# ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 4

## по курсу Объектно-ориентированное программирование

Выполнил: Акст Роман, 6203-010302D

Задание 1

Были реализованы следующие методы в классе FunctionPoint:

Метод toString() - возвращает текстовое описание точки в формате "(x; y)".

Метод equals() - сравнивает объекты точек, учитывая особенности сравнения чисел с плавающей точкой через Double.compare().

Метод hashCode() - реализация использует разбиение значений double на два int с помощью битовых операций, как требовалось в задании.

Метод clone() - реализовано простое клонирование через конструктор.

```java
@Override  new *
public String toString()
{
    return "(" + x +";" + y + ")";
}

@Override  new *
public boolean equals(Object obj)
{
    if (this == obj) return true;
    if (obj == null || getClass() != obj.getClass()) return false;

    FunctionPoint that = (FunctionPoint) obj;

    return Double.compare(that.x, x) == 0 &&
            Double.compare(that.y, y) == 0;
}

@Override  new *
public int hashCode()
{
    long xBits = Double.doubleToLongBits(x);
    long yBits = Double.doubleToLongBits(y);

    // Разбиваем каждый long на два int (старшие и младшие 32 бита)
    int x1 = (int)(xBits >> 32);
    int x2 = (int)xBits;
    int y1 = (int)(yBits >> 32);
    int y2 = (int)yBits;

    return x1 ^ x2 ^ y1 ^ y2;
}

@Override  new *
public Object clone()
{
    return new FunctionPoint(x, y);
}
```

Рисунок 1 – реализация методов для FunctionPoint

Задание 2

Были реализованы следующие методы в классе ArrayTabulatedFunction:

Метод toString() - возвращает строковое представление табулированной функции в формате "{(x1; y1), (x2; y2), ...}"(Рисунок 2)

Метод equals() - работает с любым объектом, реализующим интерфейс TabulatedFunction, с оптимизацией для ArrayTabulatedFunction(Рисунок 3)

Метод hashCode() – [эш-код рассчитывается на основе количества точек и хэш-кодов всех точек функции(Рисунок 4)

Метод clone() - неализовано глубокое клонирование с созданием нового массива точек(Рисунок 4)

```java
@Override  new *
public String toString()
{
    if (amountOfElements == 0)
    {
        return "{}";
    }

    StringBuilder sb = new StringBuilder();
    sb.append('{');

    for(int i = 0; i < amountOfElements; i++)
    {
        if (i > 0)
        {
            sb.append(", ");
        }

        sb.append(String.format("(%.2f; %.2f)", getPointX(i), getPointY(i)));
    }

    sb.append('}');
    return sb.toString();
}
```

Рисунок 2

```java
@Override  new *
public boolean equals(Object o)
{
    if (o == this) return true;
    if (o == null || !(o instanceof TabulatedFunction)) return false;

    TabulatedFunction that = (TabulatedFunction) o;
    if (that.getPointsCount() != amountOfElements) return false;

    if (o instanceof ArrayTabulatedFunction arrayThat)
    {
        for (int i = 0; i < amountOfElements; i++)
        {
            if (!this.points[i].equals(arrayThat.points[i]))
            {
                return false;
            }
        }

        return true;
    } else
    {
        // общее сравнение для любого TabulatedFunction
        for (int i = 0; i < amountOfElements; i++)
        {
            if (!this.points[i].equals(that.getPoint(i)))
            {
                return false;
            }
        }

        return true;
    }
}
```

Рисунок 3

```java
@Override  new *
public int hashCode()
{
    int result = amountOfElements; // включаем количество точек

    for (int i = 0; i < amountOfElements; i++)
    {
        result = 31 * result + points[i].hashCode();
    }

    return result;
}

@Override  new *
public ArrayTabulatedFunction clone()
{
    try
    {
        FunctionPoint[] clonedPoints = new FunctionPoint[amountOfElements];

        for (int i = 0; i < amountOfElements; i++)
        {
            clonedPoints[i] = (FunctionPoint) points[i].clone();
        }

        return new ArrayTabulatedFunction(clonedPoints);
    }
    catch(Exception e)
    {
        throw new RuntimeException("Ошибка копирования: " + e);
    }
}
```

Рисунок 4

Задание 3

Были реализованы следующие методы в классе LinkedListTabulatedFunction:

Метод toString() - аналогично ArrayTabulatedFunction, возвращает строковое представление функции.

Метод equals() - реализовано с оптимизацией для LinkedListTabulatedFunction и поддержкой любого TabulatedFunction(Рисунок 5)

Метод hashCode() - хэш-код включает количество точек и хэш-коды всех точек(аналогично ArrayTabulatedFunction).

Метод clone() - реализовано эффективное глубокое клонирование через пересборку списка(Рисунок 6)

```java
@Override  new *
public boolean equals(Object o)
{
    if (o == this) return true;
    if (o == null || !(o instanceof TabulatedFunction)) return false;

    TabulatedFunction that = (TabulatedFunction) o;
    if (that.getPointsCount() != count) return false;

    if (o instanceof LinkedListTabulatedFunction linkedThat)
    {
        FunctionNode thisCurrent = head.getNext();
        FunctionNode thatCurrent = linkedThat.head.getNext();

        while (thisCurrent != head && thatCurrent != linkedThat.head)
        {
            if (!thisCurrent.getPoint().equals(thatCurrent.getPoint()))
            {
                return false;
            }

            thisCurrent = thisCurrent.getNext();
            thatCurrent = thatCurrent.getNext();
        }

        return true;
    } else
    {
        // общее сравнение для любого TabulatedFunction
        FunctionNode currentNode = head.getNext();

        for (int i = 0; i < count; i++)
        {
            if (!currentNode.getPoint().equals(that.getPoint(i)))
            {
                return false;
            }

            currentNode = currentNode.getNext();
        }

        return true;
    }
}
```

Рисунок 5

```java
public Object clone()  new *
{
    try
    {
        if (count == 0)
        {
            return new LinkedListTabulatedFunction();
        }

        LinkedListTabulatedFunction cloned = new LinkedListTabulatedFunction();
        cloned.count = this.count;

        FunctionNode currentOriginal = this.head.getNext();
        FunctionNode firstCloned = new FunctionNode((FunctionPoint) currentOriginal.getPoint().clone());

        cloned.head.setNext(firstCloned);
        firstCloned.setPrev(cloned.head);

        FunctionNode lastCloned = firstCloned;
        currentOriginal = currentOriginal.getNext();

        while (currentOriginal != this.head)
        {
            FunctionNode newCloned = new FunctionNode((FunctionPoint) currentOriginal.getPoint().clone());
            lastCloned.setNext(newCloned);
            newCloned.setPrev(lastCloned);
            lastCloned = newCloned;
            currentOriginal = currentOriginal.getNext();
        }

        lastCloned.setNext(cloned.head);
        cloned.head.setPrev(lastCloned);

        return cloned;

    } catch (Exception e)
    {
        throw new RuntimeException("Ошибка клонирования", e);
    }
}
```

Рисунок 6

Задание 5

Интерфейс TabulatedFunction был расширен методом clone()(Рисунок 7)

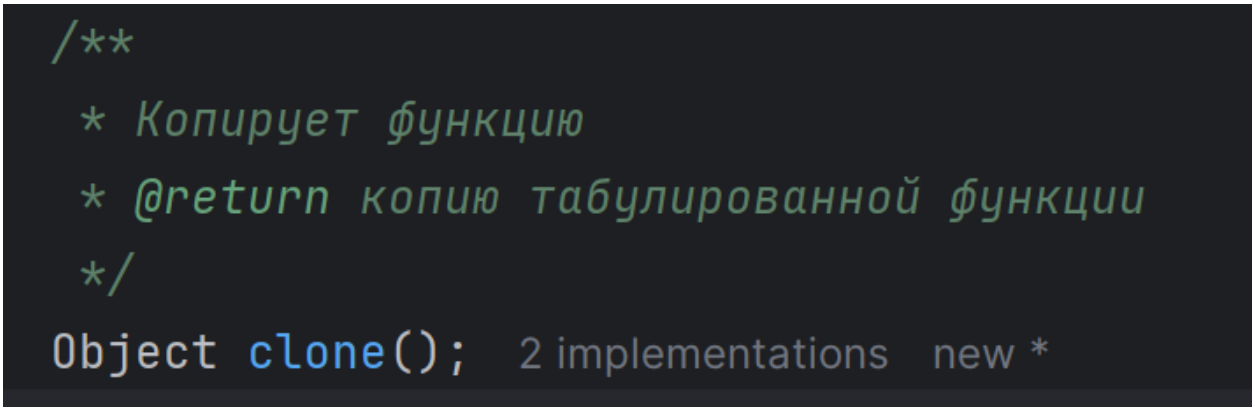


Рисунок 7

Задание 5

Для тестирования был создан класс Main, который проверяет все реализованные методы, на рисунке 8 показаны результаты выполнения работы с описанием проведенных тестов

```
1. Тест метода toString():
ArrayTabulatedFunction: {(1,00; 2,50), (2,00; 6,10), (3,00; 12,80), (4,00; 20,30), (5,00; 30,70)}
LinkedListTabulatedFunction: {(1,00; 2,50), (2,00; 6,10), (3,00; 12,80), (4,00; 20,30), (5,00; 30,70)}

2. Тест метода equals():
arrayFunc1 equals arrayFunc2: true
listFunc1 equals listFunc2: true
arrayFunc1 equals listFunc1: true
listFunc1 equals arrayFunc1: true

3. Тест метода hashCode():
Хэш arrayFunc1: -1640544643
Хэш arrayFunc2: -1640544643
Хэш listFunc1: -1640544643
Хэш listFunc2: -1640544643
Согласованность equals/hashCode для массивов: true
Согласованность equals/hashCode для списков: true

4. Тест изменения хэш-кода:
Исходный хэш списка: -1640544643
Измененный хэш списка: -2038416582
Хэш изменился после модификации: true

5. Тест метода clone():
arrayFunc1 == arrayClone: false
listFunc1 == listClone: false
arrayFunc1 equals arrayClone: true
listFunc1 equals listClone: true
Глубокое клонирование массива: true
Глубокое клонирование списка: true

=== ТЕСТИРОВАНИЕ ЗАВЕРШЕНО ===
```

Рисунок 8