A basic process to writing code

- ✓Figure out what the class is suppose to do
- ✓List the instance variables and methods
- ✓Write pseudo code for the methods
- ✓Write test code for the methods
- ✓Implement the class
- ✓Test the methods
- ✓Debug and implement as needed

A Simple Battleship like game: SimpleDotCom Game

In this game, we kill off dot coms vs battleships in as fewest number of guesses as possible. We'll continue to build it as we go and it will take several refactoring iterations to accomplish all of its goals

- Primary Objective: When the program is launched, it will launch on a virtual 7 x 7 grid
- We haven't learned about GUIs yet so this version will use the command line
- We will get a response of hit, miss or You sunk Pets.com or whatever dot com is sunk
- The game ends when three dot coms have been sunk and you will see your rating
- Each dot com takes up three cells

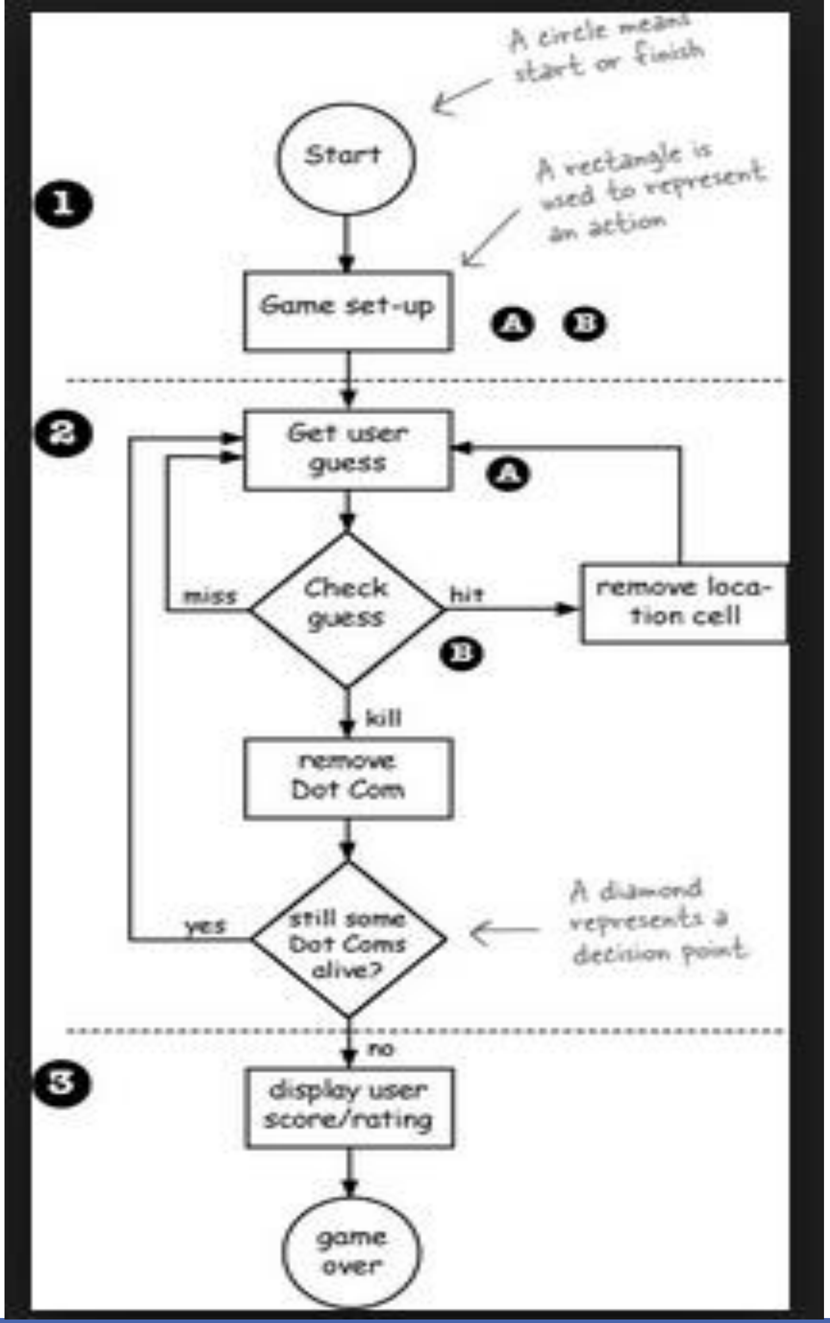
A							
B Go2.com							
C							
D		Pets.com					
E							
F							
G				AskMe.com			
	0	1	2	3	4	5	6

7 x 7 grid – virtual game board

High-Level Design

We need Classes and Methods **Figure out the general flow of the game**

1. **User Starts the game**
   - A. Game create 3 dot coms
   - B. Game places 3 dot coms on the virtual grid
2. **Game play begins** – repeat the following until there are no more dot coms
   - A. Prompt the user for a guess (eventually…)
   - B. Check the user’s guess to look for a hit, miss or kill – Take appropriate action:
     - a. If hit – delete a cell
     - b. If a kill – delete the dot com
3. **Game finishes**
   - A. Give user a rating



A circle means start or finish
Start
A rectangle is used to represent an action
1
Game set-up
A
B
2
Get user guess
A
miss
Check guess
hit
remove location cell
B
kill
remove Dot Com
yes
still some Dot Coms alive?
A diamond represents a decision point
no
3
display user score/rating
game over

High-Level Design (2)

Figure out what kind of objects we need

- Focus on things – What do we need?
- We'll need at least 2 classes – a game class and a DotCom class

Let's start with a simple version

- 1 dimension array
- 1 dot com
- No instance variables
- The whole game is coded in main()

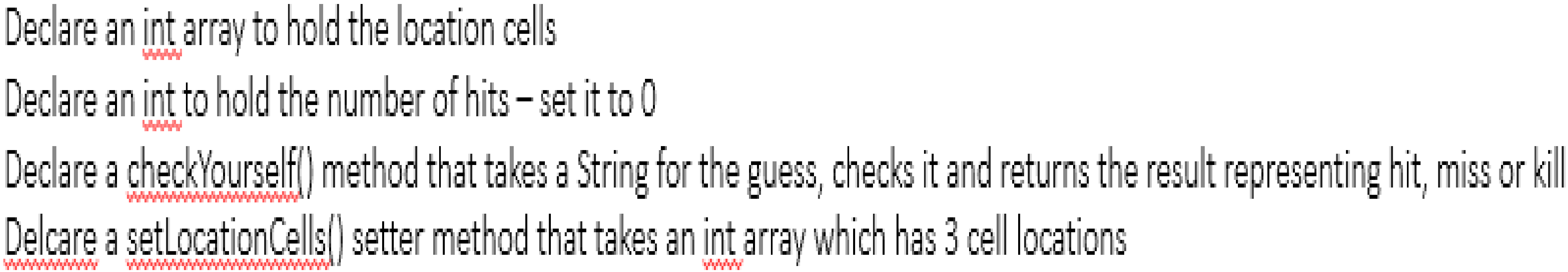
Write the Pseudo Code

What variables need to be declared?

What methods do we need? What actions does the Business Requirements tell us that must happen?

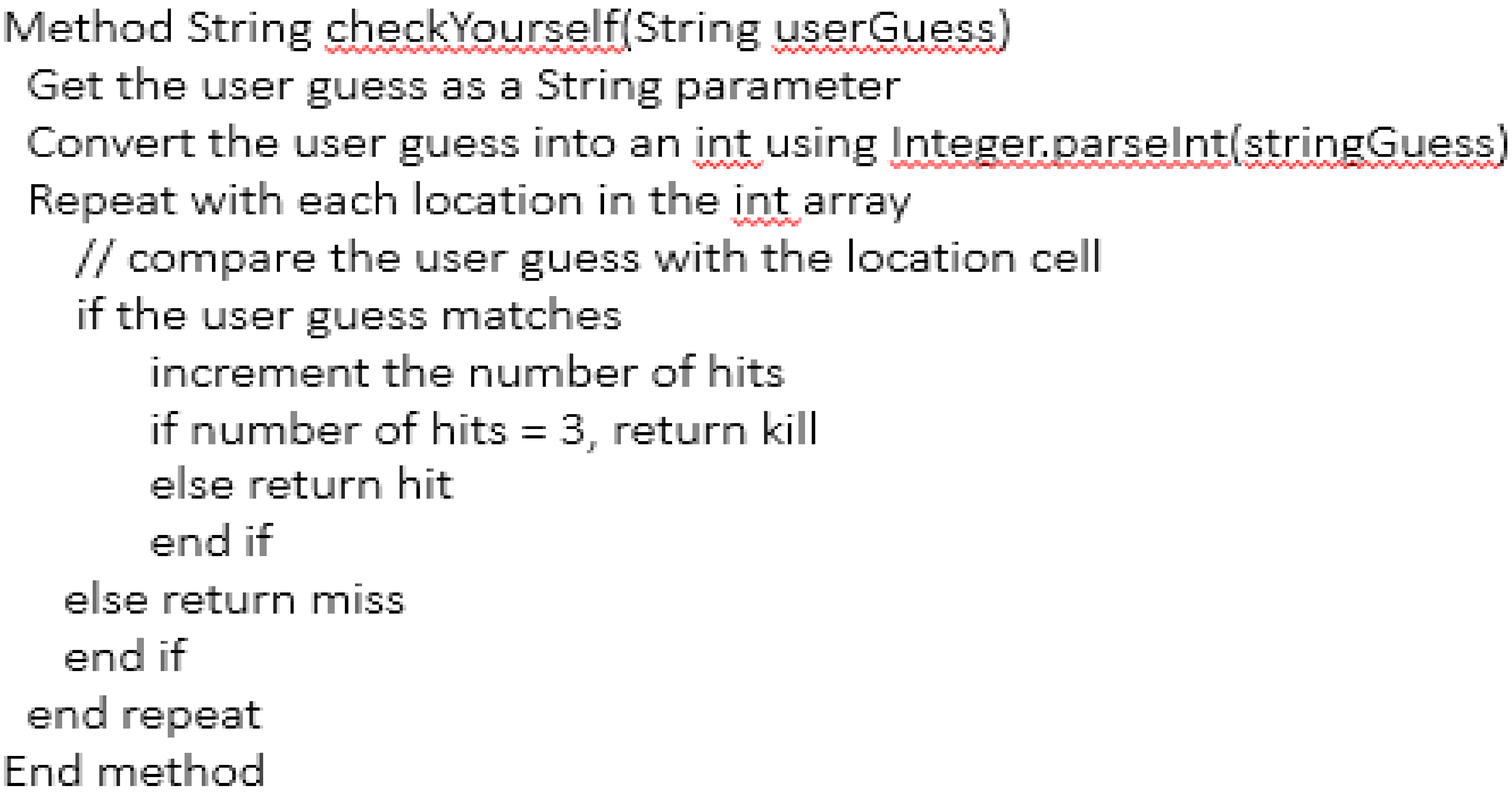
What are the attributes and behavior of the methods? Are they setters or getters?

Pseudo Code – Declare Statements



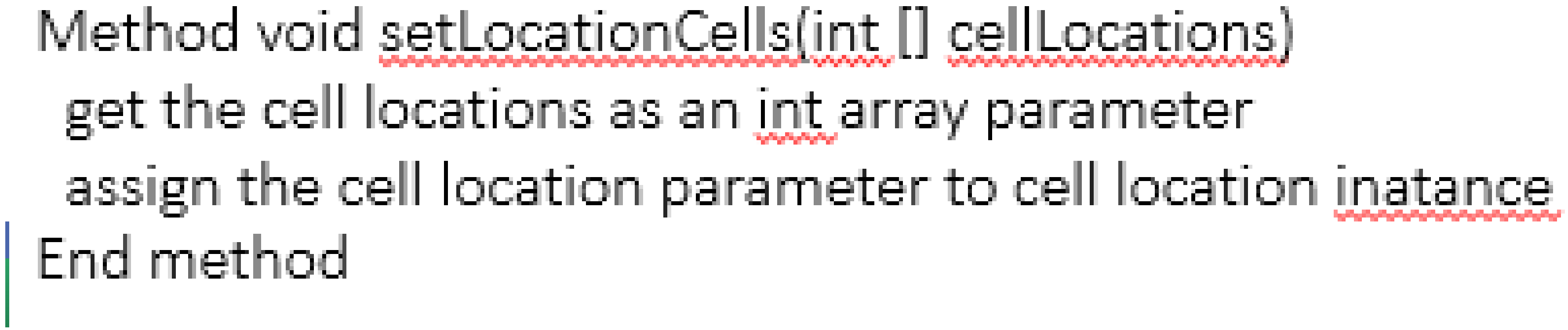
```
Declare an int array to hold the location cells
Declare an int to hold the number of hits – set it to 0
Declare a checkYourself() method that takes a String for the guess, checks it and returns the result representing hit, miss or kill
Delcare a setLocationCells() setter method that takes an int array which has 3 cell locations
```

Pseudo Code – Method checkYourself



```
Method String checkYourself(String userGuess)
  Get the user guess as a String parameter
  Convert the user guess into an int using Integer.parseInt(stringGuess)
  Repeat with each location in the int array
      // compare the user guess with the location cell
      if the user guess matches
            increment the number of hits
            if number of hits = 3, return kill
            else return hit
            end if
     else return miss
     end if
  end repeat
End method
```

Psuedo Code – setLocation Method



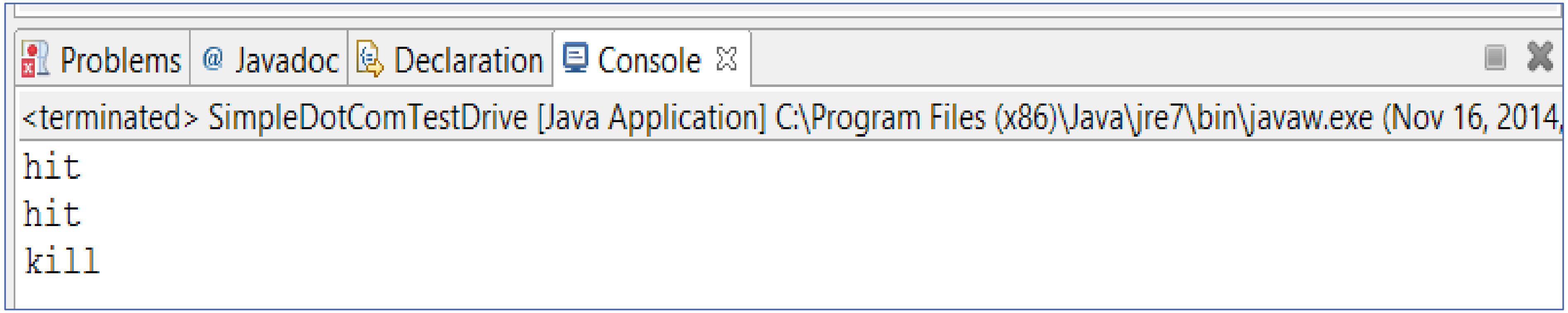
```
Method void setLocationCells(int [] cellLocations)
  get the cell locations as an int array parameter
  assign the cell location parameter to cell location inatance
End method
```

Let’s Code 2 Modules

Using the DotComGame.pdf on google group, complete the following:

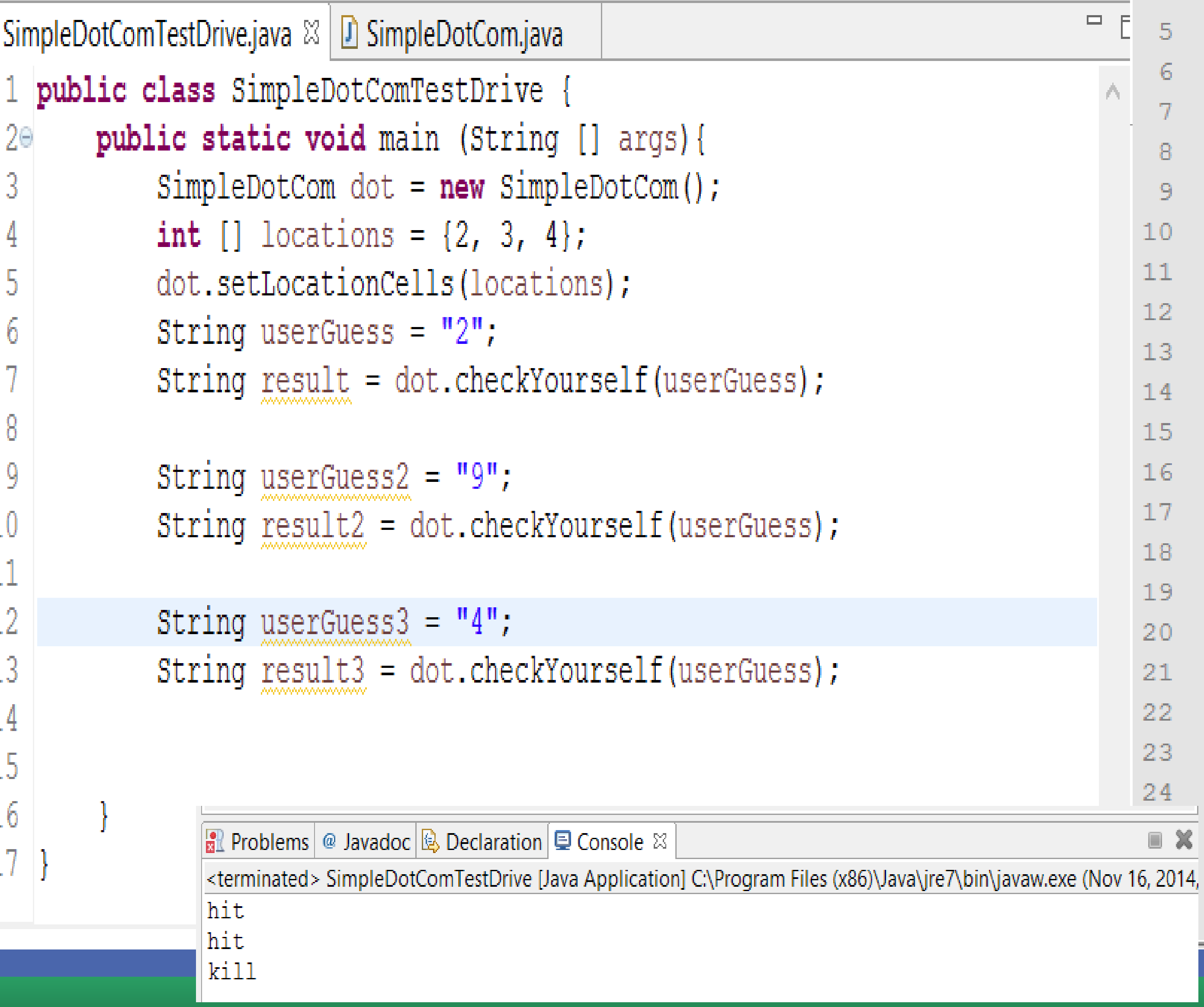
1. Create a SimpleDotCom.java class based on the requirements

2. Create a SimpleDotComTestDrive.java file to test the class



```
<terminated> SimpleDotComTestDrive [Java Application] C:\Program Files (x86)\Java\jre7\bin\javaw.exe (Nov 16, 2014,
hit
hit
kill
```

SimpleDotCom Game



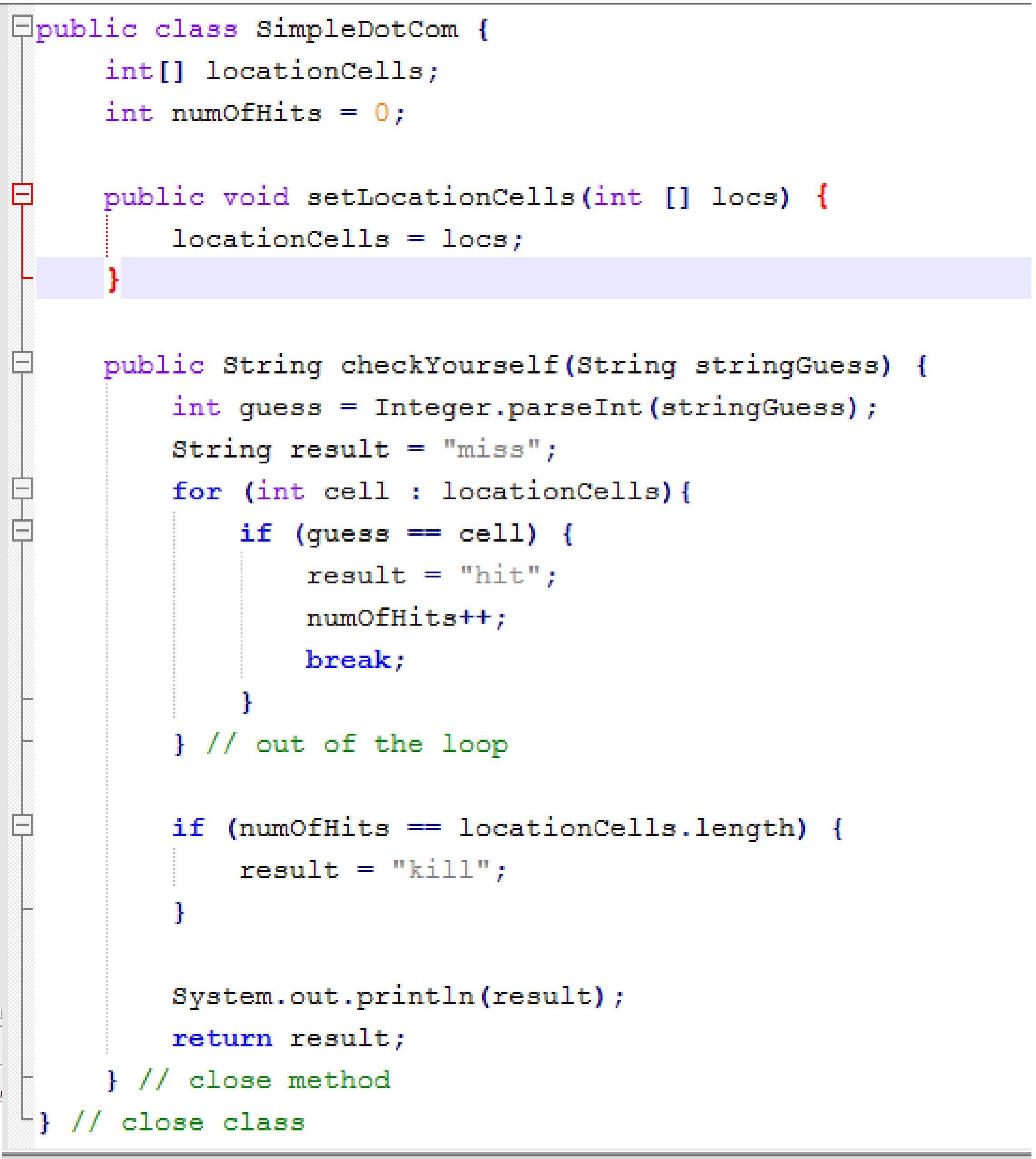
```java
public class SimpleDotComTestDrive {
    public static void main (String [] args){
        SimpleDotCom dot = new SimpleDotCom();
        int [] locations = {2, 3, 4};
        dot.setLocationCells(locations);
        String userGuess = "2";
        String result = dot.checkYourself(userGuess);

        String userGuess2 = "9";
        String result2 = dot.checkYourself(userGuess);

        String userGuess3 = "4";
        String result3 = dot.checkYourself(userGuess);


    }
}
```

```
<terminated> SimpleDotComTestDrive [Java Application] C:\Program Files (x86)\Java\jre7\bin\javaw.exe (Nov 16, 2014,
hit
hit
kill
```



```java
public class SimpleDotCom {
    int[] locationCells;
    int numOfHits = 0;

    public void setLocationCells(int [] locs) {
        locationCells = locs;
    }

    public String checkYourself(String stringGuess) {
        int guess = Integer.parseInt(stringGuess);
        String result = "miss";
        for (int cell : locationCells){
            if (guess == cell) {
                result = "hit";
                numOfHits++;
                break;
            }
        } // out of the loop

        if (numOfHits == locationCells.length) {
            result = "kill";
        }

        System.out.println(result);
        return result;
    } // close method
} // close class
```

Let’s Examine the Code in our game

checkYourself()

Line 10: converts the string to an integer using Integer.parseInt()

Line 12: for (int cell : locationCells) {} – this is an enhanced for loop that began with Java 5 – can use the original format too

The colon : means “in” so the statement means “for each int value in the array locationCells – This new format of the for loop iterates over an array while looping.

Each time thru the loop, the next element in the array will be assigned to the variable cell (holds only one element of the array)

Continuing on with DotCom Game….

So far, we've created the SimpleDotCom class and a test drive program.

We need to create the real game program

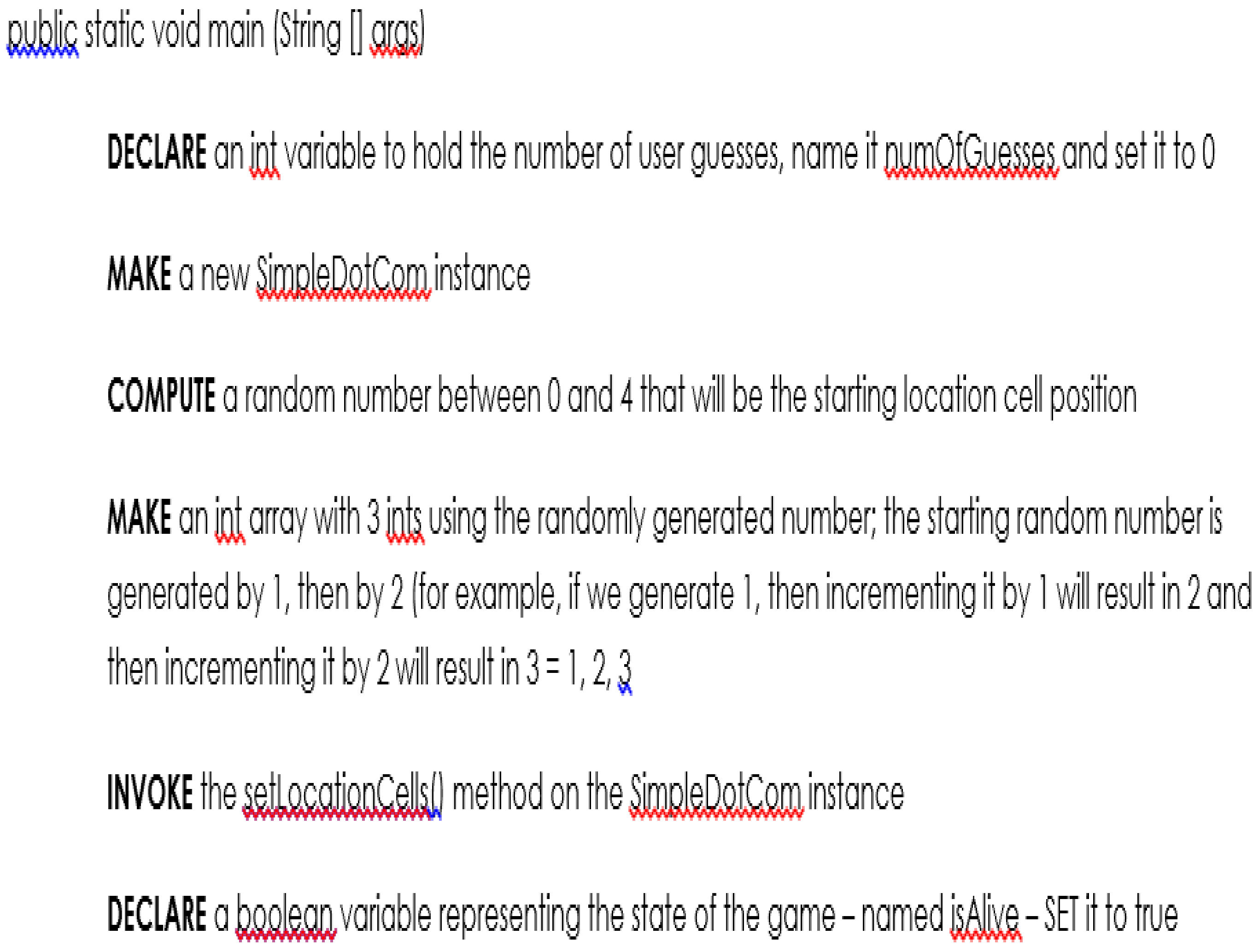
SimpleDotCom Game – Part 2

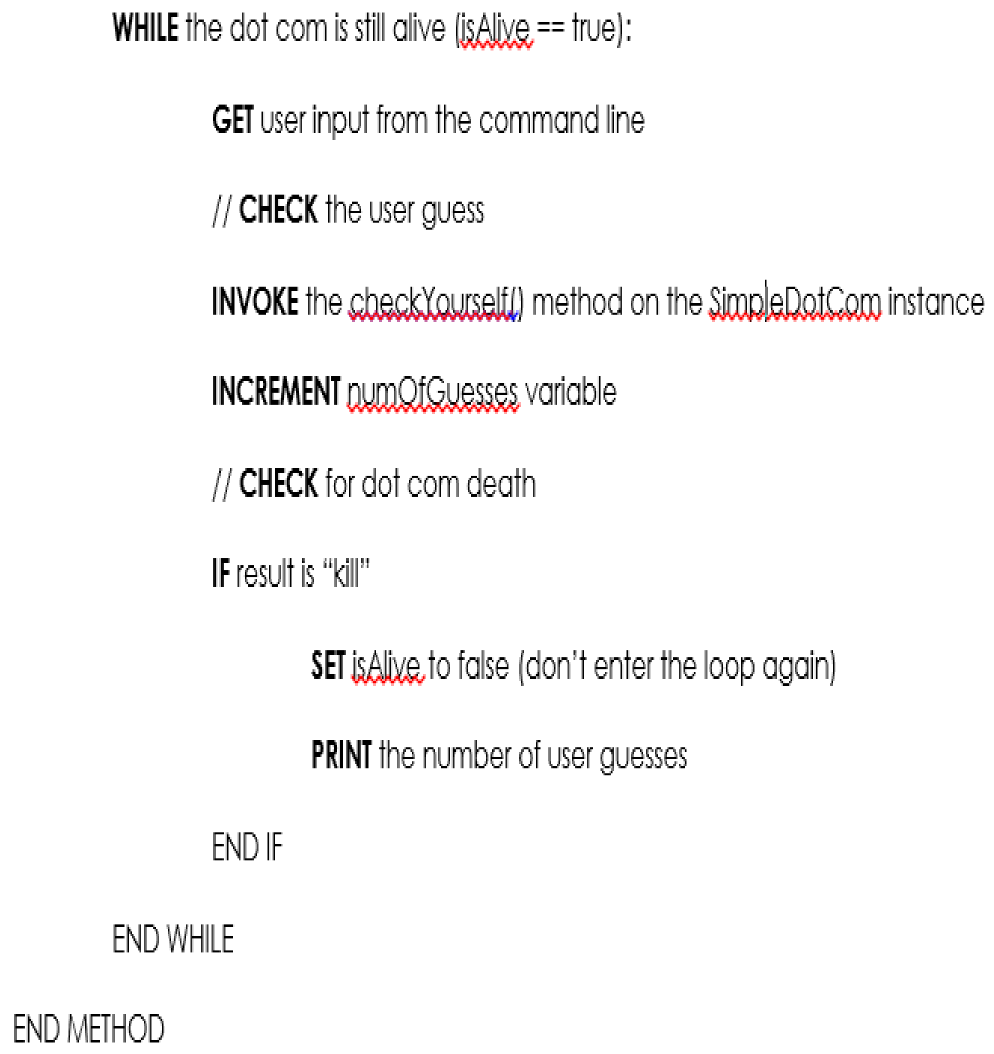
STEP 1 – Create the Pseudo Code for SimpleDotComGame class

Referring back to the code you created for SimpleDotCom.java and the software requirements, create the game's pseudo code

You've been given a few lines to start…. Page 1

SimpleDotCom Game – Pseudo Code Solution – Page 2





```
public static void main (String [] args)

    DECLARE an int variable to hold the number of user guesses, name it numOfGuesses and set it to 0

    MAKE a new SimpleDotCom instance

    COMPUTE a random number between 0 and 4 that will be the starting location cell position

    MAKE an int array with 3 ints using the randomly generated number; the starting random number is generated by 1, then by 2 (for example, if we generate 1, then incrementing it by 1 will result in 2 and then incrementing it by 2 will result in 3 = 1, 2, 3

    INVOKE the setLocationCells() method on the SimpleDotCom instance

    DECLARE a boolean variable representing the state of the game – named isAlive – SET it to true

    WHILE the dot com is still alive (isAlive == true):

        GET user input from the command line

        // CHECK the user guess

        INVOKE the checkYourself() method on the SimpleDotCom instance

        INCREMENT numOfGuesses variable

        // CHECK for dot com death

        IF result is "kill"

            SET isAlive to false (don't enter the loop again)

            PRINT the number of user guesses

        END IF

    END WHILE

END METHOD
```

SimpleDotComGame – Real Code

New Statements

- Math.random()
  - Java class Math with random () method that generates random numbers

```
int randomNum = (int) (Math.random() * 5);
```

- getUserInput()
  - A helper class that we've given you to accept user input from the command line (console in Eclipse)

```
String guess = helper.getUserInput(“enter a number”);
```

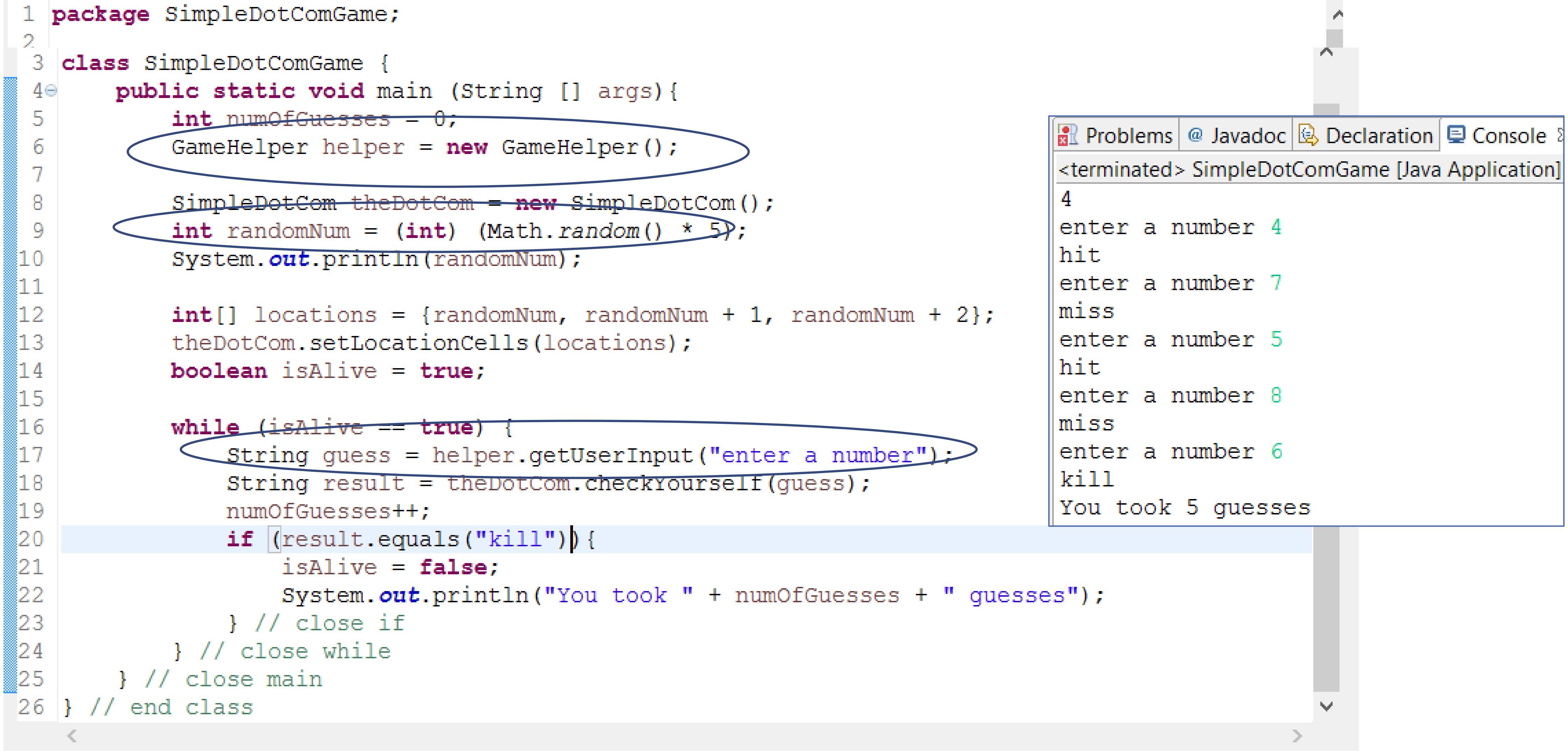
SimpleDotCom Game – Part 2

STEP 2 – Create the Real Code for SimpleDotComGame class

1st – code the helper class GameHelper.java exactly as shown on page three – it uses new features that we will talk about later as to not muddy the process of problem solving and translating pseudo code to real code

2nd – code your SimpleDotComGame.java file using your pseudo code – test as a Java Application by enter user input into the console

SimpleDotComGame.java Solution – Page 4



```java
package SimpleDotComGame;

class SimpleDotComGame {
    public static void main (String [] args){
        int numOfGuesses = 0;
        GameHelper helper = new GameHelper();

        SimpleDotCom theDotCom = new SimpleDotCom();
        int randomNum = (int) (Math.random() * 5);
        System.out.println(randomNum);

        int[] locations = {randomNum, randomNum + 1, randomNum + 2};
        theDotCom.setLocationCells(locations);
        boolean isAlive = true;

        while (isAlive == true) {
            String guess = helper.getUserInput("enter a number");
            String result = theDotCom.checkYourself(guess);
            numOfGuesses++;
            if (result.equals("kill")){
                isAlive = false;
                System.out.println("You took " + numOfGuesses + " guesses");
            } // close if
        } // close while
    } // close main
} // end class
```

```
<terminated> SimpleDotComGame [Java Application]
4
enter a number 4
hit
enter a number 7
miss
enter a number 5
hit
enter a number 8
miss
enter a number 6
kill
You took 5 guesses
```