Лабораторная работа №5

Заболотнов Николай Михайлович

6204-010302D

Task 1

В классе FunctionPoint переопределим методы toString(), equals(), hashCode(), clone() унаследованные от класс Object (рис 1). В методе equals() для проверки что объект является точкой воспользуемся оператором instanceof (рис 2). Для реализации метода hashCode() преобразуем значение double в его битовое представление в виде long, разобиваем на старшие и младшие биты с помощью операторов >> и &. Преведем все к типу int и получим хеш-код как побитового XOR (рис 3).

```
public String toString() {
    return "(" + x + "," + y + ")";
}

public Object clone() {
    return new FunctionPoint(this);
}
```

Рис 1

```
public boolean equals(Object o) {
    return o instanceof FunctionPoint && Math.abs(((FunctionPoint) o).getX() - x) < E && Math.abs(((FunctionPoint) o).getY() - y) < E;
}
```

Рис 2

```
public int hashCode() {
    long longX = Double.doubleToLongBits(x);
    long longY = Double.doubleToLongBits(y);
    return (int) (longX >> 32) ^ (int) (longX & 0xFFFFFFFFL) ^ (int) (longY >> 32) ^ (int) (longY & 0xFFFFFFFFL);
}
```

Рис 3

Задание выполнено.

Task 2

В классе ArrayTabulatedFunction так же переопределим эти методы (рис 4). Для реализации будет использовать ранее написанные методы в задании 1. В методе equals() если функция представлена в виде ArrayTabulatedFunction будет к точкам обращаться по индексу вместо метода getPoint() (рис 5). Хеш-код будет получаться побитовым XOR количества точек и хеш-кода каждой точки (рис 6).

```java
public String toString() {
    String s = "";
    for (int i = 0; i < PointsCount - 1; ++i)
        s += points[i].toString() + ", ";
    s += points[PointsCount - 1].toString();
    return "{" + s + "}";
}

public Object clone() {
    FunctionPoint[] points = new FunctionPoint[PointsCount];
    for (int i = 0; i < PointsCount; ++i)
        points[i] = (FunctionPoint) this.points[i].clone();
    return new ArrayTabulatedFunction(points);
}
```

Рис 4

```java
public boolean equals(Object o) {
    if (o instanceof ArrayTabulatedFunction && PointsCount == ((ArrayTabulatedFunction) o).PointsCount) {
        for (int i = 0; i < PointsCount; ++i)
            if (!points[i].equals(((ArrayTabulatedFunction) o).points[i]))
                return false;
        return true;
    }
    else if (o instanceof TabulatedFunction && PointsCount == ((TabulatedFunction) o).getPointsCount()) {
        for (int i = 0; i < PointsCount; ++i)
            if (!points[i].equals(((TabulatedFunction) o).getPoint(i)))
                return false;
        return true;
    }
    return false;
}
```

Рис 5

```java
public int hashCode() {
    int hash = PointsCount;
    for (int i = 0; i < PointsCount; ++i)
        hash ^= points[i].hashCode();
    return hash;
}
```

Рис 6

Задание выполнено.

Task 3

В классе LinkedListTabulatedFunction аналогично (рис 7). В методе equals() если функция представлена в виде LinkedListTabulatedFunction будет к точкам обращаться напрямую к полям вместо метода getPoint() (рис 8).

```
public String toString() {
    String s = "";
    FunctionNode node = head;
    for (int i = 0; i < size - 1; ++i, node = node.next)
        s += node.point.toString() + ", ";
    s += node.point.toString();
    return "{" + s + "}";
}

public int hashCode() {
    int hash = size;
    hash ^= head.point.hashCode();
    for (FunctionNode node = head.next; node != head; node = node.next)
        hash ^= node.point.hashCode();
    return hash;
}

public Object clone() {
    FunctionPoint[] points = new FunctionPoint[size];
    FunctionNode node = head;
    for (int i = 0; i < size; ++i, node = node.next)
        points[i] = (FunctionPoint) node.point.clone();
    return new LinkedListTabulatedFunction(points);
}
```

Рис 7

```java
public boolean equals(Object o) {
    if (o instanceof LinkedListTabulatedFunction && size == ((LinkedListTabulatedFunction) o).size) {
        FunctionNode node = ((LinkedListTabulatedFunction) o).head;
        if (!node.point.equals(head.point)) return false;
        node = node.next;
        for (FunctionNode nodeThis = head.next; nodeThis != head; nodeThis = nodeThis.next, node = node.next)
            if (!nodeThis.point.equals(node.point))
                return false;
        return true;
    }
    else if (o instanceof TabulatedFunction && size == ((TabulatedFunction) o).getPointsCount()) {
        FunctionNode node = head;
        for (int i = 0; i < size; ++i, node = node.next)
            if (!node.point.equals(((TabulatedFunction) o).getPoint(i)))
                return false;
        return true;
    }
    return false;
}
```

Рис 8

Задание выполнено.

Task 4

Внесем метод clone() в интерфейс TabulatedFunction (рис 9).

```java
Object clone();
```

Рис 9

Задание выполнено.

Task 5

Проверим работу написанных методов в классе Main.

Вывод мейна:

```
Проверка array:
{(0.0,6.0), (2.0,11.0), (4.0,16.0), (6.0,21.0), (8.0,26.0), (10.0,31.0), (12.0,36.0), (14.0,41.0), (16.0,46.0), (18.0,51.0)}
{(0.0,6.0), (2.0,11.0), (4.0,16.0), (6.0,21.0), (8.0,26.0), (10.0,31.0), (12.0,36.0), (14.0,41.0), (16.0,46.0), (18.0,51.0)}
{(0.0,6.0), (2.0,11.0), (4.0,16.0), (6.0,21.0), (8.0,26.0), (10.0,31.0), (12.0,36.0), (14.0,41.0), (16.0,46.0), (18.0,51.0)}

Сравнение для одинаковых классов
Функции равны
Сравнение для разных классов
Функции равны

hash для arr
1077673994
hash для arrCopy
1077673994
hash для list
1077673994

После изменения точки в arrCopy на 0.001
{(0.0,6.0), (2.0,11.0), (4.0,16.0), (6.0,21.0), (8.0,26.0), (10.0,31.0), (12.0,36.0), (14.0,41.0), (16.0,46.0), (18.0,51.0)}
{(0.0,6.0), (2.0,11.0), (4.0,16.0), (6.0,21.0), (8.0,26.0), (10.0,31.001), (12.0,36.0), (14.0,41.0), (16.0,46.0), (18.0,51.0)}
{(0.0,6.0), (2.0,11.0), (4.0,16.0), (6.0,21.0), (8.0,26.0), (10.0,31.0), (12.0,36.0), (14.0,41.0), (16.0,46.0), (18.0,51.0)}

Сравнение для одинаковых классов
Функции не равны
Сравнение для разных классов
Функции равны

hash для arr
1077673994
hash для arrCopy
-922006644
hash для list
1077673994
```

Рис 10

```
Проверка List:
{(0.0,6.0), (2.0,11.0), (4.0,16.0), (6.0,21.0), (8.0,26.0), (10.0,31.0), (12.0,36.0), (14.0,41.0), (16.0,46.0), (18.0,51.0)}
{(0.0,6.0), (2.0,11.0), (4.0,16.0), (6.0,21.0), (8.0,26.0), (10.0,31.0), (12.0,36.0), (14.0,41.0), (16.0,46.0), (18.0,51.0)}
{(0.0,6.0), (2.0,11.0), (4.0,16.0), (6.0,21.0), (8.0,26.0), (10.0,31.0), (12.0,36.0), (14.0,41.0), (16.0,46.0), (18.0,51.0)}

Сравнение для одинаковых классов
Функции равны
Сравнение для разных классов
Функции равны

hash для list
1077673994
hash для listCopy
1077673994
hash для arr
1077673994

После изменения точки в listCopy на 0.001
{(0.0,6.0), (2.0,11.0), (4.0,16.0), (6.0,21.0), (8.0,26.0), (10.0,31.0), (12.0,36.0), (14.0,41.0), (16.0,46.0), (18.0,51.0)}
{(0.0,6.0), (2.0,11.0), (4.0,16.0), (6.0,21.0), (8.0,26.0), (10.0,31.001), (12.0,36.0), (14.0,41.0), (16.0,46.0), (18.0,51.0)}
{(0.0,6.0), (2.0,11.0), (4.0,16.0), (6.0,21.0), (8.0,26.0), (10.0,31.0), (12.0,36.0), (14.0,41.0), (16.0,46.0), (18.0,51.0)}

Сравнение для одинаковых классов
Функции не равны
Сравнение для разных классов
Функции равны

hash для list
1077673994
hash для listCopy
-922006644
hash для arr
1077673994
```

Рис 11

Задание выполнено.