Отчет по лабораторной работе №2

Выполнил:

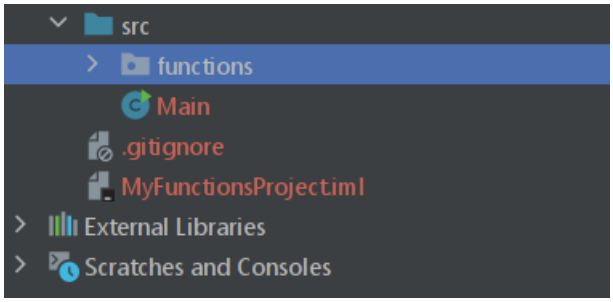
Дымченко Владислав

Группа: 6301-030301D

Задание 1

Нужно создать пакет functions, в котором далее будут создаваться классы программы. Работать будем в IntelliJ IDEA.

1. В левой части окна найдите панель project
2. Нажимаем и видим папку src.
3. Открываем папку src.
4. Выберите New и затем Package.
5. Вводим имя пакета: functions.



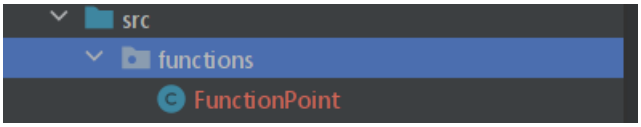
Задание 2

В пакете functions создадим класс FunctionPoint, объект которого должен описывать одну точку табулированной функции.

Состояние объектов должно содержать два аспекта: координату точки по оси абсцисс и координату точки по оси ординат. При написании класса следует учесть особенности инкапсуляции.

Создание класса FunctionPoint в пакете functions.

1. Нажимаем на пакет functions.
2. Выбираем New, потом Java Class.
3. Вводим имя класса: FunctionPoint.
4. Создался новый файл FunctionPoint.java.



Далее пишем код для класса FunctionPoint в редакторе, который открылся при создании файла FunctionPoint.java, использовав в классе следующие конструкторы:

• FunctionPoint(double x, double y) – создаёт объект точки с заданными координатами;

• FunctionPoint(FunctionPoint point) – создаёт объект точки с теми же координатами, что у указанной точки;

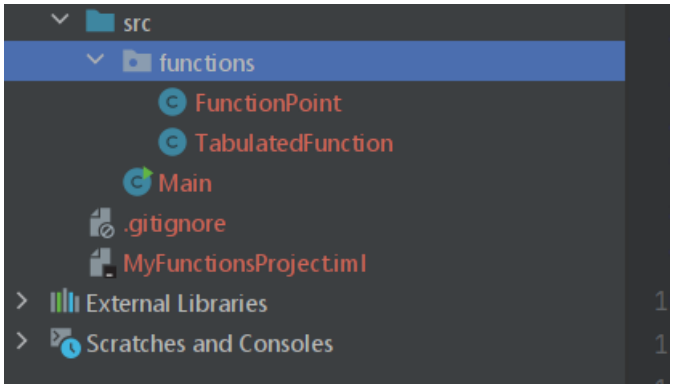
• FunctionPoint() – создаёт точку с координатами (0; 0).

Класс включает: поля x и y с модификатором доступа private,

Геттеры и сеттеры обеспеченивают инкапсуляцию

Задание 3

В пакете functions создаем класс TabulatedFunction, объект которого должен описывать табулированную функцию.



Используем конструкторы:

• TabulatedFunction(double leftX, double rightX, int pointsCount) – создаёт объект табулированной функции по заданным левой и правой границе области определения, а также количеству точек для табулирования.

• TabulatedFunction(double leftX, double rightX, double[] values) – аналогичен предыдущему конструктору, но вместо количества точек получает значения функции в виде массива.

Задание 4

В классе TabulatedFunction описываем методы, необходимые для работы с функцией.

• Метод double getLeftDomainBorder() – возвращает левую границу области определения табулированной функции.

• Метод double getRightDomainBorder() возвращает правую границу области определения табулированной функции.

• Метод double getFunctionValue(double x вычисляет значение функции с использованием линейной интерполяции

```
public double getLeftDomainBorder() { return points[0].getX(); }

public double getRightDomainBorder() { return points[pointsCount - 1].getX(); }


public double getFunctionValue(double x) {  5 usages
    if (x < getLeftDomainBorder() || x > getRightDomainBorder()) {
        return Double.NaN;
    }
```

Задание 5

В классе TabulatedFunction описываем методы, необходимые для работы с точками табулированной функции. Считаем, что нумерация точек начинается с ноля.

• Метод int getPointsCount() возвращает количество точек.

• Метод FunctionPoint getPoint(int index) возвращает копию точки, соответствующей переданному индексу.

• Метод void setPoint(int index, FunctionPoint point) заменяет указанную точку табулированной функции на переданную.

• Метод double getPointX(int index); double getPointY(int index) возвращают соответственно значение абсциссы и ординаты точки с указанным номером.

• Методы void setPointX(int index, double x); void setPointY(int index, double y) изменяют соответственно значение абсциссы и ординаты точки с указанным номером.

```
public int getPointsCount() { return pointsCount; }

public FunctionPoint getPoint(int index) {  2 usages
    if (index < 0 || index >= pointsCount) return null;
    return new FunctionPoint(points[index]);
}

public void setPoint(int index, FunctionPoint point) {  no usages
    if (index < 0 || index >= pointsCount) return;

    if (index > 0 && point.getX() <= points[index-1].getX()) { return; }
    if (index < (pointsCount-1) && point.getX() >= points[index+1].getX()) { return; }

    points[index] = new FunctionPoint(point);
}

public double getPointX(int index) {  6 usages
    if (index < 0 || index >= pointsCount) return Double.NaN;
    return points[index].getX();
}

public void setPointX(int index, double x) {  1 usage
    if (index < 0 || index >= pointsCount) return;

    if (index > 0 && x <= points[index-1].getX()) { return; }
    if (index < (pointsCount-1) && x >= points[index+1].getX()) { return; }

    points[index].setX(x);
}
```

```java
public double getPointY(int index) {  5 usages
    if (index < 0 || index >= pointsCount) return Double.NaN;
    return points[index].getY();
}

public void setPointY(int index, double y) {  3 usages
    if (index < 0 || index >= pointsCount) return;
    points[index].setY(y);
}
```

Задание 6

В классе TabulatedFunction описываем методы, изменяющие количество точек табулированной функции.

• Метод void deletePoint(int index) удаляет заданную точку табулированной функции.

• Метод void addPoint(FunctionPoint point) добавляет новую точку табулированной функции.

```
public void deletePoint(int index) {  1 usage
    if (pointsCount <= 2 || index < 0 || index >= pointsCount) return;

    System.arraycopy(points,  srcPos: index+1, points, index,  length: pointsCount-index-1);
    pointsCount--;
}

public void addPoint(FunctionPoint point) {  1 usage
    if (pointsCount >= points.length) {
        FunctionPoint[] newPoints = new FunctionPoint[points.length * 2];
        System.arraycopy(points,  srcPos: 0, newPoints,  destPos: 0, pointsCount);
        points = newPoints;
    }

    int pos = 0;
    while (pos < pointsCount && points[pos].getX() < point.getX()) {
        pos++;
    }

    System.arraycopy(points, pos, points,  destPos: pos + 1,  length: pointsCount - pos);

    points[pos] = new FunctionPoint(point);
    pointsCount++;
}
```

Задание 7

Создали класс Main для проверки функциональности:

1) Создана функция f(x) = x + 5 на интервале [0, 10]

2) Вычислили значения внутри и вне области определения

3) Проверили операции модификации, добавления и удаления точек

4) Проверили линейную интерполяцию

Вывод:

Создание функции f(x) = x + 5 на интервале [0, 10]:

Область определения: [0,0, 10,0]

Кол-во точек: 11

Точки табулированной функции:

Точка 0: (0,0; 5,0)

Точка 1: (1,0; 6,0)

Точка 2: (2,0; 7,0)

Точка 3: (3,0; 8,0)

Точка 4: (4,0; 9,0)

Точка 5: (5,0; 10,0)

Точка 6: (6,0; 11,0)

Точка 7: (7,0; 12,0)

Точка 8: (8,0; 13,0)

Точка 9: (9,0; 14,0)

Точка 10: (10,0; 15,0)

Вычисленные значения функции:

f(-5,0) = не определено (не входит в область определения)

f(-2,0) = не определено (не входит в область определения)

f(0,0) = 5,0 (должно быть: 5,0)

f(1,0) = 6,0 (должно быть: 6,0)

f(2,5) = 7,5 (должно быть: 7,5)

f(3,0) = 8,0 (должно быть: 8,0)

f(4,5) = 9,5 (должно быть: 9,5)

f(5,0) = 10,0 (должно быть: 10,0)

f(6,5) = 11,5 (должно быть: 11,5)

f(7,0) = 12,0 (должно быть: 12,0)

f(8,5) = 13,5 (должно быть: 13,5)

f(10,0) = 15,0 (должно быть: 15,0)

f(12,0) = не определено (не входит в область определения)

f(15,0) = не определено (не входит в область определения)

Проверка модификации точек:

Изменение ординаты точки с индексом 2 на правильное значение:

До: (2,0; 7,0) После: (2,0; 7,0)

Корректное изменение абсциссы с обновлением Y:

До: (4,0; 9,0) После: (4,2; 9,2)

Добавление точек:

Количество точек до добавления: 11

Количество точек после добавления: 12

Значение в добавленной точке: f(3,3) = 8,3

Точки после добавления:

(0,0; 5,0) (1,0; 6,0) (2,0; 7,0) (3,0; 8,0) (3,3; 8,3) (4,2; 9,2) (5,0; 10,0) (6,0; 11,0) (7,0; 12,0) (8,0; 13,0) (9,0; 14,0) (10,0; 15,0)

Удаление точек:

Количество точек до удаления: 12

Количество точек после удаления: 11

Обновление всех значений Y в соответствии с f(x) = x + 5:

Точка 0: (0,0; 5,0)

Точка 1: (1,0; 6,0)

Точка 2: (3,0; 8,0)

Точка 3: (3,3; 8,3)

Точка 4: (4,2; 9,2)

Точка 5: (5,0; 10,0)

Точка 6: (6,0; 11,0)

Точка 7: (7,0; 12,0)

Точка 8: (8,0; 13,0)

Точка 9: (9,0; 14,0)

Точка 10: (10,0; 15,0)

Интерполяция:

f(0,3) = 5,3 (должно быть: 5,3) правильно

f(1,1) = 6,1 (должно быть: 6,1) правильно

f(2,8) = 7,8 (должно быть: 7,8) правильно

f(3,9) = 8,9 (должно быть: 8,9) правильно

f(5,2) = 10,2 (должно быть: 10,2) правильно

f(6,7) = 11,7 (должно быть: 11,7) правильно

f(8,1) = 13,1 (должно быть: 13,1) правильно

f(9,4) = 14,4 (должно быть: 14,4) правильно

Границы:

Левая граница: f(0,0) = 5,0

Правая граница: f(10,0) = 15,0