**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 Азманов І. П.

Прийняв:

доцент Льовкін В. М.

Запоріжжя

2021

**2 РОЗРАХУНКОВО-ГРАФІЧНЕ ЗАВДАННЯ**

2.1 Мета роботи

2.1.1 Навчитися розробляти програмне забезпечення, працюючи в команді.

2.1.2 Навчитися оформлювати програмну документацію.

2.2 Основні теоретичні відомості

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.

У технічне завдання допускається включати додатки.

У залежності від особливостей програми чи програмного виробу допускається уточнювати зміст розділів, вводити нові розділи чи поєднувати окремі з них.

У розділі «Вступ» вказують найменування, коротку характеристику області застосування програми чи програмного виробу та об’єкта, у якому використовують програму чи програмний виріб.

У розділі «Підстави для розробки» повинно бути зазначено:

* документ (документи), на підставі якого ведеться розроблення;
* організація, що затвердила цей документ, і дата його затвердження;
* найменування і (або) умовне позначення теми розробки.

У розділі «Призначення розробки» повинно бути зазначене функціональне та експлуатаційне призначення програми чи програмного виробу.

Розділ «Вимоги до програми чи програмного виробу» повинен містити наступні підрозділи:

* вимоги до функціональних характеристик;
* вимоги до надійності;
* умови експлуатації;
* вимоги до складу та параметрів технічних засобів;
* вимоги до інформаційної та програмної сумісності;
* вимоги до маркування та упакування;
* вимоги до транспортування та збереження;
* спеціальні вимоги.

У підрозділі «Вимоги до функціональних характеристик» повинно бути зазначено вимоги до складу виконуваних функцій, організації початкових і вихідних даних, тимчасових характеристик тощо.

У підрозділі «Вимоги до надійності» повинно бути зазначено вимоги до забезпечення надійного функціонування (забезпечення стійкого функціонування, контролю початкової та вихідної інформації, часу відновлення після відмовлення тощо).

У підрозділі «Умови експлуатації» повинно бути зазначено умови експлуатації (температура навколишнього повітря, відносна вологість тощо для обраних типів носіїв даних), при яких повинні забезпечуватися задані характеристики, а також вид обслуговування, необхідна кількість і кваліфікація персоналу.

У підрозділі «Вимоги до складу і параметрів технічних засобів» указують необхідний склад технічних засобів із зазначенням їхніх основних технічних характеристик.

У підрозділі «Вимоги до інформаційної і програмної сумісності» повинно бути зазначено вимоги до інформаційних структур на вході і виході та методів розв’язання, вихідних кодів, мов програмування та програмних засобів, що використовуються програмою. За необхідності повинен забезпечуватися захист інформації та програм.

У підрозділі «Вимоги до маркування та упакування» у загальному випадку указують вимоги до маркування програмного виробу, варіанти та способи упакування.

У підрозділі «Вимоги до транспортування та збереження» повинні бути зазначені для програмного виробу умови транспортування, місця збереження, умови збереження, умови складування, терміни збереження в різних умовах.

У розділі «Вимоги до програмної документації» повинен бути зазначений попередній склад програмної документації і, за необхідності, спеціальні вимоги до неї.

У розділі «Техніко-економічні показники» повинні бути зазначені: орієнтована економічна ефективність, передбачувана річна потреба, економічні переваги розробки порівняно з кращими вітчизняними та закордонними аналогами.

У розділі «Стадії та етапи розробки» установлюють необхідні стадії розроблення, етапи та зміст робіт (перелік програмних документів, що повинні бути розроблені, погоджені та затверджені), а також, як правило, терміни розроблення, та визначають виконавців.

У розділі «Порядок контролю та приймання» повинні бути зазначені види іспитів і загальні вимоги до приймання роботи.

2.3 Завдання до роботи

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

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

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

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

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

2.3.6 Завантажити у віддалений репозиторій існуючі файли проекту.

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

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

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

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

2.3.11 Завершити роботу над проектом.

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

2.3.13 Оформити звіт з роботи.

2.3.14 Відповісти на контрольні запитання.

2.4 Виконання роботи

Заводжу акаунт на сервісі GitHub та створюю команду для роботи в Slack (рис. 2.4.1).

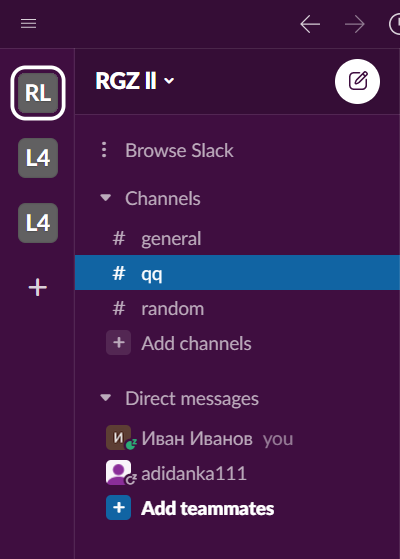


Рисунок 2.4.1 — Створюю команду в Slack

Узгоджуємо розподіл зобов’язань для виконання завдання, використовуючи Slack (рис. 2.4.2).

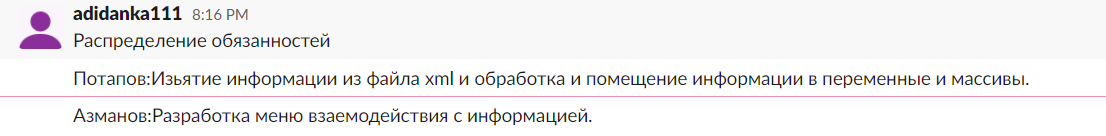


Рисунок 2.4.2 — Розподіл завдань за допомогою Slack

Створюю репозиторій для роботи над проектом та дозволяю доступ до проекту обом користувачам (рис. 2.4.3).

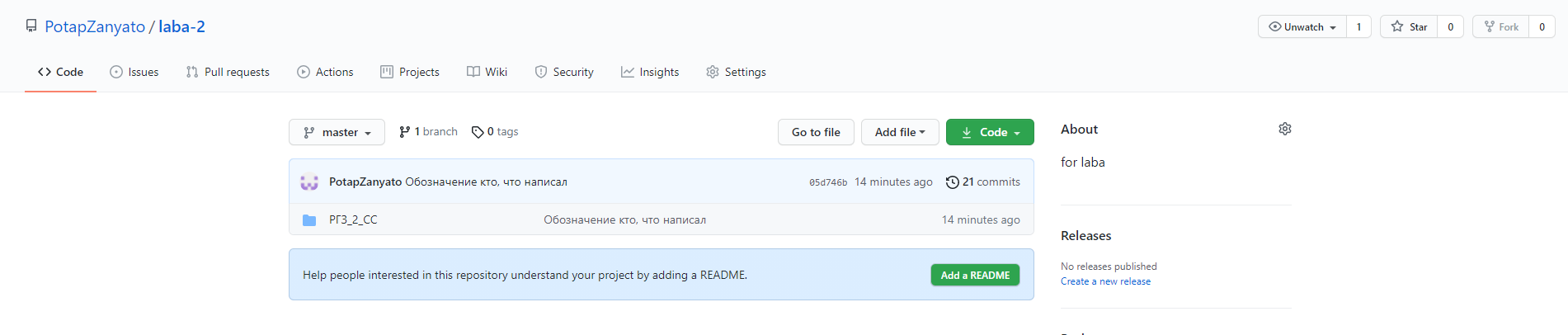


Рисунок 2.4.3 — Створення репозиторія та додання його до GitHub

Завантажую у віддалений репозиторій існуючі файли проекту (рис. 2.4.4).

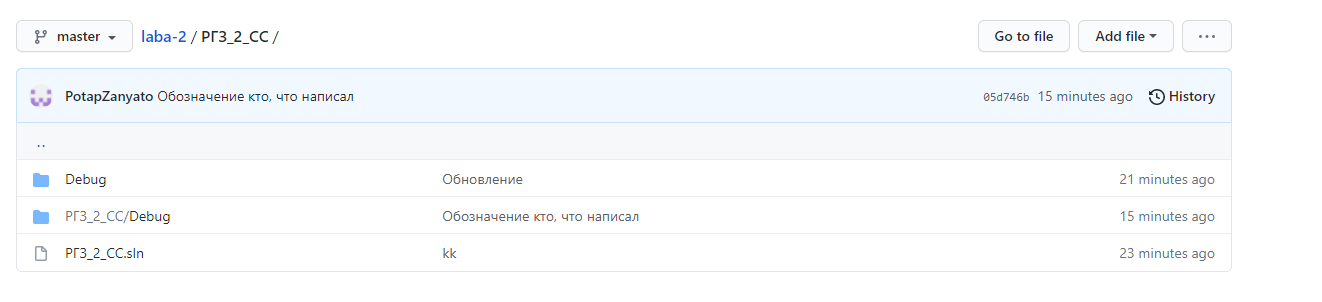


Рисунок 2.4.4 — Завантаження у репозиторій файли проекту

Створюю дві гілки проекту: для презентації ревізій та для відлагодження проекту (рис. 2.4.5).



Рисунок 2.4.5 — Створення двох гілок проекта.

Вихідний код програми:

#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];

}

}

Визначаємо додаткову функцію та реалізуємо її, працюючи над нею одночасно вдвох (рис. 2.4.6).

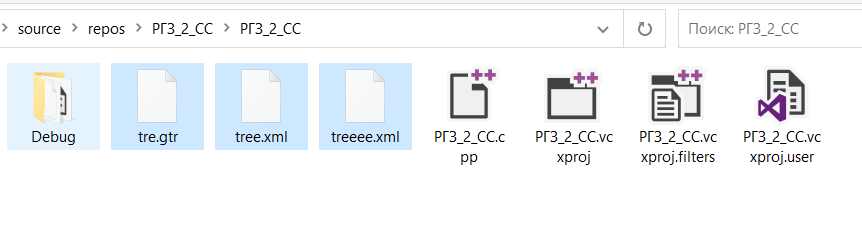


Рисунок 2.4.6 — Результат роботи функції, над якого ми працювали в дуо. Функція обробляє дані та залежно від введеного файлу створює новий файл

3 Технічне завдання

3.1 Найменування програми

Найменування програми — «Генеалогічне дерево».

3.1.1 Коротка характеристика області застосування

Програма "Генеалогічне дерево" призначена для отримання інформації про всі покоління сім'ї шляхом зчитування інформації з введеного документа, після чого йде запис переробленої інформації в документ з іншим розширенням.

3.2 Підстави для розробки

Підставою для розробки є договір 12 від 28.05.2021. Договір затверджений Директором ТОВ "Soft Skills" Льовкіним Валерієм Миколайовичем, іменованим надалі Замовником, а також Азмановим Іллею Павловичем і Потаповим Данилом Сергійовичем, іменовані надалі Виконавцями, 28.05.2021.

Згідно з договором, виконавці зобов'язані розробити та продемонструвати систему «Генеалогічне дерево» на обладнанні замовника не пізніше 20.06.2021, надати вихідні коди та документацію до розробленої програми не пізніше 20.06.2021.

Найменування теми розробки «Розробка інформаційно-довідкової програми Генеалогічне дерево».

Умовне позначення теми розробки (шифр теми) – «Ген-01».

3.3 Призначення розробки

Програма буде використовуватися будь-якими групами користувачів, які хочуть дізнатися інформацію про свій рід.

3.3.1 Функціональне призначення

Для користувачів, яким буде дозволений доступ, програма надає можливість ввести файл з певним розширенням (для взаємодії доступні файли з розширенням .xml, а також .gtr), після чого має відбуватися зчитування даних і в залежності від обраного розширення запис в інше розширення.

3.3.2 **Експлуатаційне призначення**

Програма повинна експлуатуватися в залі очікування інформації про свій рід. Запущена з правами звичайного користувача, вона може транслюватися на великі монітори для виведення інформації.

3.4 Вимоги до функціональних характеристик

Після запуску програми користувачеві відображається консоль, в якій потрібно ввести назву вихідного файлу і його розширення, з якого будуть зчитуватися дані.

Парсинг виконується (в xml документі) за допомогою 4 ключових функцій, які описані в самій програмі. Назви цих функцій наступні:

* GetName;
* GetSurName;
* GetState;
* GetBirth.

Кожна з цих функцій діє за аналогічним принципом: ведеться пошук ключових слів, після виявлення яких відбувається зчитування основної інформації щодо родичів з древа.

Інформація про родичів представлена в наступних даних:

* Покоління;
* Ім'я;
* Прізвище;
* Стать;
* Дата народження.

Після виявлення йде автоматичний запис отриманих даних у файл, без виведення на консоль отриманої і переробленої інформації.

3.4.1 Вимоги до організації вхідних і вихідних даних

Користувачеві надається можливість ввести файл з певним розширенням (дається два на вибір - xml та gtr), далі в залежності від обраного параметра генерується файл з протилежним розширенням, в якому записані перетворені дані.

Після запуску програми, валідація даних виконується на стороні користувача:

* назва-послідовність файлу не більше ніж з 200 будь-яких символів з обов'язковим зазначенням розширенням через крапку;
* файл, з якого повинна зчитуватися інформація, повинен зберігатися у кореневій теці проекту.

3.4.1.2 **Вимоги до тимчасових характеристик**

Вимог немає.

3.4.2 Вимоги до надійності

Імовірність безвідмовної роботи програми повинна становити не менше 99.99% за умови того, що повинен бути вказаний точний і правильний шлях до файлу, а також варто не забувати про правильність заповнення цього файлу, інакше відбудеться помилка зчитування та запису даних.

3.4.2.1 Вимоги до забезпечення надійного (стійкого) функціонування програми

Має вказуватися правильна назва файлу, а також повинна проводитися перевірка самим користувачем на розміщення файлу в кореневій теці проекту.

Надійне (стійке) функціонування програми має бути забезпечене виконанням замовником сукупності організаційно-технічних заходів, перелік яких наведено нижче:

* організацією безперебійного живлення технічних засобів;
* використанням ліцензійного програмного забезпечення;
* коректність введення і вказівки інформації;
* захист інформації;
* випробування програмних засобів на наявність комп'ютерних вірусів;
* місце для зберігання вихідних файлів.

3.4.2.2 Час відновлення після відмови

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.

3.4.3 Умови експлуатації

Програма запускається на комп'ютері, доступним користувачам.

3.4.3.1 Кліматичні умови експлуатації

Спеціальні умови не потрібні.

3.4.3.2 Вимоги до видів обслуговування

Програма не вимагає проведення будь-яких видів обслуговування.

3.4.3.3 Вимоги до чисельності та кваліфікації персоналу

При установці і налаштуванні системи не потрібно спеціальних навичок, так як використовується вже повністю налаштоване ПО.

До кваліфікації того, хто хоче дізнатися інформацію про свій рід, спеціальні вимоги не пред'являються.

3.4.4 Вимоги до складу та параметрів технічних засобів

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, монітор, миша, клавіатура;
* IDE.
* жорсткий диск на 100 Гб.

3.4.5 Вимоги до інформаційної та програмної сумісності

Спеціальних вимог не потрібно.

3.4.6 Вимога до маркування та упаковки

Програмний виріб передається по мережі Internet у вигляді архіву та завантажується з офіційного сайту виробника. Спеціальних вимог до маркування не пред'являється. Для перевірки автентичності програмного забезпечення рекомендується перевіряти контрольні суми завантажених файлів зі значеннями, зазначеними на офіційному сайті.

3.4.7 Вимоги до транспортування та зберігання

Спеціальних вимог не пред'являється.

3.4.8 Спеціальні вимоги

Програма повинна забезпечувати взаємодію з Користувачем за допомогою графічного інтерфейсу користувача, розробленого згідно з рекомендаціями компанії-виробника операційної системи.

3.5 Вимоги до програмної документації

Попередній склад програмної документації:

* технічне завдання (включає опис застосування);
* програма і методика випробувань;
* керівництво системного програміста;
* керівництво оператора;
* керівництво програміста;
* відомість експлуатаційних документів;
* формуляр.

3.6 Техніко-економічні показники

Програма "Генеалогічне дерево" придатна для міні-компаній, які не розглядають можливість роботи через Internet. Однак, в разі безкоштовного поширення програми, потреба в ній може бути досить високою - можливий більший інтерес про своїх предків. Економічний ефект при цьому може бути забезпечений за рахунок платної установки системи.

3.7 Стадії та етапи розробки

Розробка повинна бути проведена в три стадії:

* технічне завдання;
* технічний проект;
* впровадження.

На стадії "Технічне завдання" повинен бути виконаний етап розробки, узгодження та затвердження цього Технічного завдання.

На стадії "Технічний проект" повинні бути виконані перераховані нижче етапи робіт:

* розробка програми;
* розробка програмної документації;
* випробування програми.

На стадії «Впровадження» повинен бути виконаний етап розробки «Підготовка і передача програми».

На етапі підготовки та передачі програми повинна бути виконана робота з підготовки та передачі програми та програмної документації в експлуатацію на об'єктах замовника.

3.8 Порядок контролю та приймання

Приймально-здавальні випробування програми повинні проводитися згідно розробленої виконавцями і узгодженої замовником «Програми і методики випробувань».

Хід проведення приймально-здавальних випробувань замовник і виконавці документують в протоколі випробувань.

На підставі протоколу випробувань Виконавці спільно з замовником підписують акт приймання-здачі програми в експлуатацію.

ВИСНОВКИ

*3.5.1 Які основні документи оформлюються при розробленні програмного забезпечення?*

Технічне завдання, специфікація, опис програми, текст програми, керівництво програміста,керівництво системного програміста,керівництво оператора.

*3.5.2 Відповідно до якого стандарту оформляється документ «Технічне завдання»?*

Технічне завдання оформляють відповідно до ГОСТ 19.106-78 на аркушах формату 11 і 12 по ГОСТ 2.301-68, як правило, без заповнення полів листа. Номери аркушів (сторінок) проставляються у верхній частині аркуша над текстом.

*3.5.3 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.

У технічне завдання допускається включати додатки.

*3.5.4 Які розділи містить опис програми?*

Згідно з ГОСТом 19.402–78 опис програми повинен містити наступні розділи:

* загальні відомості;
* функціональне призначення;
* опис логічної структури;
* використані технічні засоби;
* виклик і завантаження;
* початкові дані, вихідні дані.

У залежності від особливостей програми допускається вводити додаткові розділи чи поєднувати окремі розділи.

*3.5.5 З яких розділів складається керівництво системного програміста?*

Керівництво програміста оформлюється згідно з ГОСТом 19.504–79, відповідно до якого воно має містити наступні розділи:

* призначення та умови застосування програми;
* характеристики програми;
* звернення до програми;
* вхідні та вихідні дані;
* повідомлення.

У залежності від особливостей документа допускається поєднувати окремі розділи або вводити нові.