Державний університет «Одеська Політехніка»
Інститут комп’ютерних систем
Кафедра інформаційних систем

Лабораторна робота №1
з дисципліни «Об’єктно-орієнтоване програмування»
Тема: «Установка програмного забезпечення. Базовий синтаксис Java»

Виконав:
студент групи АІ-201
Мартинюк Денис

Прийняв:
доц. Годовиченко М.А.

Одеса 2021

Цели лабораторной работы:

- установить программное обеспечение для программирования на языке Java;
- изучить базовый синтаксис языка Java и сравнить с базовым синтаксисом языка C;
- реализовать несколько тестовых задач на языке Java.

Ход работы:

1. Установка программного обеспечения.

Программное обеспечение было проверено. Также был создан и запущен проект на Java. Также был установлен плагин EduTools.

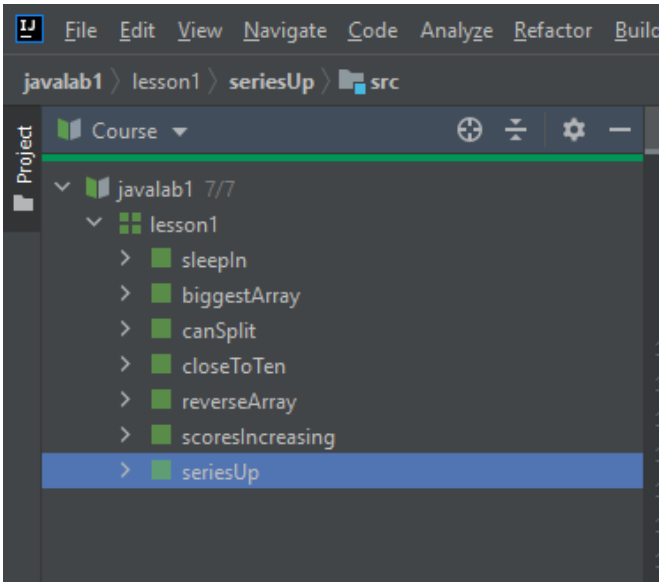
2. Изучение базового синтаксиса Java.

В рамках выполнения лабораторной работы был изучен базовый синтаксис языка Java: объявление и инициализация переменных, операторы ветвления, циклы и работа с массивами.

Было установлено, что базовый синтаксис языков Java и C очень похож. Отличия состоят в объявлении и базовой работе с массивами. Также, в Java есть возможность узнать длину массива.

3. Реализация тестовых задач на языке Java.

В рамках данной лабораторной работы было выполнено 7 тестовых заданий. Рассмотрим процесс выполнения данного задания и приведем исходный код каждого выполненного задания. На скриншоте, представленном ниже, видно, что все задания были выполнены.



1) Метод *sleepIn()*

Для решения данной задачи требуется составить логическое выражение, результат которого является выходом данной функции. В нашем случае метод должен вернуть true, если входной аргумент **weekday** равен **false** или **vacation** равен **true**.

```
public class Task
{

    public boolean sleepIn(boolean weekday  boolean vacation)
    {
        return !weekday || vacation;
    }

}
```

2) Метод *closeToTen()*

Для решения этой задачи мы находим абсолютное значение каждого из входных параметров и сравниваем их(0 является является доп.условием).

В зависимости какое число является наиближайшим после нахождения его абсолютного значения мы вернем соответствующий параметр.

```
public class Task
{

    public int closeToTen(int a, int b)
    {

        // Getting absolute values of numbers
        int x = Math.abs(10-a);
        int y = Math.abs(10-b);


        // Check for 3 conditions simultaneously
        return ( x < y ) ? a : (x > y) ? b : ( 0 );

    }

}
```

3) Метод *reverseArray()*

Для решения этой задачи был использован стандартный алгоритм реверса:

массив поделили на 2 части и в каждой итерации идет обмен значений между первой половиной и второй половиной массива.

Возвращаемое значение - перевернутый массив.

```
public class Task
{

    public int[] reverseArray(int[] array)
    {

        for (int i = 0; i < array.length/2; i++)
        {

            // Swap
            int temp = array[i];
            array[i] = array[array.length-i-1];
            array[array.length-i-1] = temp;

        }

        return array;

    }

}
```

4) Метод *biggestArray()*

Для решения этой задачи было использовано библиотечную функцию Java для поиска суммы всех елементов массивов. В качестве возвращаемого значения выступает массив с наибольшей суммой.

```
import java.util.Arrays;

public class Task
{

    public int[] biggestArray(int[] a  int[] b)
    {

        // If sum of *a* > sum of *b* - return *a*
        return (Arrays.stream(a).sum() > Arrays.stream(b).sum()) ? a : b;

    }


}
```

5) Метод *seriresUp()*

Данная задача была решена с использованием двух циклов (один вложенный), при этом количество итераций внутреннего цикла напрямую зависит от количества пройденных итераций внешним циклом, таким образом, дополнительно объявив независимый «итератор» мы можем заполнять массив нашими семплами пока внешний цикл не дойдет до своей границы.

```
public class Task
{

    public int[] seriesUp(int n)
    {

        // Allocate memory for the future array
        int[] Array = new int[n*(n+1)/2];

        // External counter to set values of array by sample
        int iteratorCount = 0;

        // Stops when we reach value of @param *a*
        for (int i = 1; i <= n; ++i)
        {

            // Keeps the sample and fills an array
            for (int j = 1; j <= i; ++j)
            {
                Array[iteratorCount++] = j;
            }

        }

        return Array;

    }

}
```

6) Метод canSplit()

Данная задача была решена простым перебором комбинаций.

В каждой итерации наружного цикла мы делим входной массив на две части в точке под индексом текущей итерации и в каждой из этих половин высчитываем сумму. Если суммы равны, значит мы разделили массив в верном месте и можно вернуть **true** , в противном случае метод вернет **false**.

```
public class Task
{

    public boolean canSplit(int[] a)
    {

        // Trying to split array in every point and check if valid condition
        for(int i = 0; i < a.length; i++)
        {

            // Every split we count sums of left and right parts
            int LeftPartSum = 0;
            for (int j = 0; j < i; j++)
            {
                LeftPartSum += a[j];
            }


            int RightPartSum = 0;
            for (int j = i; j < a.length; j++)
            {
                RightPartSum += a[j];
            }


            // If true - we split successfully
            if(LeftPartSum == RightPartSum){return true;}

        }

        return false;

    }

}
```

7) Метод scoresIncreasing()

Для решения этой задачи мы проходимся по всему массиву и сравниваем соседний элемент от текущей итерации. Если наше условие удовлетворяет, флаг помечается как **true**, но как только одна из итераций найдет неудовлетворение условию, флаг будет помечен как **false**.

Возращаемое значение функции и есть этот флаг.

```
public class Task
{

    public boolean scoresIncreasing(int[] a)
    {

        // Flag to check every element in loop
        boolean isNextGreater = false;

        for (int i = 0; i < a.length - 1 ; i++)
        {

            if(a[i+1] >= a[i]){ isNextGreater = true;}

            /*
            If in loop one condition hasn't succeeded, stop looping,
            flag shows last condition
            */

            else{ isNextGreater = false; break;}

        }

        return isNextGreater;

    }

}
```

Выводы к лабораторной работе:

В рамках выполнения данной лабораторной работы было установлено требуемое программное обеспечение, которое позволяет создавать и запускать проекты на языке Java. При установке программного обеспечения проблем и затруднений не возникло.

Изучен базовый синтаксис ЯП Java, так как Java основывается на языке C(си-подобный), то проблем с этим на текущем этапе выявлено не было.

Были выполнены тестовые задания. Трудностей выявлено не было.