Лабораторна робота №3

Утилітарні класи. Обробка масивів та рядків

Мета: Розробка власних утилітарних класів; набуття навичок вирішення прикладних задач з використанням масивів і рядків

1 ВИМОГИ

1. Розробити та продемонструвати консольну програму мовою *Java* в середовищі *Eclipse* для вирішення прикладної задачі за номером, що відповідає збільшеному на одиницю залишку від ділення на 15 зменшеного на одиницю номера студента в журналі групи.
2. При вирішенні прикладних задач використовувати латинку.
3. Продемонструвати використання об'єктів класу StringBuilder або StringBuffer.
4. Застосувати функціональну (процедурну) декомпозицію - розробити власні утилітарні класи (особливий випадок допоміжного класу, див. Helper Class) та для обробки даних використовувати відповідні статичні методи.
5. Забороняється використовувати засоби обробки регулярних виразів: класи пакету java.util.regex (Pattern, Matcher та ін.), а також відповідні методи класу String (matches, replace, replaceFirst, replaceAll, split).

1.1 Розробник

- П.І.Б: Абдуллін О. Р.
- Группа: КІТ-119а
- Варіант: 1

1.2 Загальне завдання

1) Ввести декілька рядків. Розбити рядки на три групи: починається з голосної; починається з приголосної; починається не з букви. Знайти найкоротший рядок в кожній групі. Вивести цей рядок та його довжину.

2 ОПИС ПРОГРАМИ

2.1 Було використано наступні засоби:

Random rand = new Random() - генерування випадкових чисел;

StringBuilder sb = new StringBuilder(str) – створення рядку типу StringBuilder;

sb.length() – визначення довжини рядка;

sb.indexOf(".", start) – визначення позиції заданого символа;

sb.charAt(i) – визначення символу, котрий стоїть на заданій позиції;

sb.substring(start,end+1) – виділення строки за заданими позиціями;

2.2 Ієрархія та структура класів

Було створено 2 класи

1) public class Main, який містить метод main, у якому задаються параметри для роботи з класом TypeOfString.
2) public class TypeOfString, класс для вирішення завдання.

2.3 Важливі фрагменти програми

Метод підрахунку кількості речень у тексті

```
public static int countsentences(String str)
{
	StringBuilder sb = new StringBuilder(str);
	int length = sb.length();
	int index = 0;
	int count = 0;
	for(; index < length; index++)
	{
		int temp;
		temp = findposition(sb, index);
		if(temp != -1)
		{
			count++;
			index = temp;
		}
	}
	return count;
}
```

Метод пошуку самого першого знаку, який свідчить про кінець речення, та вже не був використований

```
public static int findposition(StringBuilder sb, int start)
{
	int[] index = new int [3];
	int min = 0;
	int i = 1;
	index[0] = sb.indexOf(".", start);
	index[1] = sb.indexOf("!", start);
	index[2] = sb.indexOf("?", start);
	min = index[0];
	for(; 3 > i; i++)
	{
		if(min > index[i] && index[i] != -1 )
			min = index[i];
		if(min == -1)
			min = index[i];
	}
	return min;
}
```

Метод підрахунку кількості різних типів речень

```
public static void findtypes(String str, int sentences)
{
	StringBuilder sb = new StringBuilder(str);
	int vowel = 0;
	int conconent = 0;
	int other = 0;
	int index = 0;
```

```
	int start = 0;
	int type = -1;
	for(int i = 0; i < sentences; i++)
	{
		index = findposition(sb,index);
		type = typeofstring(senetence(sb, start, index));
		if(type == 0)
			vowel++;
		else if(type == 1)
			conconent++;
		else
			other++;
		start = index + 1;
		index++;
	}
	makemas(sb, vowel, conconent, other, sentences);
}
```

Метод визначення типу речення

```
public static int typeofstring(StringBuilder sb)
{
	int i = 0;
	char firstchar = sb.charAt(i);
	while(firstchar == ' ')
	{
		i++;
		firstchar = sb.charAt(i);
	}
	boolean isvowel = vowel(firstchar);
	if(isvowel == true)
		return 0;
	boolean isconconent = conconent(firstchar);
	if(isconconent == true)
		return 1;
	return 2;
}
```

Метод виділення речення з тексту

```
public static StringBuilder senetence (StringBuilder sb, int start, int end)
{
	String str;
	char firstchar = sb.charAt(start);
	while(firstchar == ' ')
	{
		start++;
		firstchar = sb.charAt(start);
	}
	str = sb.substring(start,end+1);
	StringBuilder strb = new StringBuilder(str);
	return strb;
}
```

Метод визначення найдовшого речення, та вивід у консоль

```
public static void longer(StringBuilder mas[], int count)
{
	int n = 0;
	int max = mas[0].length();
	for(int i = 1; count > i; i++)
	{
		if(mas[i].length() > max)
		{
			max = mas[i].length();
			n = i;
		}
```

```
        }
        System.out.print("The longest sentence is: ");
        System.out.println(mas[n]);
        System.out.print("It has length: ");
        System.out.println(max);
        System.out.println();
}
```

3 РЕЗУЛЬТУТ ВИКОНАННЯ РОБОТИ ПРОГРАММИ

Введений текст**:** It's a text. 1It's a sentece wich started neither vowel nor conconent! You are the best! Is that sentence started on vowel? Is that sentence started on conconent? This sentence stareted on conconent. 5*5=25?

```
Please, enter the text.
It's a text. 1It's a sentece wich started neither vowel nor conconent! You are the

This senteces are started on vowel:
It's a text.
You are the best!
Is that sentence started on vowel?
Is that sentence started on conconent?
The longest sentence is: Is that sentence started on conconent?
It has length: 38

This senteces are started on conconent:
This sentence stareted on conconent.
The longest sentence is: This sentence stareted on conconent.
It has length: 36

This senteces are started neither vowel nor conconent:
1It's a sentece wich started neither vowel nor conconent!
5*5=25?
The longest sentence is: 1It's a sentece wich started neither vowel nor conconent!
It has length: 57
```

Висновок

Під час виконання лабораторної роботи було набуто навички роботи з алгоритмами обробки масивів та рядків в середовищі Eclipse IDE.