**Міністерство освіти і науки України**

**Національний університет «Запорізька політехніка»**

Кафедра програмних засобів

Звіт

з самостійної роботи №2

З дисципліни “[Soft skills, групова динаміка та комунікації](https://moodle.zp.edu.ua/course/view.php?id=472)”

На тему: «Розрахунково-графічне завдання №2 командна розробка програмного забезпечення»

Виконали:

Студент групи КНТ – 113 сп Даніл Ілларіонов

Студентка групи КНТ – 133 Наталія Сазонова

Прийняв:

Викладач В.М. Льовкін

**2024**

**2 РОЗРАХУНКОВО-ГРАФІЧНЕ ЗАВДАННЯ No 2**

**КОМАНДНА РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ**

**2.1 Мета роботи:**

2.1.1 Навчитися розробляти програмне забезпечення, працюючи в команді.

2.1.2 Навчитися оформлювати програмну документацію.

**2.2 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. Створити програму, яка дозволяє скорочувати рядки, залишаючи їх унікальними. Під скороченням розуміється заміна частини символів (зокрема всередині рядка) трьома крапками (трикрапку рахувати як один символ). Користувач задає набір рядків і вибирає довжину для всіх рядків одразу. Кожен рядок повинен бути скорочений до заданої в якості довжини кількості символів. При цьому отримані в результаті скорочення рядки всі повинні бути унікальними. Якщо рядки не унікальні з самого початку, то додати в кінці імені цифру (щоб розрізняти рядки між собою).

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 в двох.

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.

**2.3 Короткі теоретичні відомості**

Основними документами при розробленні програмного забезпечення є:

– технічне завдання;

– специфікація;

– опис програми;

– текст програми;

– керівництво програміста;

– керівництво системного програміста;

– керівництво оператора.

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 [15].

За необхідності в технічне завдання можна включати потрібні додатки.

Згідно з ГОСТом 19.202-78 специфікація повинна містити розділи:

– документація;

– комплекси;

– компоненти.

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.

Згідно з ГОСТом 19.401–78 основна частина документу «Текст програми» повинна складатися з текстів одного чи декількох розділів, яким дано найменування, допускається вводити найменування також і для сукупності розділів, кожний з цих розділів реалізується одним з типів символічного запису, наприклад:

– символічний запис вихідною мовою;

– символічний запис проміжними мовами;

– символічне представлення машинних кодів.

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.

Згідно з ГОСТом 19.503–79 керівництво системного програміста повинно містити наступні розділи:

– загальні відомості про програму;

– структура програми;

– налаштування програми;

– перевірка програми;

– додаткові можливості;

– повідомлення системному програмісту.

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.

**2.4 Текст програми**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace WindowsFormsApp2

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void button1\_Click(object sender, EventArgs e)

{

int limit = Convert.ToInt32(textBox1.Text);

string text = richTextBox1.Text;

// Split the text into lines

string[] lines = text.Split(new[] { '\n', '\r' }, StringSplitOptions.RemoveEmptyEntries);

// Process each line

foreach (string line in lines)

{

// Split the line by spaces

string[] words = line.Split(new[] { ' ' }, StringSplitOptions.RemoveEmptyEntries);

}

string lastLine = "";

string temp = "";

if (limit < lines[0].Length)

{

if (limit > 2)

{

string first = Convert.ToString(lines[0][0]);

for (int i = 1; i < limit - 2; i++)

{

first += Convert.ToString(lines[0][i]);

}

first += "..." + lines[0][lines[0].Length - 1];

listBox1.Items.Add(first);

lastLine = first;

}

if (limit == 2)

{

string first = Convert.ToString(lines[0][0]);

first += "...";

listBox1.Items.Add(first);

lastLine = first;

}

}

else

{

MessageBox.Show("Limit must be lesser than the length of the word", "Warning", MessageBoxButtons.OK, MessageBoxIcon.Warning);

}

void Shorten(int imp)

{

if (limit < lines[imp].Length)

{

if (limit > 3)

{

int unqNum = 0;

string temp1 = Convert.ToString(lines[imp][0]);

for (int i = 1; i < lines[imp].Length - 1; i++)

{

if (lines[imp][i] != lines[imp - 1][i])

{

if (i > 1)

{

temp1 = temp1 + "...";

}

temp1 += Convert.ToString(lines[imp][i]);

unqNum = i;

break;

}

}

if (unqNum != 0)

{

if (unqNum + (limit - 4) < lines[imp].Length - 1)

{

if (unqNum > 1)

{

for (int i = unqNum; i < unqNum + (limit - 5); i++)

{

temp1 += lines[imp][i + 1];

}

}

else

{

for (int i = unqNum; i < unqNum + (limit - 4); i++)

{

temp1 += lines[imp][i + 1];

}

}

}

if (unqNum + (limit - 4) > lines[imp].Length - 1)

{

if (unqNum > 1)

{

for (int i = unqNum; i > unqNum - (limit - 5); i--)

{

temp1 += lines[imp][i - 1];

}

string original = temp1;

int startIndex = 4;

int endIndex = temp1.Length - 1;

string partToInvert = original.Substring(startIndex, endIndex - startIndex + 1);

char[] charArray = partToInvert.ToCharArray();

Array.Reverse(charArray);

string invertedPart = new string(charArray);

string result = original.Substring(0, startIndex) + invertedPart + original.Substring(endIndex + 1);

temp1 = result;

}

}

temp1 += "...";

temp1 += Convert.ToString(lines[imp][lines[imp].Length - 1]);

listBox1.Items.Add(temp1);

lastLine = temp1;

}

if (unqNum == 0)

{

listBox1.Items.Add(lastLine + " #1");

}

}

if (limit == 3)

{

int unqNum = 0;

string temp1 = Convert.ToString(lines[imp][0]);

for (int i = 1; i < lines[imp].Length; i++)

{

if (lines[imp][i] != lines[imp - 1][i])

{

unqNum = i;

temp1 += Convert.ToString(lines[imp][i]);

temp1 += "...";

listBox1.Items.Add(temp1);

lastLine = temp1;

break;

}

}

if (unqNum == 0)

{

listBox1.Items.Add(lastLine + " #1");

}

}

if (limit == 2)

{

int unqNum = 0;

string temp1;

for (int i = 1; i < lines[imp].Length; i++)

{

if (lines[imp][i] != lines[imp - 1][i])

{

temp1 = Convert.ToString(lines[imp][i]);

unqNum = i;

temp1 += "...";

listBox1.Items.Add(temp1);

lastLine = temp1;

break;

}

}

if (unqNum == 0)

{

listBox1.Items.Add(lastLine + " #1");

}

}

}

}

for (int i = 1; i < lines.Length; i++)

{

Shorten(i);

}

}

private void button2\_Click(object sender, EventArgs e)

{

richTextBox1.Text = "";

listBox1.Items.Clear();

}

}

}

**2.5 Опис розподілу відповідальності в команді.**

2.5.1 Написання коду - Даніл Ілларіонов.

2.5.2 Технічне завдання та керівництво програміста - Наталія Сазонова.

**2.6 Копії екранних форм з результатами виконання завдань.**

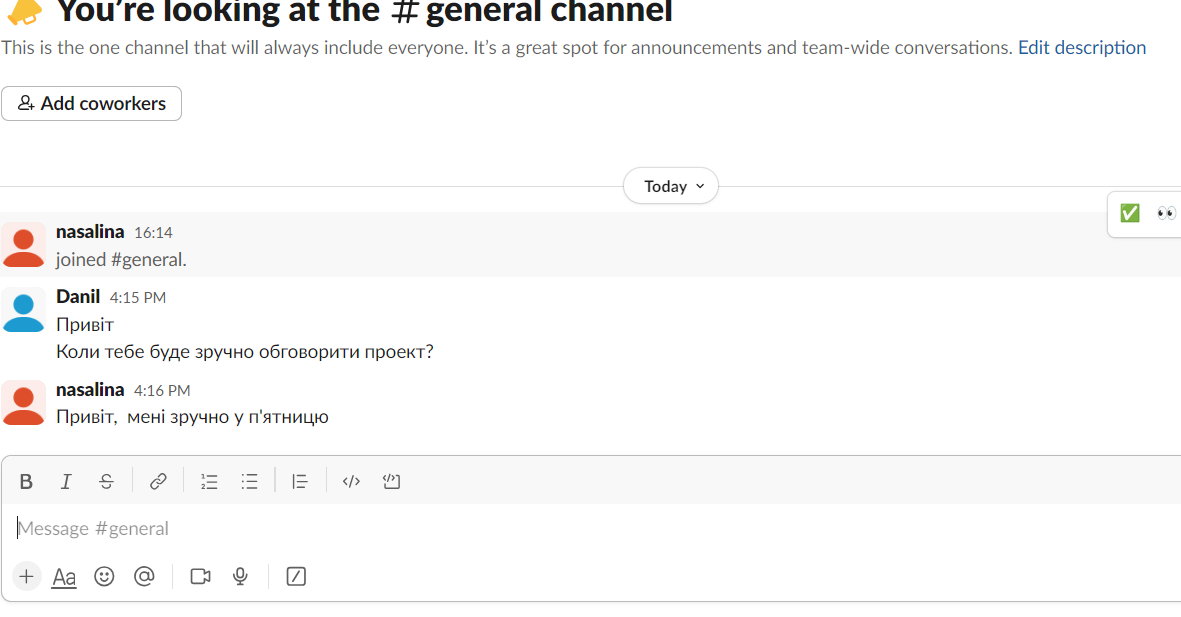


Рис. 2.1 Створення групи для обговорення завдання у Slack.

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..7 Оформлений програмний документ.**

2.7.1Технічне завдання.

2.7.1.1 Вступ.

Найменування програми: "Скорочення рядків".

Ця програма призначена для скорочення рядків тексту до заданої довжини. Вона розроблена для спрощення візуального сприйняття великих текстів, особливо тих, що містять довгі рядки.

Область застосування. Програма може бути корисною в різних контекстах, де потрібно скорочувати текст для кращого візуального сприйняття, наприклад:

а) Опрацювання великих текстів: Скорочення допомагає швидко ознайомитись з основним змістом тексту, не заглиблюючись у деталі;

б) Візуалізація даних: Скорочення може бути корисним при відображенні великих обсягів даних в таблицях, списках та інших форматах.

Об’єкт використання: Ця програма використовує Windows Forms для створення графічного інтерфейсу користувача.

2.7.1.2 Підстава для розробки.

а) Документ, на підставі якого ведеться розроблення – Методичні вказівки до виконання самостійної роботи з дисципліни “Soft skills, групова динаміка та комунікації”.

б) Організація, що затвердила цей документ, і дата його затвердження – В.М. Льовкін, А.О. Олійник. – Запоріжжя : НУ «Запорізька політехніка», 2023.

2.7.1.3 Призначення розробки.

Функціональне призначення:

– прийом вихідних даних в вигляді довільної кількості рядків тексту від користувача;

– прийом от користувача даних обмежуючих довжину опрацьованих рядків тексту;

– скорочення кожного рядка тексту до заданої довжини шляхом заміни частини символів трьома крапками ("..." - це один символ);

– забезпечення унікальності скорочених рядків.

Експлуатаційне призначення:

– спрощує процес скорочення рядків, забезпечуючи їх унікальність;

– зручний інструмент для роботи з текстовими даними, де необхідно скоротити довжину рядків, зберігаючи їх унікальність;

– може бути використаний для стиснення великих текстів, та візуалізації великих текстових масивів.

2.7.1.4 Вимоги до програми чи програмного виробу.

2.7.1.4.1 Вимоги до функціональних характеристик.

Вимоги до складу виконуваних функцій:

– Скорочення рядків. Програма повинна мати здатність скорочувати рядки тексту до заданої довжини;

– Обробка вхідних даних. Програма повинна приймати вхідний текст, який потрібно скоротити, та ціле число, що задає ліміт довжини рядка;

– Відображення результатів. Програма повинна відображати скорочені рядки в зручному для користувача форматі;

– Обробка помилок. Програма повинна обробляти випадки неправильних вхідних даних, наприклад, коли ліміт перевищує довжину першого рядка.

Вимоги до початкових даних:

– текст, який потрібно скоротити, надається користувачем через поле;

– максимальна довжина рядка задається користувачем через поле.

Вимоги до вихідних даних.

Опрацьований текст повинен відображатися з скороченими рядками заданої довжини в окремому полі.

Вимоги до тимчасових характеристик:

– Швидкість виконання. Програма повинна виконувати скорочення рядків з максимально можливою швидкістю;

– Час відгуку. Програма повинна реагувати на введення данних користувачем з мінімальним запізненням;

– Час обробки. Програма повинна ефективно обробляти текст.

2.7.1.4.2 Вимоги до надійності:

– Забезпечення стійкого функціонування. Програма повинна працювати стабільно і без збоїв при введенні допустимих даних;

– Захист від непередбачених ситуацій. Програма повинна мати механізми захисту від непередбачених ситуацій, таких як помилки введення, некоректний формат даних;

– Контроль початкової інформації. Програма повинна перевіряти коректність введених даних (перевірка, чи ліміт є цілим числом) та видавати відповідні повідомлення про помилки;

– Перевірка вихідної інформації. Програма повинна перевіряти отримані скорочені рядки на відповідність вимогам (унікальність, відповідність заданому ліміту довжини);

– Швидкість відновлення після відмовлення. Програма повинна забезпечувати швидке відновлення роботи після відмови, якщо така відбудеться.

2.7.1.4.3 Умови експлуатації.

Середовище:

– операційна система: Microsoft Windows 10 (64-бітна версія);

– температура: від 5°C до 40°C;

– відносна вологість: від 20% до 80%;

Зберігання даних:

– Тип даних. Текстові дані (рядки) зберігаються в оперативній пам'яті;

– Спосіб зберігання даних. Програма використовує зберігання в оперативній пам'яті для введеного користувачем тексту, тому немає специфічних вимог до зберігання.

Персонал:

– Необхідний персонал. Немає потреби в спеціалізованому персоналі. Програма розроблена для загальних користувачів;

– Рівень кваліфікації. Для використання програми достатньо базових знань роботи з комп'ютером.

– Платформа. Програма повинна працювати на комп'ютерах з операційною системою Windows (оскільки використовується `WindowsForms);

– Обмеження на ресурси. Програма не повинна вимагати великої кількості ресурсів (оперативної пам'яті, процесорного часу), щоб працювати ефективно;

– Мова інтерфейсу. Інтерфейс програми повинен бути українською мовою.

4) Вимоги до складу та параметрів технічних засобів:

Апаратні засоби:

– комп'ютер з процесором Intel або AMD;

– оперативна пам'ять не менше 1 ГБ;

– жорсткий диск з вільним місцем не менше 100 МБ;

– екран з роздільною здатністю не менше 1024x768 пікселів.

Програмне забезпечення:

– операційна система Microsoft Windows.

– .NET Framework.

5) Вимоги до інформаційної та програмної сумісності:

– формат даних. Програма повинна обробляти текст у стандартному форматі Unicode (UTF-8);

– сумісність з іншим програмним забезпеченням. Програма не повинна конфліктувати з іншим програмним забезпеченням, встановленим на комп'ютері;

– версії програмного забезпечення. Програма повинна працювати з відповідними версіями .NET Framework та операційної системи Windows.

**2.7.2 Керівництво програміста.**

2.7.2.1 Призначення та умови застосування програми

2.7.2.1.1 Призначення та функції.

Програма розроблена для скорочення рядків тексту, зберігаючи їх унікальність. Під скороченням розуміється заміна частини символів (зокрема всередині рядка) трьома крапками (…). Користувач задає набір рядків і вибирає довжину для всіх рядків одразу. Кожен рядок повинен бути скорочений до заданої в якості довжини кількості символів. При цьому отримані в результаті скорочення рядки всі повинні бути унікальними. Якщо рядки не унікальні з самого початку, то додати в кінці імені цифру (щоб розрізняти рядки між собою).

Функції програми:

* введення даних. Програма дозволяє користувачеві вводити текст у текстове поле;
* розділення на рядки: Програма розділяє введений текст на рядки за символами нового рядка;
* Обробка рядків:

; Програма перевіряє, чи довжина кожного рядка більша за заданий ліміт.

Якщо довжина рядка більша за ліміт, програма скорочує його до заданої довжини, замінюючи зайві символи трьома крапками (…).

Програма перевіряє унікальність отриманих рядків. Якщо рядок не унікальний, програма додає в кінці рядка номер (наприклад, "#1", "#2" тощо).

Виведення результатів: Програма відображає отримані скорочені рядки в елементі списку (ListBox).

Умови застосування:

Обсяг оперативної пам'яті: Програма не вимагає великого обсягу оперативної пам'яті, достатньо стандартної кількості, доступної на сучасних комп'ютерах.

Вимоги до складу та параметрів периферійних пристроїв: Програма не вимагає спеціальних периферійних пристроїв.

Вимоги до програмного забезпечення: Програма працює на платформі Microsoft Windows і потребує встановленої платформи .NET Framework.

Інші вимоги: Програма призначена для використання з текстовими файлами та не потребує доступу до мережі Інтернет.

Часові характеристики:

Швидкість виконання: Програма працює швидко, оскільки виконує прості операції над текстовими даними. Час виконання залежить від кількості рядків та їхньої довжини. Зазвичай, скорочення невеликого об'єму тексту відбувається майже миттєво.

Час відгуку: Програма реагує на введення даних користувача майже миттєво, за винятком випадків, коли виконується велика кількість рядків.

Режим роботи:

\* Однозадачний: Програма працює в однозадачному режимі. Вона обробляє один набір рядків одночасно.

\* Режим користувача: Програма призначена для використання в режимі безпосереднього взаємодії з користувачем. Користувач може вводити текст, вибирати параметри та отримувати результат.

Засоби контролю правильності виконання:

\* Перевірка довжини рядків: Програма перевіряє, чи довжина кожного рядка більша за заданий ліміт.

\* Перевірка унікальності рядків: Програма перевіряє, чи всі скорочені рядки є унікальними. Якщо рядок не унікальний, додається номер для його розрізнення.

\* Виведення попередження: Якщо ліміт заданий більшим за довжину першого рядка, програма виводить попередження користувачу.

Засоби самовідновлення:

\* Відсутні: Програма не має вбудованих механізмів самовідновлення.

Звертання до програми

Способи передачі управління:

\* Запуск виконуваного файлу: Програма запускається шляхом подвійного клацання на виконуваному файлі.

\* Виклик з командного рядка: Програму можна запустити з командного рядка, вказавши шлях до виконуваного файлу.

Передача параметрів даних:

\* Графічний інтерфейс: Програма має графічний інтерфейс, що дозволяє користувачу взаємодіяти з нею через елементи управління, такі як текстові поля, кнопки.

\* Введення з клавіатури: Користувач вводить текст у текстове поле `richTextBox1` та задає ліміт довжини у текстове поле `textBox1`.

\* Зовнішні файли: Програма не передбачає можливості зчитування даних з зовнішніх файлів.

«Початкові та вихідні дані»

Початкові дані:

\* Текст: Текстові дані, які потрібно скоротити, вводиться користувачем у текстовое поле `richTextBox1`.

\* Ліміт довжини: Користувач задає бажану довжину для скорочення рядків у текстовому полі `textBox1`.

Організація початкової інформації:

\* Текст: Текст зберігається у вигляді рядків, розділених символами нового рядка.

\* Ліміт довжини: Це ціле число, що вказує на бажану довжину рядків.

Вихідні дані:

\* Скорочені рядки: Програма виводить скорочені рядки у вигляді списку в елементі `listBox1`.

Організація вихідної інформації:

\* Скорочені рядки: Кожен рядок у списку `listBox1` являє собою скорочений рядок тексту.

Кодування:

\* Текст: Текст вводиться і виводиться у форматі Unicode.

\* Ліміт довжини: Це ціле число, тому не потребує спеціального кодування.

Повідомлення

Повідомлення для користувача:

\* "Limit must be lesser than the length of the word": Це повідомлення з'являється, якщо користувач вводить ліміт, який більший за довжину першого рядка тексту.

\* Змістовна частина: Повідомлення інформує користувача, що введений ліміт є завеликим для обробки.

\* Дія користувача: Користувачу необхідно внести новий ліміт, менший за довжину першого рядка.

### Повідомлення для програміста:

\* У програмі не виявлено повідомлень, призначених для програміста.

**2.8 Висновок**