**Міністерство освіти і науки України**

**Національний університет «Запорізька політехніка»**

кафедра програмних засобів

РОЗРАХУНКОВО-ГРАФІЧНЕ ЗАВДАННЯ № 2

З дисципліни «Soft skills, Групова динаміка та комунікації»

На тему: «КОМАНДНА РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ»

Виконав:

студент групи КНТ-120 Потапов Д. С.

Прийняв:

доцент Льовкін В. М.

Запоріжжя

2021

# РОЗРАХУНКОВО-ГРАФІЧНЕ ЗАВДАННЯ № 2

1.1 Мета роботи

1.1.1 Навчитися розробляти програмне забезпечення, працюючи в команді.

1.1.2 Навчитися оформлювати програмну документацію.

1.2 Завдання до роботи

1.2.1 Ознайомитися з основними теоретичними відомостями за темою роботи, використовуючи дані методичні вказівки, а також рекомендовану літературу.

1.2.2 Сформувати команду з двох студентів та отримати у викладача індивідуальне завдання.

1.2.3 Завести акаунт на сервісі GitHub та створити команду для роботи в Slack.

1.2.4 Узгодити розподіл зобов’язань для виконання завдання, використовуючи Slack.

1.2.5 Створити репозиторій для роботи над проектом та дозволити доступ до проекту обом користувачам.

1.2.6 Завантажити у віддалений репозиторій існуючі файли проекту.

1.2.7 Створити дві гілки проекту: для презентації ревізій та для відлагодження проекту.

1.2.8 Налаштувати доступ до Git-репозиторію в інтегрованому середовищі розробки Eclipse.

1.2.9 Виконати реалізацію проекту в Eclipse та зазначити, яка частина коду ким була розроблена.

1.2.10 Визначити додаткову функцію та реалізувати її, працюючи над нею одночасно вдвох.

1.2.11 Завершити роботу над проектом.

1.2.12 Відповідно до діючих стандартів індивідуально оформити програмний документ, узгоджений з викладачем, на розроблене програмне забезпечення.

1.2.13 Оформити звіт з роботи.

1.2.14 Відповісти на контрольні запитання.

1.3 Основні теоретичні відомості

1.3.1 Програмні засоби командної розробки Для роботи з Git в інтегрованих середовищах розробки існують

спеціальні плагіни, наприклад, EGit для Eclipse. Для того щоб інсталювати даний плагін, необхідно в Eclipse об-рати пункт меню Help → Eclipse Marketplace та в рядку пошуку ввести EGit, після чого інсталювати відповідний плагін. Далі необхідно створити репозиторій на основі нового проекту. Після того, як проект створено, необхідно обрати з контекстного меню проекту Team → Share Project та виконати конфігурацію репозиторію (рис. 1.1).

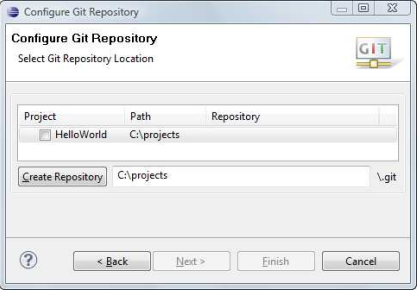


Рисунок 1.1 – Конфігурація Git репозиторію

У результаті проект належить репозиторію на основі гілки master (рис. 1.2).

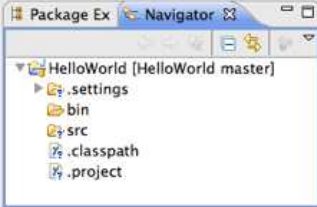


Рисунок 1.2 – Результат додавання проекту до репозиторію

Для відслідковування змін необхідно з контекстного меню обрати Team→ Add to Index. У результаті файли проекту додано до контролю версій (рис. 1.3).

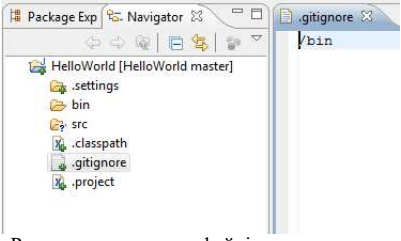


Рисунок 1.3 – Результат додавання файлів проекту до контролю версій

Далі можна виконати фіксацію, викликавши пункт меню Team→ Commit. Для налаштування SSH в Eclipse (рис. 1.4) при підключенні до GitHub необхідно обрати пункт меню Window → Preferences і далі General → Network Connections → SSH2. У даному вікні необхідно перевірити наявність ключів та коректність налаштування.

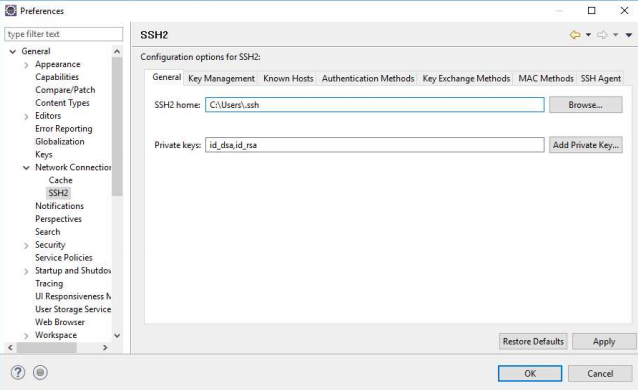


Рисунок 1.4 – Налаштування SSH ключів

Звідси з закладки Key Management за необхідності можна створити ключі. Далі необхідно завантажити публічний SSH ключ до налаштувань облікового запису GitHub. Після цього необхідно обрати з контекстного меню Team → Remote → Push... та скопіювати і вставити SSH URL репозиторію GitHub. На наступних кроках необхідно ввести ідентифікаційну фразу SSH ключа.

# 2 ВИКОНАННЯ РОБОТИ

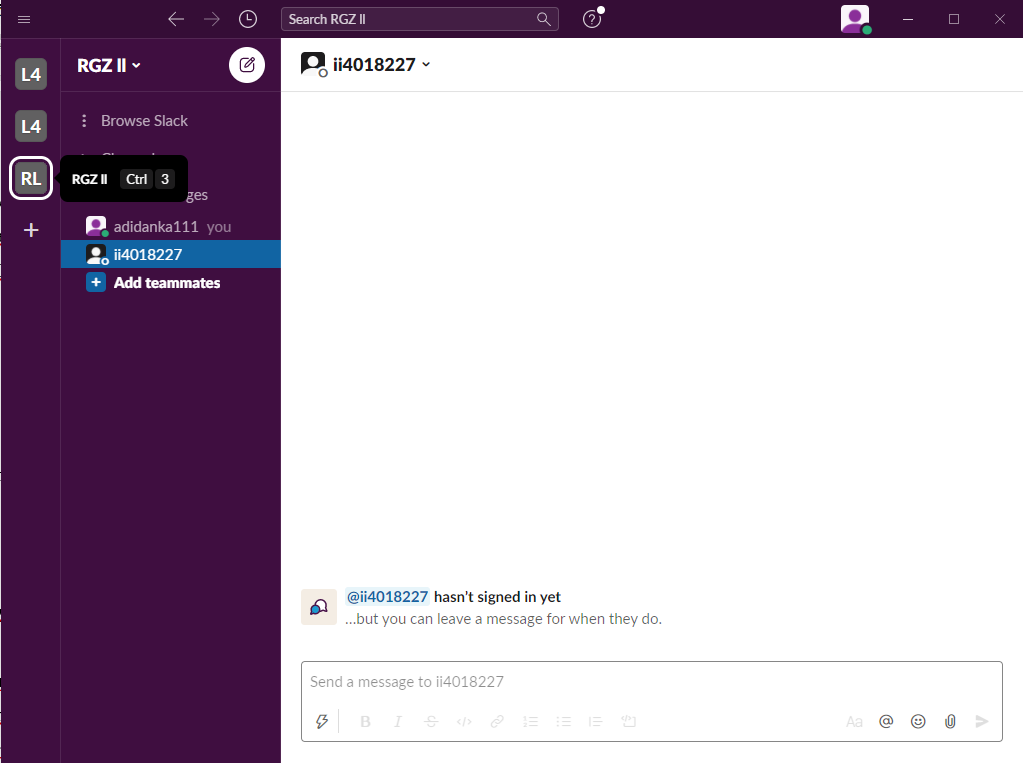


Рисунок 2.1 — Завести акаунт на сервісі GitHub та створити команду для роботи в Slack.

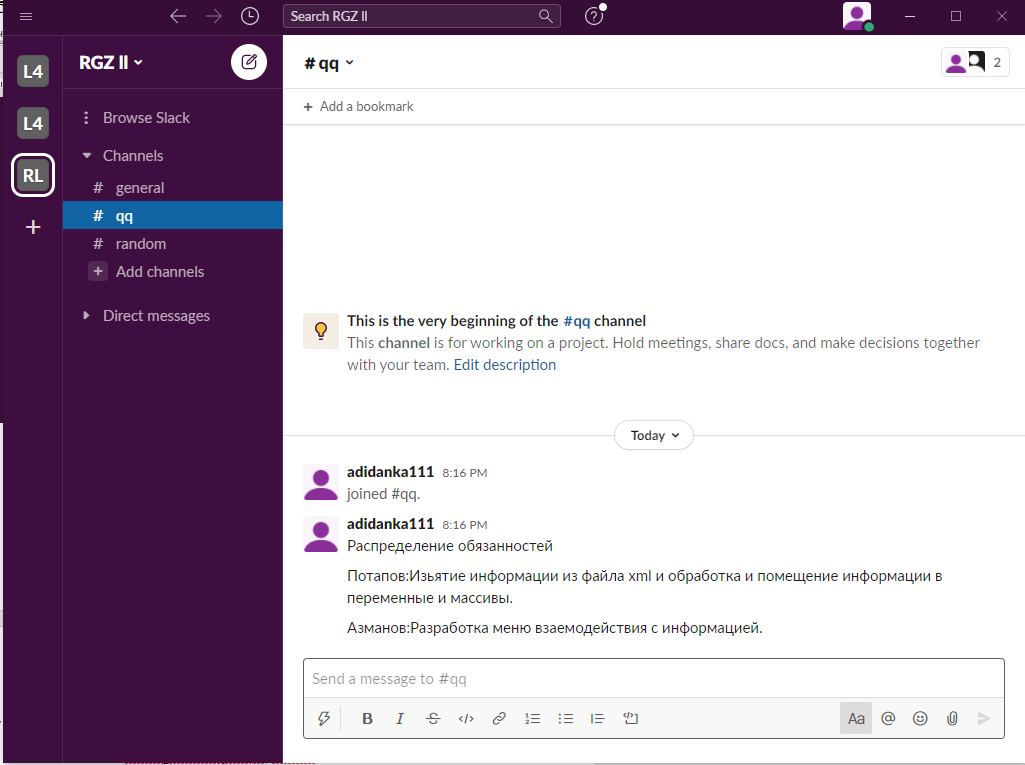


Рисунок 2.2 — Узгодити розподіл зобов’язань для виконання завдання, використовуючи Slack.

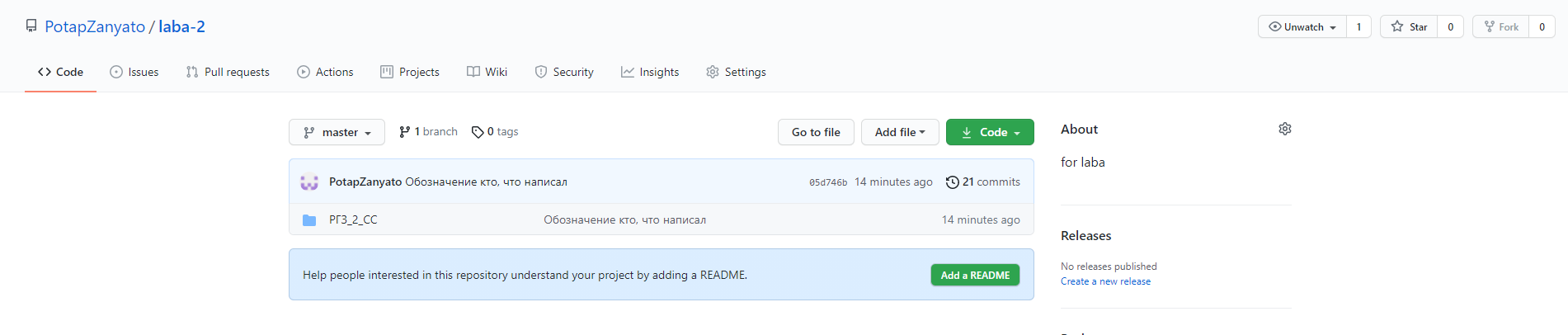


Рисунок 2.3 — Створити репозиторій для роботи над проектом та дозволити доступ до проекту обом користувачам.

( <https://github.com/PotapZanyato/laba-2.git>)

Посилання на проект.

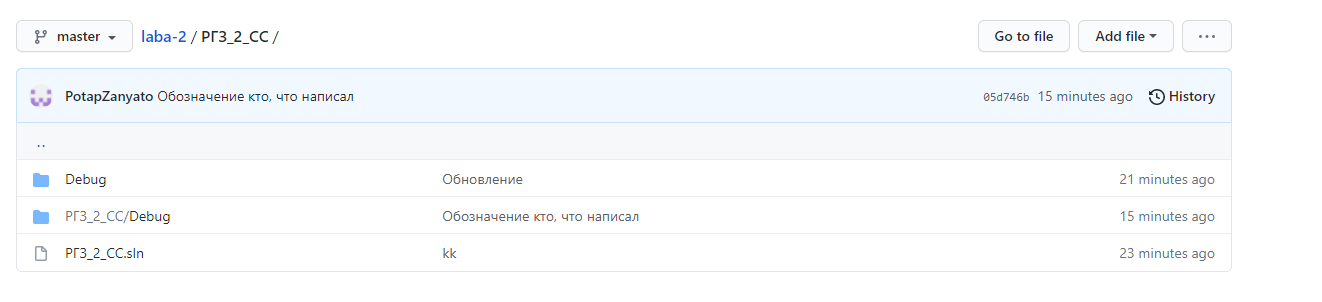


Рисунок 2.4 — Завантажити у віддалений репозиторій існуючі файли проекту.



Рисунок 2.5 — Створити дві гілки проекту: для презентації ревізій та для відлагодження проекту.

Виконати реалізацію проекту в Eclipse та зазначити, яка частина коду ким була розроблена. (Eclipse завантажує проект і видає помилки, за компілятор ми взяли Visual studio).

Відповідно до діючих стандартів індивідуально оформити програмний документ, узгоджений з викладачем, на розроблене програмне забезпечення (Керівництво програміста).

Код:

#include <iostream>

#include <string>

#include <fstream>

#include <iomanip>

using namespace std;

void GetName(string s, string name[5]);

void GetSurName(string s, string surname[5]);

void GetState(string s, string state[10]);

void GetBirth(string s, string birth[10]);

int main()

{

setlocale(LC\_ALL, "ru");

string file;

cout << "Введите название файла, с которого хотите считать информацию: ";

cin >> file;

if (file == "treeee.xml")

{

fstream f;

f.open(file);

string s[5];

int k = 0;

while (getline(f, s[k]))

{

k++;

}

cout << endl;

f.close();

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

string name[5];

GetName(s[0], name);

string surname[5];

GetSurName(s[0], surname);

int l = 0;

string state[10];

GetState(s[0], state);

fstream fin;

fin.open("tre.gtr", fstream::out);

string birth[10];

GetBirth(s[0], birth);

int v = 1;

for (int k = 0; k < 3; k++, l = l + 2, v++)

{

fin << v << "\t" << name[l] << "\t" << surname[k] << "\t" << state[k] << "\t" << birth[k] << endl;

}

fin.close();

string name1[5], surname1[5], state1[10], birth1[10];

GetName(s[1], name1);

GetSurName(s[1], surname1);

GetState(s[1], state1);

GetBirth(s[1], birth1);

fin.open("tre.gtr", fstream::app);

int v1 = 1;

int l1 = 0;

for (int k1 = 0; k1 < 3; k1++, l1 = l1 + 2, v1++)

{

fin << v1 << "\t" << name1[l1] << "\t" << surname1[k1] << "\t" << state1[k1] << "\t" << birth1[k1] << endl;

}

}

else

{

fstream f;

f.open(file);

string str[6][5];

string name[4] = { "<generation name="," surname="," state="," Birth=" };

string gen = "</generation></generation></generation>";

cout << endl;

for (int i = 0; i < 6; i++)

{

for (int j = 0; j < 5; j++)

{

f >> str[i][j];

cout << str[i][j] << "\t";

}

cout << endl;

}

cout << endl << endl;

string fin, fin\_1;

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

{

for (int j = 0; j < 4; j++)

{

fin += name[j] + str[i][j + 1];

}

}

fin += gen;

fstream f1;

f1.open("tree.xml");

f1 << fin << endl;

for (int i = 3; i < 6; i++)

{

for (int j = 0; j < 4; j++)

{

fin\_1 += name[j] + str[i][j + 1];

}

}

fin\_1 += gen;

f1 << fin\_1;

f1.close();

}

}

void GetName(string s, string name[5])

{

int i = 0;

int j;

int n = 0;

while (i < s.size())

{

if (s.find("name", i) != string::npos)

{

for (j = s.find("name", i) + 6; j < s.find("surname", i) - 2; j++)

{

name[n] += s[j];

}

i = j;

n++;

}

else

{

break;

}

}

}

void GetSurName(string s, string surname[5])

{

string m[5];

int i1 = 0;

int j1;

int n1 = 0;

while (i1 < s.size())

{

if (s.find("surname", i1) != string::npos)

{

for (j1 = s.find("surname", i1) + 9; j1 < s.find("state", i1 + 5) - 2; j1++)

{

m[n1] += s[j1];

}

n1++;

}

else

{

break;

}

i1 = j1;

}

for (int b = 0; b < 5; b++)

{

surname[b] = m[b];

}

}

void GetState(string s, string state[10])

{

int i = 0;

int j;

int n = 0;

while (i < s.size())

{

if (s.find("state", i) != string::npos)

{

for (j = s.find("state", i) + 7; j < s.find("Birth", i + 6) - 2; j++)

{

state[n] += s[j];

}

i = j;

n++;

}

else

{

break;

}

}

}

void GetBirth(string s, string birth[10])

{

int i = 0;

int j;

int n = 0;

string m[10];

while (i < s.size())

{

if (s.find("Birth", i) != string::npos)

{

for (j = s.find("Birth", i) + 7; j < s.find("><", i + 3) - 1; j++)

{

m[n] += s[j];

}

i = j;

n++;

}

else

{

break;

}

}

for (int b = 0; b < 10; b++)

{

birth[b] = m[b];

}

}

# 3 КЕРІВНИЦТВО ПРОГРАМІСТА

3.1 Призначення та умови застосування програми

Програма призначена для того, щоб проводити парсинг генеалогічного древа.Для роботи потрібен документ xml з даними.

Комп'ютер користувача, що включає в себе:

* процесор x86 з тактовою частотою, не менше 1 ГГц;
* оперативну пам'ять об'ємом, не менше 1 Гб;
* відеокарту пам'яттю не менше 512 Мб;
* монітор, миша, клавіатура;
* компiлятор.

3.2 Характеристики програми

Програма не виконує величезну кількість функцій тому працює досить швидко, при заупске програми виводиться меню де можна вибрати функції.В меню можна вибрати всю доступну інформацію, або знайти інформацію по поколінню, статi.

Час відновлення після відмови, викликаного збоєм електроживлення технічних засобів (іншими зовнішніми факторами), не фатальним збоєм (не крахом) операційної системи, не повинно перевищувати 5 хвилин за умови дотримання умов експлуатації технічних і програмних засобів.

Час відновлення після відмови, викликаного несправністю технічних засобів, фатальним збоєм (крахом) операційної системи, не повинно перевищувати часу, необхідного на усунення несправностей технічних засобів і перевстановлення програмних засобів.

Функцiя GetName — дозволяє додати в масив імена строки документа xml.

void GetName(string s, string name[5])

{

int i = 0; iтератор

int j; iтератор

int n = 0; iтератор

while (i < s.size())🡨Цикл до кінця рядка і потрібен для того щоб почати новий пошук з місця де закінчився старий

{

if (s.find("name", i) != string::npos)🡨 Перевірка на наявність потрібного слова

{

for (j = s.find("name", i) + 6; j < s.find("surname", i) - 2; j++)🡨Цикл для запису потрібного слова в осередок масиву

{

name[n] += s[j];

}

i = j;

n++;

}

else🡨 Зупинка циклу

{

break;

}

}

}

s.find("name", i) — знаходить перше входження слова name в строку i робить його точкою початка запису iменi в массив.

s.find("surname", i) — Бере слово surname як орiентир для того щоб цикл знав де закiнчувати записувати слово в массив.

Решта функцій, а саме:GetSurName, GetState, GetBirth. Працюють за таким же самим принципом, змінюються тільки ключовi слова, якi потрібно поставити як точку початку та кiнця запису слова.

Витягнути інформацію з файлу через цикл в масив рядків

fstream f;

f.open(file); 🡨Відкриття файлу

string s[5]; 🡨 Створення массиву

int k = 0;

while (getline(f, s[k]))🡨 Заповнюємо масив даних з файлу

{

k++;

}

cout << endl;

Частина про gtr документ.

fstream f;

f.open(file); 🡨Відкриття файлу

string str[6][5];🡨 Створення массиву

string name[4] = { "<generation name="," surname="," state="," Birth=" };🡨 Створення масива шаблонiв запису.

string gen = "</generation></generation></generation>";

for (int i = 0; i < 6; i++)

{

for (int j = 0; j < 5; j++)

{

f >> str[i][j];🡨 Заповнюємо масив даних з файлу

}

}

string fin, fin\_1;🡨 Створення об'єктів типу стринг

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

{

for (int j = 0; j < 4; j++)

{

fin += name[j] + str[i][j + 1];🡨 Поєднання даних з шаблонними тегами

}

}

fin += gen;🡨 Закриття тегів

fstream f1;

f1.open("tree.xml");

f1 << fin << endl;🡨 Пeрeміщення 1 змінної в файл

for (int i = 3; i < 6; i++)

{

for (int j = 0; j < 4; j++)

{

fin\_1 += name[j] + str[i][j + 1];🡨 Поєднання даних з шаблонними тегами

}

}

fin\_1 += gen;🡨 Закриття тегів

f1 << fin\_1;🡨 Переміщення 1 змінної в файл

f1.close();🡨Закриття файлу

3.3 Звернення до програми

Щоб запустити програму вам знадобиться відкрити компілятор, і xml або gtr документ з даними, документ потрібно кинути в папку з програмою, а так само в самій програмі вказати назву документа і його розширення.

3.4 Вхідні та вихідні дані

Вхідні дані програма бере з xml або gtr документа.

В залежності який документ було поставлено виробляються різні дії.

Документ xml: запис даних в масив відділення від тегів і запис відсортованих даних в файл.

Документ gtr: запис даних в масив додавання даних масиву до шаблонів тегів і запис в файл

3.5 Повідомлення

Для взаємодії з програмою потрібен правильно відредагований xml або gtr файл де правильно записана інформація, після запуску потрібно ввести назву документа і його розширення і програма автоматично зробить парсинг даних і запише в новий створений файл

# 4 Контрольнi запитання

4.1 Які основні документи оформлюються при розробленніпрограмного забезпечення?

Відповідь**:** технічне завдання; специфікація; опис програми; текст програми; керівництво програміста;керівництво системного програміста;керівництво оператора.

4.2 Які документи на Ваш погляд не є обов’язковими?

Відповідь**:** текст програми;

4.3 Які розділи містить технічне завдання?

Згідно з ГОСТом 19.201–78 технічне завдання повинно містити такі розділи: вступ; підстави для розробки; призначення розробки; вимоги до програми чи програмному виробу; вимоги до програмної документації; техніко-економічні показники; стадії та етапи розробки; порядок контролю та приймання.

4.4 Які розділи містить керівництво оператора?

Керівництво оператора оформлюється згідно з ГОСТом 19.505–78, відповідно до якого воно має містити наступні розділи: призначення програми; умови виконання програми; виконання програми; повідомлення оператору.

4.5 З яких розділів складається специфікація?

Згідно з ГОСТом 19.202-78 специфікація повинна містити розділи:документація; комплекси; компоненти.

# Висновки

На даному розрахунково графічному зображенні я навчився працювати в команді і розподіляти обов'язки, для більш ефективної роботи. Також я ознайомився з пунктами оформлення різної документації до програмного забезпечення і написав одне з них (в моєму випадку керівництво програміста).