МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Запорізький національний технічний університет

Кафедра програмних засобів

ЗВІТ

з лабораторної роботи № 3

**ОБЧИСЛЕННЯ МЕТРИК РОЗМІРУ**

**ТА СКЛАДНОСТІ ПРОГРАМ**

з дисципліни

«Якість програмного забезпечення та тестування»

Виконав:

студент групи КНТ-136сп В.В. Гарковенко

Перевірила:

доцент, к.т.н. Т.І. Каплієнко

2018

**Мета роботи:** Вивчити основні метрики коду програмних систем та реалізувати їх із застосуванням мови С++ та Qt.

**Завдання до роботи:**

1. Ознайомитися з теоретичними відомостями щодо метрик програмних систем.
2. Реалізувати на мові C++ з використанням крос-платформного інструментарію Qt програму розрахунку метрик: кількість рядків коду; кількість пустих рядків; кількість рядків, що містять коментарі; кількість рядків, що містять вихідний код і коментарі; відсоток коментарів; словник операторів (кількість унікальних операторів програми, також символи-роздільники, імена процедур і знаки операцій); словник операндів (кількість унікальних операндів програми); загальна кількість операторів в програмі; загальна кількість операндів в програмі; довжина програми; обсяг програми; цикломатична складність; максимальний рівень вкладеності і довжина програми.
3. Виконати аналіз отриманих результатів.
4. Оформити звіт та відповісти на контрольні питання.

**Виконання:**

Вихідний код:

mainwindow.cpp

#include "mainwindow.h"

#include "ui\_mainwindow.h"

int n1, n2, m, N1, N2, N, Ropt;

MainWindow::MainWindow(QWidget \*parent) :

QMainWindow(parent),

ui(new Ui::MainWindow)

{

ui->setupUi(this);

}

MainWindow::~MainWindow()

{

delete ui;

}

void MainWindow::on\_analyseButton\_clicked()

{

// Get String from UI

QString str = ui->plainTextEdit->toPlainText();

// Source lines of code

int nSLOC = str.count("\n");

MainWindow::ui->labelSLOC->setText("Source Lines Of Code: " + QString::number(nSLOC+1));

// Empty Strings

int nEL = str.count("\n\n");

ui->labelEL->setText("Empty lines: " + QString::number(nEL));

// Comments

int nCom = str.count("//");

ui->labelComments->setText("Comments: "+QString::number(nCom)+" "+QString::number(((float)nCom)/(nSLOC+1)\*100)+"%");

//Only Code

ui->labelOnlyCode->setText("Only Code: "+ QString::number((nSLOC+1)-nCom-nEL));

//Operators

QStringList listOperators = str.split(" ");

QString typesOperators = "+ - = += -= ++ -- \* << >> < > != == || && & |";

QStringList splitList = str.split(QRegExp("(void, int, string, float, double, byte)[A-Za-z0-9]\*"),QString::SkipEmptyParts);

// List on Types - splitted

QStringList sTypes;

QStringList result;

listOperators.removeDuplicates();

sTypes = typesOperators.split(" ");

QStringList::Iterator iter = sTypes.begin();

for(int i = listOperators.count()-1; i>=0; --i)

{

const QString& item = listOperators[i];

iter=sTypes.begin();

while(iter!=sTypes.end())

{

if(item== \*iter)

{

result<<item;

}

++iter;

}

}

int r = 0;

for(int i = splitList.count()-1;i>=0;--i)

{

const QString& item = splitList[i];

r=r+item.count(")\n");

}

Ropt=r;

n1 = result.count();

ui->labelOperators->setText("Vocabulary Operators: " + QString::number(result.count()+r));

//Operands

QRegExp rx("([\\w\\']+)[\\s,.;]");

QStringList listOperands;

int pos = 0;

while ((pos = rx.indexIn(str, pos)) != -1) {

listOperands << rx.cap(1);

pos += rx.matchedLength();

}

listOperands.removeDuplicates ();

QString typesOperands = "int void string double float iterator List const array for while do goto";

QStringList ltypes;

ltypes = typesOperands.split(" ");

QStringList::Iterator itt = ltypes.begin();

for(int i = listOperands.count()-1; i >= 0; --i)

{

const QString& item = listOperands[i];

itt = ltypes.begin();

while (itt != ltypes.end()){

if(item==\*itt)

listOperands.removeAt(i);

++itt;}

if ((item.toFloat()!=0)) {listOperands.removeAt(i);}

}

for(int i = listOperands.count()-1; i >= 0; --i)

{

const QString& item = listOperands[i];

if (item=="0") {listOperands.removeAt(i);}

}

n2=listOperands.count();

ui->labelOperands->setText("Vocabulary Operands: "+QString::number(listOperands.count()));

// Total Operators

QStringList listTOpertator = str.split(" ");

QString typesTotOp = "+ - = += ++ -- \* << >> < > != == || && &";

QStringList newtypesTotOp;

QStringList ress;

newtypesTotOp = typesTotOp.split(" ");

QStringList::Iterator itTotOp = newtypesTotOp.begin();

for(int i = listTOpertator.count()-1; i >= 0; --i)

{

const QString& item = listTOpertator[i];

itTotOp = newtypesTotOp.begin();

while (itTotOp != newtypesTotOp.end()){

if(item==\*itTotOp){

ress << item;

}

++itTotOp;

}

}

N1 = ress.count();

ui->labelTOpertators->setText("Total Operators: "+ QString::number(ress.count()+Ropt));

// Total Operands

QRegExp regExp("([\\w\\']+)[\\s,.;]");

QStringList listTotOd;

int p = 0;

while ((p = regExp.indexIn(str, p)) != -1) {

listTotOd << regExp.cap(1);

p += regExp.matchedLength();

}

QString types = "int void string double float iterator List const array for while do goto";

QStringList ltypesTotOd;

ltypesTotOd = types.split(" ");

QStringList::Iterator itTotOd = ltypesTotOd.begin();

for(int i = listTotOd.count()-1; i >= 0; --i)

{

const QString& item = listTotOd[i];

itTotOd = ltypesTotOd.begin();

while (itTotOd != ltypesTotOd.end()){

if(item==\*itTotOd)

listTotOd.removeAt(i);

++itTotOd;}

if ((item.toFloat()!=0)) {listTotOd.removeAt(i);}

}

for(int i = listTotOd.count()-1; i >= 0; --i)

{

const QString& item = listTotOd[i];

if (item=="0") {listTotOd.removeAt(i);}

}

N2 = listTotOd.count();

ui->labelTotalOperands->setText("Total Operands: "+ QString::number(listTotOd.count()));

//Length program

m=n1+n2+Ropt;

ui->labelLengthVoc->setText("Length of vocabulary: "+ QString::number(m));

N=N1+N2+Ropt;

ui->labelLength->setText("Length: "+ QString::number(N));

//Cyclomatic

QStringList listCyclo = str.split(QRegExp("(for|while)"), QString::SkipEmptyParts);

int s=0;

for(int i = listCyclo.count()-1; i >= 0; --i)

{

const QString& item = listCyclo[i];

if ((item.count("{")>=item.count("}"))&&(item.count("}")!=0)) {s=s+1; /\*ui->plainTextEdit->insertPlainText(\*it+"+++");\*/}

}

ui->labelCyclomatic->setText("Cyclomatic: " + QString::number(s+str.count("if(")));

//CI

int n = str.count("if(");

ui->labelCI->setText("CI: "+ QString::number(n));

}

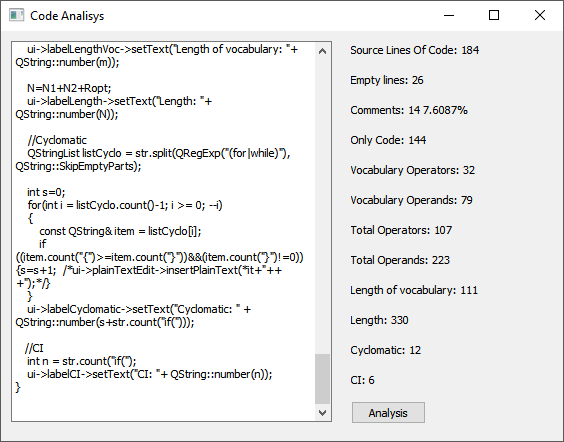


Рисунок 3.1 – Головне вікно програми

**Висновок:** При виконанні практичної роботи вивчив метрики коду програмних систем та реалізував їх із застосуванням мови С++ та Qt.