Универзитет „Св. Кирил и Методиј“ - Скопје
ФАКУЛТЕТ ЗА ИНФОРМАТИЧКИ НАУКИ И КОМПЈУТЕРСКО ИНЖЕНЕРСТВО

Аудиториски вежби X*

Наследување и полиморфизам

Напреден развој на софтвер

Интервју за работа

Дали сакате да работите во некоја од следните компании?

Прашање на интервју

The classic "eval" interview question

- Прв пат поставено на интервју во Амазон
- Прашањето е доста обемно и содржи многу важни вештини со кои треба да ги поседува еден квалитетен софтверски инженер:
  - **ООП дизајн**
  - **рекурзија**
  - бинарни дрва
  - **полиморфизам**
  - ...

Прашањето
1/2

Во одреден момент, кандидатот конечно сфаќа дека секоја аритметичка операција може да се претстави како бинарно дрво, ако препоставиме дека ги користиме само основните бинарни оператори како $+, -, *, /$. Листовите се броеви, а сите внатрешни јазли се оператори. Евалуацијата на изразот значи изминување на ова дрво.
Не го разбирате ова? Не е важно...
Ова е сеуште интересен проблем.

Прашањето
2/2

Првиот дел од прашањето е како од стринг кој претставува некаков аритметички израз (пр. “2 + (2)”) го трансформираме во дрво за изразот.
Вториот дел е: да речеме дека ова е проект за двајца луѓе и вашиот партнер е задолжен за трансформацијата на изразот во дрво, а на вас останува лесниот дел. Треба да напишете соодветни класи кои ќе ги употреби вашиот партнер за да ја заврши неговата задача.

Одговорот

Најдоброто решение користи полиморфизам!

Да се обидеме да одговориме на следните прашања?

- Кои се заедничките карактеристики или однесување на различни изрази?
- Кои се разликите?
- Кои методи/полиња треба да се наследат, а кои да се имплементираат посебно?

Expression interface

Решение 1/4

```
public interface Expression {

    public double eval();

    public String expression();

}
```

BinaryExpression abstract class

Решение 2/4

```
public abstract class BinaryExpression implements Expression
     {
    protected Expression left;
    protected Expression right;

    public BinaryExpression(Expression left, Expression
        right) {
        this.left = left;
        this.right = right;
    }

}
```

ValueNode class

Решение 3/4

```
public class ValueNode implements Expression {
    private double value;

    public ValueNode(double value) {
        this.value = value;
    }

    @Override
    public double eval() {
        return value;
    }

    @Override
    public String expression() {
        return String.format("%.2f", value);
    }

}
```

PlusExpression class

Решение 4/4

```
public class PlusExpression extends BinaryExpression {

    public PlusExpression(Expression left, Expression right)
         {
        super(left, right);
    }

    @Override
    public double eval() {
        return left.eval() + right.eval();
    }

    @Override
    public String expression() {
        return String
                .format("(%s + %s)", left.expression(),
                    right.expression());
    }

}
```

Демо

```java
public class Demo {
    public static void main(String[] args) {
        Expression expression = new ValueNode(2); // 2
        processExpression(expression);
        expression = new PlusExpression(new ValueNode(2), new ValueNode(3)); // (2 * 3)
        processExpression(expression);
        Expression mulExpression = new MultiplyExpression(new ValueNode(2), new ValueNode(3));
        Expression divExpression = new DivideExpression(new ValueNode(10), new ValueNode(2));
        expression = new PlusExpression(mulExpression, divExpression); // (2 * 3)
        processExpression(expression); // (2 * 3) + (10 / 2)
    }

    public static void processExpression(Expression expression) {
        System.out.println(expression.expression());
        System.out.println(String.format("Result: %.2f", expression.eval()));
    }
}
```

Материјали и прашања

Предавања, аудиториски вежби, соопштенија
courses.finki.ukim.mk

Изворен код на сите примери и задачи
bitbucket.org/tdelev/finki-nrs

Прашања и одговори
qa.finki.ukim.mk