::**GetString**() {

return sMainString;

}

void text::**SetString**(QString sMainString) {

this->sMainString = sMainString;

}

QString text::**CheckWords**(QString sString) {

char cArrayOfSymbol[] = {'A','B','C','D','E','F','G','H','I','J','K','L','M','N','O','P','Q','R','S','T','U','V','W','X','Y','Z'};

sString = "Кількість літер у тексті\n";

sMainString = sMainString.toUpper();

int iArrayOfSymbol[26] = {0};

for(unsigned i = 0; i < sizeof(cArrayOfSymbol); ++i) {

int iCountOfSymbol = sMainString.count(cArrayOfSymbol[i]);

iArrayOfSymbol[i] = iCountOfSymbol;

}

int iTmp = 0;

char cTmp = 0;

bool exit = false;

while(!exit) {

exit = true;

for(unsigned i = 0; i < sizeof(cArrayOfSymbol)-1; ++i) {

if(iArrayOfSymbol[i] < iArrayOfSymbol[i+1]) {

iTmp = iArrayOfSymbol[i];

cTmp = cArrayOfSymbol[i];

iArrayOfSymbol[i] = iArrayOfSymbol[i+1];

cArrayOfSymbol[i] = cArrayOfSymbol[i+1];

iArrayOfSymbol[i+1] = iTmp;

cArrayOfSymbol[i+1] = cTmp;

exit = false;

}

}

}

for(unsigned i = 0; i < sizeof(cArrayOfSymbol)-1; ++i) {

if(iArrayOfSymbol[i] != 0) {

sString = sString + cArrayOfSymbol[i] + " | " + QString::number(iArrayOfSymbol[i]) + "\n";

//field->setText(sString);

}

}

return sString;

}

QString text::**CheckSigns**(QString sString) {

char cArrayOfSymbol[] = {'A','B','C','D','E','F','G','H','I','J','K','L','M','N','O','P','Q','R','S','T','U','V','W','X','Y','Z',

'!','@','#','$','%','^','&','?','\*','(',')','-','\_','+','{','}','[',']',';',':','"','<','>',',','.','/'};

sString = "Всі символи які є у тексті\n";

sMainString = sMainString.toUpper();

int iArrayOfSymbol[52] = {0};

for(unsigned i = 0; i <= sizeof(cArrayOfSymbol) - 1; ++i) {

int iCountOfSymbol = sMainString.count(cArrayOfSymbol[i]);

iArrayOfSymbol[i] = iCountOfSymbol;

}

int iTmp = 0;

char cTmp = 0;

bool exit = false;

while(!exit) {

exit = true;

for(unsigned i = 0; i < sizeof(cArrayOfSymbol) - 1; ++i) {

if(iArrayOfSymbol[i] < iArrayOfSymbol[i+1]) {

iTmp = iArrayOfSymbol[i];

cTmp = cArrayOfSymbol[i];

iArrayOfSymbol[i] = iArrayOfSymbol[i+1];

cArrayOfSymbol[i] = cArrayOfSymbol[i+1];

iArrayOfSymbol[i+1] = iTmp;

cArrayOfSymbol[i+1] = cTmp;

exit = false;

}

}

}

for(unsigned i = 0; i < sizeof(cArrayOfSymbol) - 1; ++i) {

if(iArrayOfSymbol[i] != 0) {

sString = sString + cArrayOfSymbol[i] + " | " + QString::number(iArrayOfSymbol[i]) + "\n";

//field->setText(sString);

}

}

return sString;

}

QString text::**ChangeWord**(QTextEdit \*field, QString sFindWord, QString sReplaceWord, QString sCountOfReplace) {

int countZamin = 1;

unsigned countWords = sMainString.split(" ").count();

if(sMainString.contains(sFindWord, Qt::CaseInsensitive) == true) {

for(unsigned i = 0; i < countWords; ++i) {

int index1 = sMainString.indexOf(sFindWord);

sMainString.replace(index1, sFindWord.length(), sReplaceWord);

if(sMainString.contains(sFindWord, Qt::CaseInsensitive) == true) {

countZamin++;

}

}

}

field->setText(sMainString);

sCountOfReplace = "Кількість замін = " + QString::number(countZamin);

return sCountOfReplace;

}

QString text::**CalculationWord**(QString QSwordCount) {

sMainString = sMainString.simplified();

sMainString.remove(QRegExp("\\s+$"));

int iWordCount = sMainString.split(QRegExp("( {1,})+")).count();

QSwordCount = "Кількість слів у тексті = " + QString::number(iWordCount);

return QSwordCount;

}

QString text::**Find10**(QString s10Words) {

sMainString = sMainString.toLower();

QRegExp reg("[?!.;,:]");

QRegExp reg2("\n");

QRegExp reg3("\\s+");

s10Words = "";

sMainString = sMainString.replace(reg, "");

sMainString = sMainString.replace(reg2, " ");

sMainString = sMainString.replace(reg3, " ");

QStringList allWords = sMainString.split(' ');

if(allWords.count() < 10) {

return s10Words;

}

else {

QMap<QString, int> wordsMap;

QStringList::const\_iterator iter;

QMap<QString, int>::iterator it;

for(iter = allWords.constBegin(); iter != allWords.constEnd(); ++iter) {

if(!wordsMap.contains(\*iter)) {

wordsMap.insert((\*iter), 1);

}

else {

wordsMap[\*iter]++;

}

}

int max = 0;

for(it = wordsMap.begin(); it != wordsMap.end(); ++it) {

if(it.value() > max) {

max = it.value();

}

}

int count = 0;

int countOfPrint = 0;

while(count != 10) {

for(it = wordsMap.begin(); it != wordsMap.end(); ++it) {

if((it.value() == max) && (count != 10)) {

s10Words = s10Words + "\n" + it.key() + " = " + QString::number(it.value());

countOfPrint++;

count++;

}

}

max--;

if(max == 0) {

count = 10;

}

}

if(countOfPrint < 10) {

s10Words = " ";

return s10Words;

}

else {

return s10Words;

}

}

}

**Dialog1.cpp**

#include "dialog1.h"

#include "ui\_dialog1.h"

Dialog1::**Dialog1**(QWidget \*parent) :

QDialog(parent),

ui(new Ui::Dialog1)

{

ui->setupUi(this);

}

Dialog1::~***Dialog1***()

{

delete ui;

}

QString Dialog1::**sFindWord**()

{

return ui->findWord->toPlainText();

}

QString Dialog1::**sReplaceWord**()

{

return ui->replaceWord->toPlainText();

}

**Mainwindow.cpp**

#include "mainwindow.h"

#include "ui\_mainwindow.h"

#include "text.h"

MainWindow::**MainWindow**(QWidget \*parent) :

QMainWindow(parent),

ui(new Ui::MainWindow)

{

ui->setupUi(this);

}

MainWindow::~***MainWindow***()

{

delete ui;

}

void MainWindow::**on\_pushButtonLettersInTheText\_clicked**(){

text tForm;

QString sMainString = ui->mainTextEdit->toPlainText();

try {

if(sMainString.isEmpty()) {

throw 12;

}

tForm.SetString(sMainString);

QMessageBox::information(this, "Liter", tForm.CheckWords(" "));

}

catch(int i) {

QMessageBox::critical(this, "Warning", "Немає тексту Помилка №" + QString::number(i));

}

}

void MainWindow::**on\_pushButtonAllSymbol\_clicked**(){

text tForm;

QString sMainString = ui->mainTextEdit->toPlainText();;

tForm.SetString(sMainString);

try {

if(sMainString.isEmpty()) {

throw 14;

}

QMessageBox::information(this, "Liter", tForm.CheckSigns(" "));

}

catch(int i) {

QMessageBox::critical(this, "Warning", "Немає тексту Помилка №" + QString::number(i));

}

}

void MainWindow::**on\_pushButtonReplaceWord\_clicked**() {

dialog1 = new Dialog1(this);

if(dialog1->*exec*() == QDialog::Accepted) {

text tForm;

QString sMainString = ui->mainTextEdit->toPlainText();

QString findWordss = dialog1->sFindWord();

QString replaceWordss = dialog1->sReplaceWord();

tForm.SetString(sMainString);

try {

if((findWordss.isEmpty() && replaceWordss.isEmpty()) || (sMainString.contains(findWordss, Qt::CaseInsensitive) == false)) {

throw 10;

}

QMessageBox::information(this, "Count of replace", tForm.ChangeWord(ui->mainTextEdit, findWordss, replaceWordss, " "));

sMainString = ui->mainTextEdit->toPlainText();

tForm.SetString(sMainString);

QMessageBox msgBox;

msgBox.setText("Зберегти новий текст в файл?");

msgBox.setStandardButtons(QMessageBox::Ok | QMessageBox::Cancel);

msgBox.setIcon(QMessageBox::Information);

msgBox.setDefaultButton(QMessageBox::Ok);

int res = msgBox.*exec*();

if (res == QMessageBox::Ok) {

QString fileName = QFileDialog::getSaveFileName(this, tr("Save File"), QDir::currentPath(),

tr("Text Files (\*.txt)"));

QFile fileOut(fileName);

if(fileOut.*open*(QIODevice::WriteOnly | QIODevice::Text)) {

QTextStream writeStream(&fileOut);

writeStream << ui->mainTextEdit->toPlainText();

fileOut.*close*();

}

}

}

catch(int i) {

QMessageBox::critical(this, "Warning", "Некоректно введені дані Помилка №" + QString::number(i));

}

}

}

void MainWindow::**on\_pushButtonCalculationWords\_clicked**() {

text tForm;

QString sMainString = ui->mainTextEdit->toPlainText();

tForm.SetString(sMainString);

try {

if(sMainString.isEmpty()) {

throw 14;

}

QMessageBox::information(this, "Calculation", tForm.CalculationWord(" "));

}

catch(int i) {

QMessageBox::critical(this, "Warning", "Немає тексту Помилка №" + QString::number(i));

}

}

void MainWindow::**on\_pushButtonReadFromFile\_clicked**() {

QString fileName = QFileDialog::getOpenFileName(this, tr("Open File"), "",

tr("Text Files (\*.txt)"));

QFile file(fileName);

if((file.exists()) && (file.*open*(QIODevice::ReadOnly))) {

QString str = "";

while(!file.*atEnd*()) {

str=str+file.readLine();

}

ui->mainTextEdit->setText(str);

file.*close*();

}

}

void MainWindow::**on\_pushButton10Words\_clicked**() {

text tForm;

QString sMainString = ui->mainTextEdit->toPlainText();

tForm.SetString(sMainString);

try {

if(sMainString.isEmpty()) {

throw 16;

}

if(tForm.Find10(" ") == "") {

throw 17;

}

if(tForm.Find10(" ") == " ") {

throw 18;

}

QMessageBox::information(this, "10 Words", tForm.Find10(" "));

}

catch(int i) {

QMessageBox::critical(this, "Warning", "Не коректний текст для функції Помилка №" + QString::number(i));

}

}

**Висновки**

Виконуючи цю курсову роботу я закріпив принципи ООП і використання мови С++.