به نام خدا

نام: فاطمه زهرا بخشنده **استاد: دکتر سعید پارسا**

شماره دانشجویی: 98522157 **درس اصول طراحی کامپایلر**

گزارش تمرین 2:

سوال چهارم:

کد این تمرین در refactoring.py قرار دارد. گرامر های جاوا در پوشه grammars اضافه شدند و فایل های lexer و مابقی فایل های ساخته شده از گرامر نیز در پوشه gen موجود است.

ابتدا با استفاده از argparser آدرس فایل ورودی و آدرس فایل خروجی که کد ریفکتور شده را در آن می‌نویسیم را ورودی می گیریم. برای این مقادیر، مقدار دیفالت نیز قرار داده شده است، و help نیز ایجاد شده است.

```
arg_parser = argparse.ArgumentParser(description='Java Refactor Comments')
arg_parser.add_argument(
    '-n', '--file',
    help='input java file path',
    default=r"Game.java"
)
arg_parser.add_argument(
    '-out', '--out',
    help='output java file path',
    default=r"GameRefactored.java"
)
```

```
PS D:\uni\compiler\assignments\HW2\HW2_98522157\Q4> python refactoring.py -h
usage: refactoring.py [-h] [-n FILE] [-out OUT]

Java Refactor Comments

optional arguments:
  -h, --help            show this help message and exit
  -n FILE, --file FILE  input java file path
  -out OUT, --out OUT   output java file path
```

سپس تابع refactor را با آرگیومنت های دریافت شده فراخوانی می کنیم. در تابع refactor با استفاده از حرکت روی token ها:

کامنت های multiline که تنها در یک خط نوشته شده اند را به کامنت inline تبدیل می کنیم.

کامنت‌های multiline چندخطی را تنها خط اولشان را نگه می‌داریم و آن ها را به به inline تبدیل می‌کنیم.

اگر چند کامنت inline یا multiline در چند خط پشت سر هم آمده باشند تنها کامنت اول را نگه میداریم و سایر خطوط را پاک می کنیم.

در آخر می توانیم مطمئن باشیم تمام کامنت ها یک خطی و inline هستند و هیچ دو کامنتی در دو خط پشت سر هم نیستند. در فایل Game.java حالات مختلف کامنت قرار دارد و می توان چک کرد که در کد ریفکتور شده تنها کامنت های یک خطی inline خواهیم داشت.

کد refactor شده را در آدرس فایل مقصد ذخیره می کنیم. همچنین محتوای کد ریفکتور شده را در خروجی چاپ می کنیم.

عکس از نحوه ورودی دادن به برنامه:

```
PS D:\uni\compiler\assignments\HW2\HW2_98522157\Q4> python refactoring.py -n Game.java -out GameRefactored.java
import java.util.Random;
 import java.util.ArrayList;

 //first line comment



 public abstract class Piece
 {
     protected String name;
     protected int x;
     protected int y;
```

آدرس فایل ورودی آن فایل Game.java است که در همین zip موجود است. آدرس فایل خروجی آن هم فایل GameRefactored.java است که در همین پوشه ایجاد شده است. و می بینیم که کد ریفکتور شده در آن قرار دارد.

خروجی کامل (محتوای فایل GameRefactored.java):

```
import java.util.Random;
import java.util.ArrayList;

//first line comment
```

```java
public abstract class Piece
{
    protected String name;
    protected int x;
    protected int y;
    protected boolean[][] flag = new boolean[8][4];

    protected Piece(String name, int x, int y)
    {
        this.name = name;
        this.x = x;
        this.y = y;
    }

    protected abstract void move();
}

// block comment line 1

public class Castle extends Piece
{
    protected Castle(String name, int x, int y)
    {
        super(name, x, y);
        this.flag[this.y][this.x] = true;
    }

    // block comment
    @Override
    protected void move()
    {
        switch ((int) (Math.random() * 3))
        {
            case 0: move0(); break;
            case 1: move1(); break;
            case 2: move2(); break;
        }
    }

    private void move0()
    {
        // line comment
        if (this.y < 7)
        {
            System.out.printf("x:%d y:%d ==> ", this.x, this.y);
```

```
            this.y++;
            this.flag[this.y][this.x] = true;
            System.out.printf("x:%d y:%d\n", this.x, this.y);
        }
        else move();
    }

    // block comment line 1

    private void move1()
    {
        if (this.x < 3)
        {
            System.out.printf("x:%d y:%d ==> ", this.x, this.y);
            this.x++;
            this.flag[this.y][this.x] = true;
            System.out.printf("x:%d y:%d\n", this.x, this.y);
        }
        else move();
    }

    // line comment

    private void move2()
    {
        if (this.x > 1)
        {
            System.out.printf("x:%d y:%d ==> ", this.x, this.y);
            this.x--;
            this.flag[this.y][this.x] = true;
            System.out.printf("x:%d y:%d\n", this.x, this.y);
        }
        else move();
    }
}
```