

```
package refactorizacion;

/**
 *
 * @author Daniel Castro Cruz
 */
import java.util.Random;
public class Dice
{ public static final int DEFAULT_SIDES = 6;

    private static Random ourRandNumGen = new Random();

    private final int iMyNumSides;
    private int iMyResult;
```

En la imagen podemos ver tres errores. El primero sería que deberíamos dejar un espacio entre el import y la clase para mayor legibilidad.

El segundo fallo es que la llave se abre en una línea diferente a la clase.

Además, el espacio entre las variables Random y iMyNumSides puede incluso dificultar la legibilidad.

Por último, iMyNumSides es una constante, por lo que el nombre debería estar en mayúsculas.

```
public Dice(int numSides) {
    assert numSides > 1 : "Violation of precondition: numSides = " +
    IMYNUMSIDES = numSides;
    iMyResult = 1;
    assert getResult() == 1 && getNumSides() == numSides;
}
```


Personalmente añadiría un espacio entre el constructor y el assert y quitaría el espacio entre el assert y la siguiente línea.

Además la línea del assert es demasiado larga, podemos dividirla en dos.

```
public Dice(int numSides, int result) {
    assert numSides > 1 && 1 <= result && result <= numSides : "Violat
    IMYNUMSIDES = numSides;
    iMyResult = result;
}
```


Y prácticamente lo mismo en este constructor.

```
public Dice(int numSides, int result) {
    assert numSides > 1 && 1 <= result && result <= numSides : "Violat
    IMYNUMSIDES = numSides;
    iMyResult = result;
}
```



```
public boolean equals(Object otherObj) {
    boolean result = true;
    if (otherObj == null)
        result = false;
    else if (this == otherObj)
        result = true;
    else if (this.getClass() != otherObj.getClass())
        result = false;
    else {
        Dice otherDie = (Dice) otherObj;
        result = this.iMyResult == otherDie.iMyResult && this.IMYNUMS
    }
    return result;
}

public String toString() {
    return "Num sides " + getNumSides() + " result " + getResult();
}
```



Estos métodos los pondría con los demás métodos para mayor legibilidad, no al final.
Además separaría en dos líneas la sentencia señalada ya que es un poco larga.

```
package refactorizacion;
```

```
/**
 *
```

```
 * @author Daniel Castro Cruz
```

```
 */
```

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

```
public class Dice {
```

```
    public static final int DEFAULT_SIDES = 6;
```

```
    private static Random ourRandNumGen = new Random();
```

```
    private final int IMYNUMSIDES;
```

```
    private int iMyResult;
```

```
    public Dice() {
        this(DEFAULT_SIDES);
    }
```

```
    public Dice(int numSides) {
        assert numSides > 1 : "Violation of precondition: numSides = "
            + numSides + "numSides must be greater than 1";
        IMYNUMSIDES = numSides;
        iMyResult = 1;
        assert getResult() == 1 && getNumSides() == numSides;
    }
```

```
    public Dice(int numSides, int result) {
        assert numSides > 1 && 1 <= result
            && result <= numSides : "Violation of
precondition";
        IMYNUMSIDES = numSides;
        iMyResult = result;
    }

    public int roll() {
        iMyResult = ourRandNumGen.nextInt(IMYNUMSIDES) + 1;

        assert (1 <= getResult()) && (getResult() <= getNumSides());

        return iMyResult;
    }

    public boolean equals(Object otherObj) {
        boolean result = true;
        if (otherObj == null)
            result = false;
        else if (this == otherObj)
            result = true;
        else if (this.getClass() != otherObj.getClass())
            result = false;
        else {
            Dice otherDie = (Dice) otherObj;
            result = this.iMyResult == otherDie.iMyResult
                && this.IMYNUMSIDES == otherDie.IMYNUMSIDES;
        }
        return result;
    }

    public String toString() {
        return "Num sides " + getNumSides() + " result " + getResult();
    }

    public int getNumSides() {
        return IMYNUMSIDES;
    }

    public int getResult() {
        return iMyResult;
    }
}
```