Some slides contain lots of animation; you must be in class to fully understand them.

Writing a Java Program (1/2)

** using our new programming skills to write a real program (and learning some new ones on the way!)
** arrays
** enhanced **`for`** loop

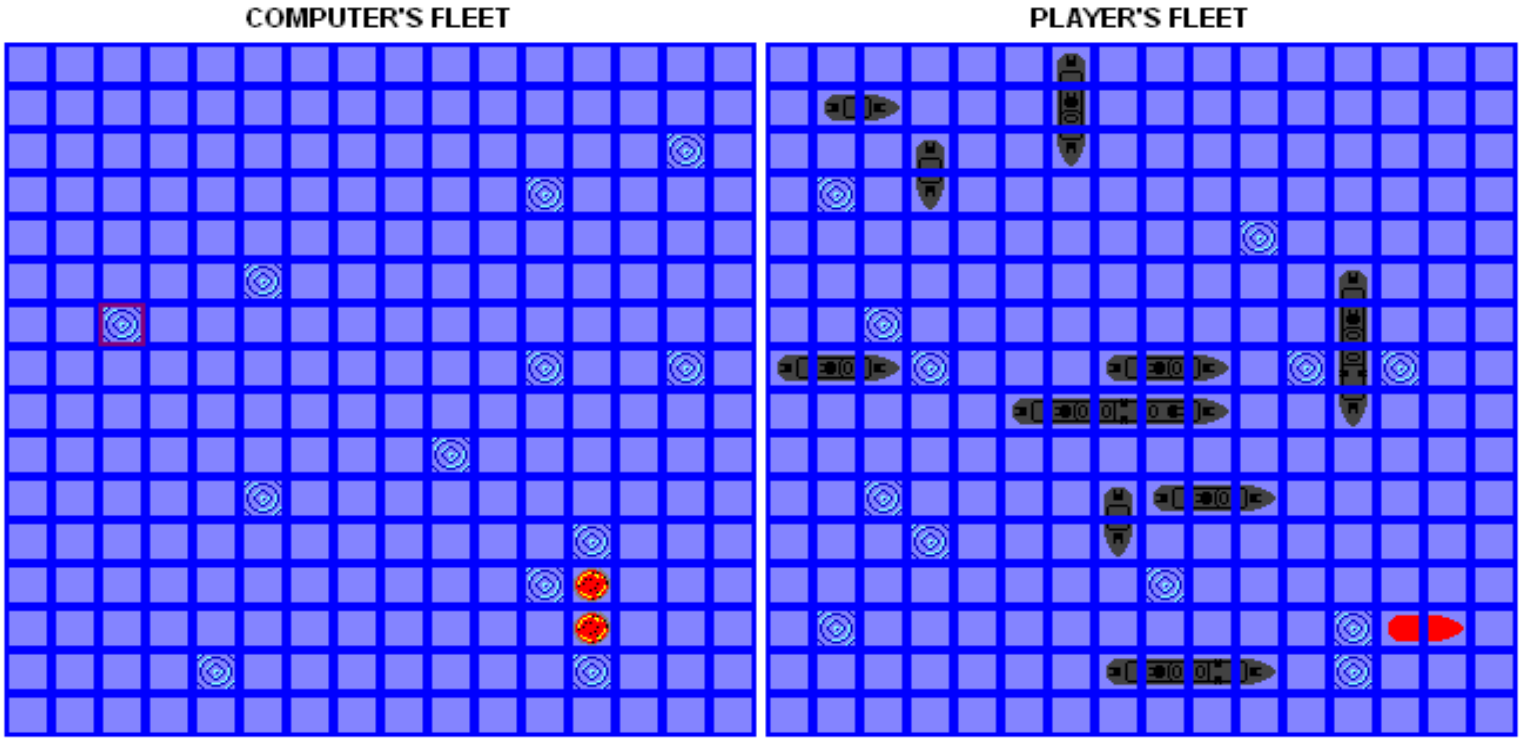
Chapter 5 (*) – "Head First Java" book

(*) *Example from the chapter*

A Battleship Style Game: "Sink a Dot Com“ (1/2)

Your Task:

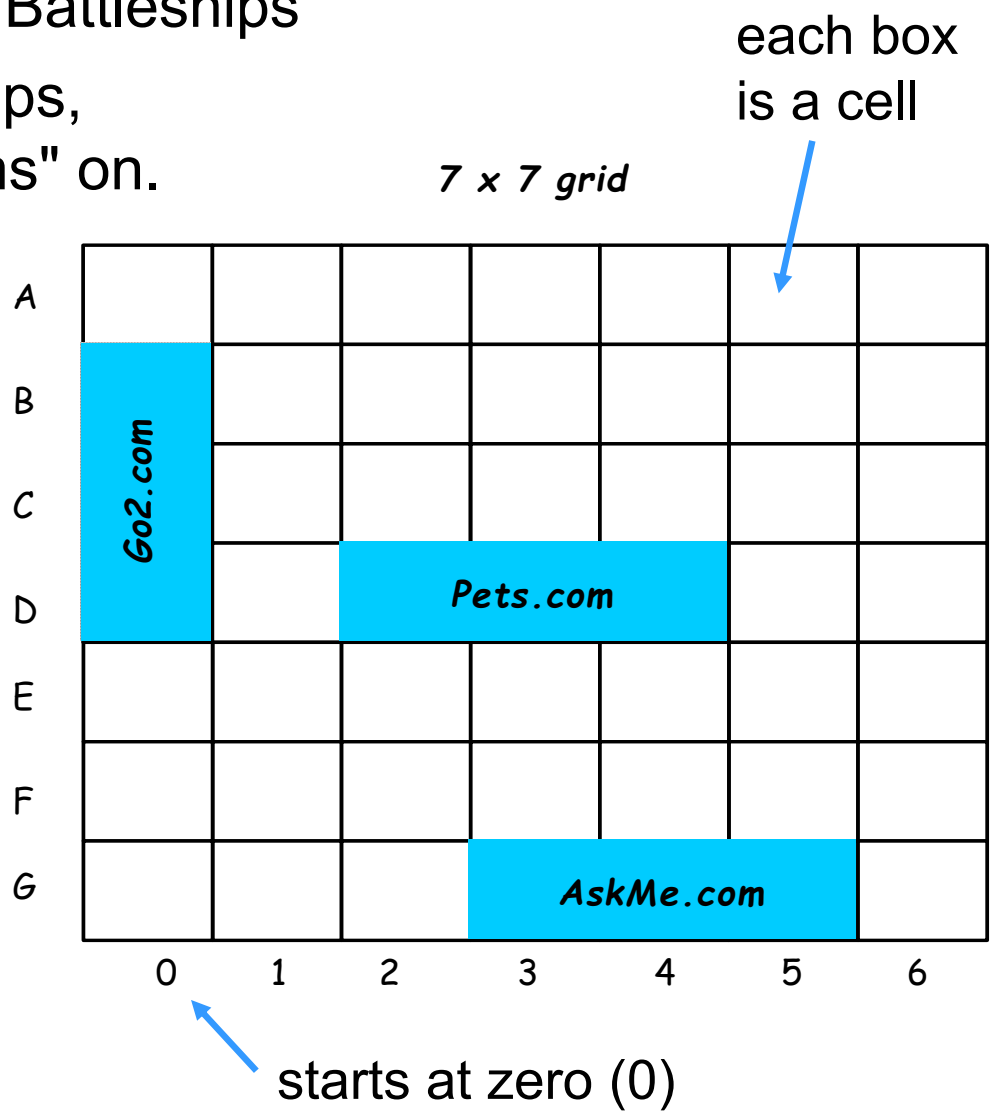
- Build a game like the battleship guessing game.
- The basic idea of battleship is a number of ships are placed on a grid world board.
- In the board game version, there are 2 players who each have their own board:



From http://www.superkids.com/aweb/tools/logic/bship/.

A Battleship Style Game: "Sink a Dot Com“ (2/2)

- Here, we will sink "Dot Coms“, instead of Battleships
- Just play against the computer's battleships, i.e. no board of your own to put "Dot Coms" on.



GOAL

- Sink all the computer's “Dot Coms” in the fewest number of steps/guesses.
- You are given a rating, based on how well you perform.

SETUP

- A virtual 7x7 board with 3 randomly placed “Dot Coms”.
- After that, the player should be prompted to enter their first guess.

Queen Mary University of London

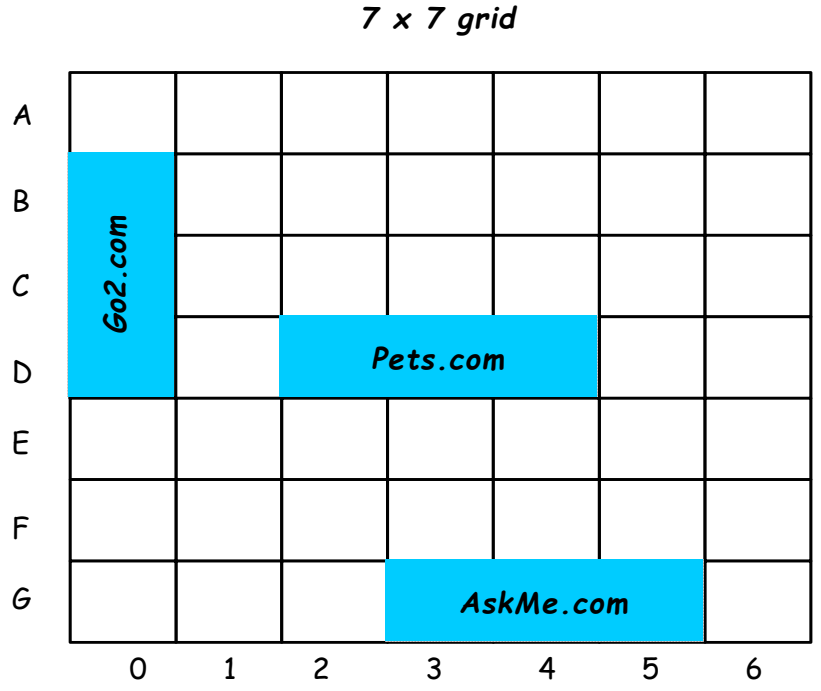
Dot Com = Company that conducts its business on the internet

"Sink a Dot Com" game: how to play and sample interaction

How to play:

- No GUI yet, as we do not know how to do that.
- Use the command line.
- Computer should prompt for a guess, and the player should enter a cell, e.g. **A3**, **C5**, etc.
- The computer should respond with hit or miss.
- After all 3 "Dot Coms" have been sunk, the game ends by printing out your rating.

```
>java DotComBust
Enter a guess  A3
miss
Enter a guess  D2
hit
Enter a guess  D3
hit
Enter a guess  D4
Ouch! You sunk Pets.com :(
kill
...
...
```

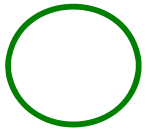


Program's high level design & Conversion to Flowchart

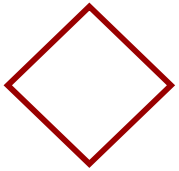
First …

- Work out the program interactions.
- Write everything in English, i.e. do not write Java code (for now).
- Define the general flow of the game, from the specification.

Then …

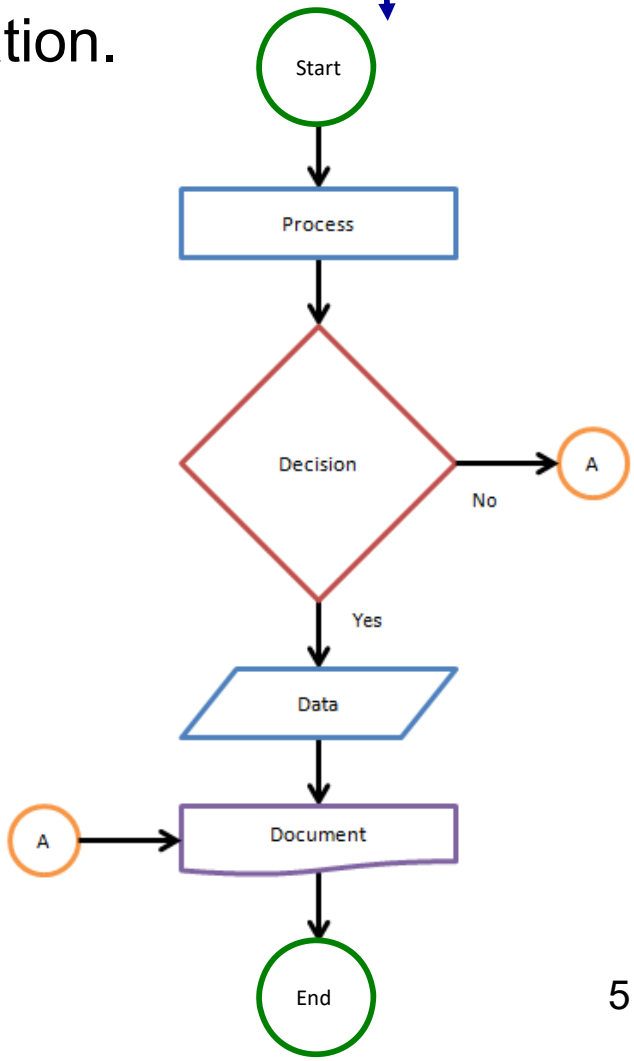


A circle means start or finish.



A diamond represents a decision point.

A rectangle represents an action point.



Game sequence

User starts the Game

Game creates three Dot Coms

Game places the three Dot Coms onto a virtual grid

Game Play begins

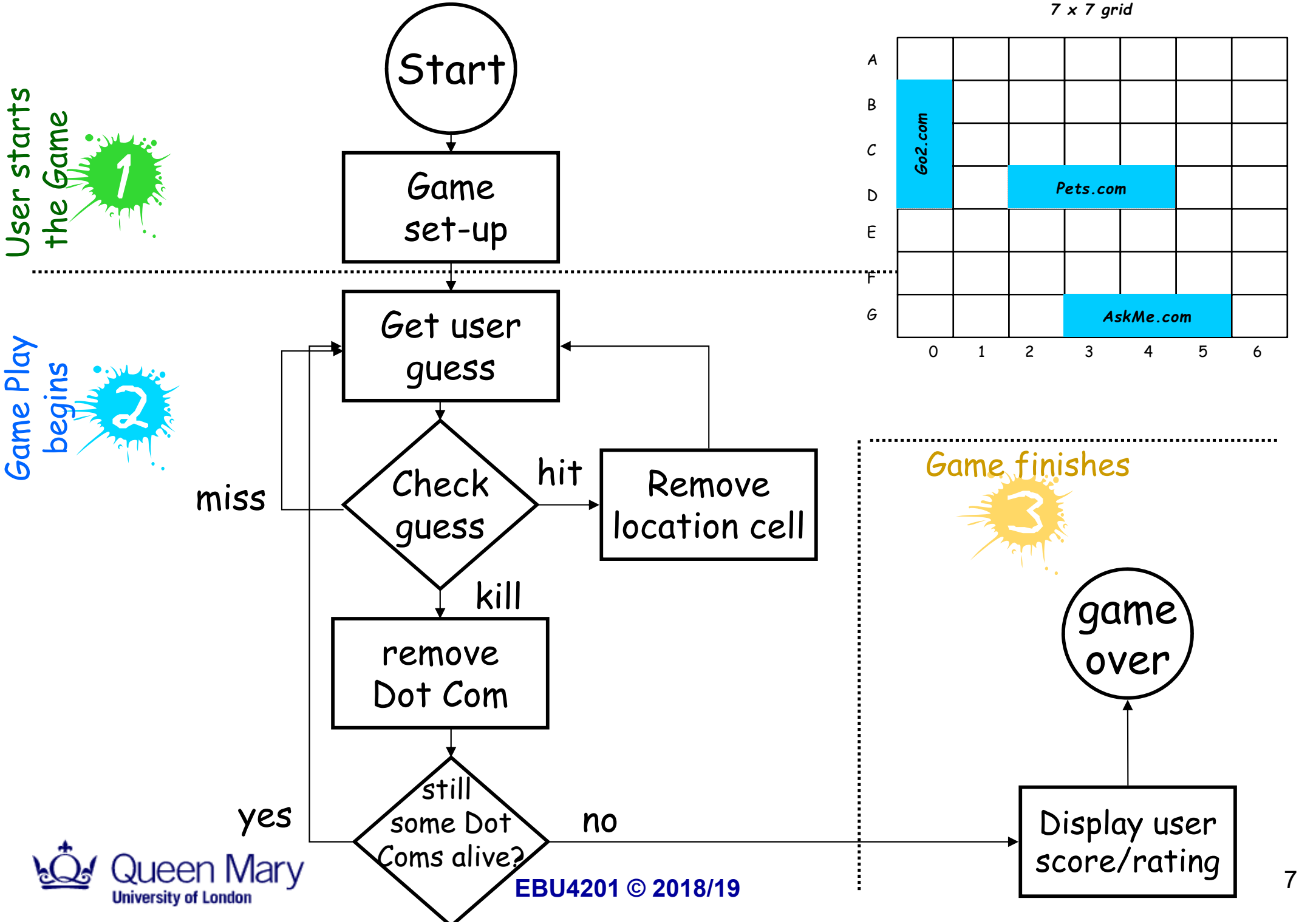
Repeat the following until there are no more Dot Coms:

Prompt user for a guess ("A2", "C0", etc)

Check the user guess against all Dot Coms to look for hit, miss or kill. Take appropriate action: if hit, delete cell. If kill, delete Dot Com.

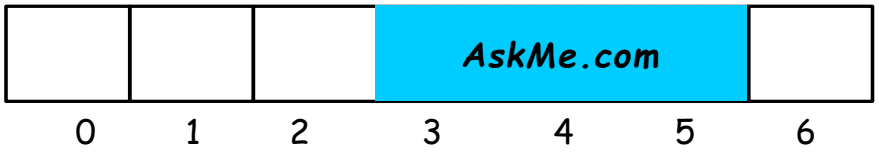
Game finishes

Give the user a rating based on the number of guesses


User starts the Game
1
Start
Game set-up
Get user guess
Check guess
miss
hit
Remove location cell
kill
remove Dot Com
still some Dot Coms alive?
yes
no
Game Play begins
2
7 x 7 grid
A
B
C
D
E
F
G
0
1
2
3
4
5
6
Go2.com
Pets.com
AskMe.com
Game finishes
3
game over
Display user score/rating

Developing the classes for the “Sink a Dot Com” game

- Always try to make things simpler on the first iteration of your programs!
  - You can go back and improve what you know works ...
- An even simpler "Sink a Dot Com":
  - Only 1 “Dot Com” on the board, instead of 3.
  - No 2D grid; put the “Dot Com” just in a row.
  - The goal is still the same.



- We are going to need 2 classes (at least – see later):
  - **`Game`** Class
  - **`DotCom`** Class
- Simple version:
  - The **`Game`** class has no instance variables and all the "coding stuff" is in **`main()`**.
  - When launched, it will create 1 instance of a **`DotCom`** class and choose a location for it.
    - in 3 consecutive array cell locations
  - After that, play the game!

UML Diagrams? Sample Interaction?

UML & Example Game Interaction

SimpleDotCom
int[] locationCells; int numberOfHits;
String checkYourself(String guess); void setLocationCells(int[] loc)

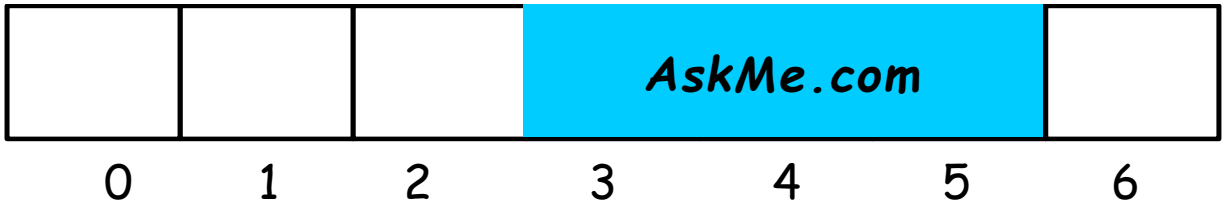
SimpleDotComGame
void main(String[] args)

A complete game interaction:

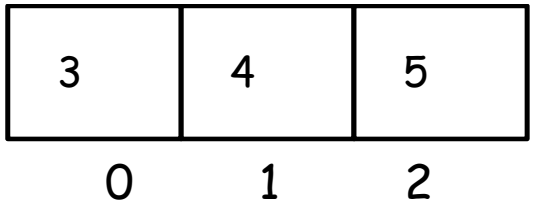
```
>java SimpleDotComGame
enter a number   3
hit
enter a number   4
hit
enter a number   2
miss
enter a number   5
kill
You took 4 guesses
```

Rethinking the arrays in the game …

- Conceptually, we tend to think of the array like this:



- However, what we actually need is only:



Steps to develop a Java class (Revision)

1. Decide what the class is supposed to do.
2. List the instance variables and methods.
3. Write "prep code" for the methods.
4. Write test code for the methods.
5. Implement the class.
6. Test the methods.
7. Debug and re-implement, as necessary.

"Prep code" for SimpleDotCom

- A form of pseudocode, to help focus on the logic without the worry of syntax.

SimpleDotCom
int[] locationCells; int numberOfHits;
String checkYourself(String guess); void setLocationCells(int[] loc)

DECLARE an int array for location cells. Call it **locationCells**.

DECLARE an int to hold the number of hits. Call it **numberOfHits** and **SET** it to 0.

DECLARE a **checkYourself()** method that takes a String for the user's guess ("1","3",etc.), checks it, and returns a result representing a "hit", "miss", or "kill".

DECLARE a **setLocationCells()** setter method that takes an int array (which has the three cell locations as ints (2,3,4,etc.).

The checkYourself() method

METHOD: `String checkYourself(String userGuess)`

GET the user guess as a String parameter
CONVERT the user guess to an int
REPEAT with each of the location cells in the int array
//COMPARE the user guess to the location cell
IF the user guess matches
INCREMENT the number of hits
//FIND OUT if it was the final location cell:
IF number of hits is 3, RETURN "*kill*"
ELSE if it was not a kill, RETURN "*hit*"
ELSE the user guess did not match, so RETURN "*miss*"
END IF
END REPEAT
END METHOD

The `setLocationCells()` method

METHOD: `void setLocationCells(int[] cellLocations)`

GET the cell locations as an int array parameter.

ASSIGN the cell locations parameter to the cell location instance variable

END METHOD

Time to start coding!

- We have our **`SimpleDotCom`** class; shouldn't we just write it up and create our game?
  - No!
- We need to test it!
  - How do you test it? What would prove that it works?
  - Writing test code helps clarify what it is we want the method to do.

- The main thing we want to test is the **`checkYourself()`** method.
  - Basic idea of this method: check to see if the **`DotCom`** object has been hit.
  - To test it, we could:
    1. Instantiate a **`SimpleDotCom`** object.
    2. Assign it a location (i.e. an array of 3 **`int`** values, like **`{2,3,4}`**).
    3. Create a **`String`** to represent a user guess (e.g. **`"2"`**, **`"0"`**, etc).
    4. Invoke the **`checkYourself()`** method, passing it a fake guess.
    5. Print out the result and see if it is what we expected.

Test Code for SimpleDotCom class

```
public class SimpleDotComTest {
  public static void main(String[] args) {
    SimpleDotCom dot = new SimpleDotCom();
    int[] locations = {2, 3, 4};
    dot.setLocationCells(locations);

    String userGuess = "2"; // result should be "hit" or "kill"
    String result = dot.checkYourself(userGuess);
    String testResult = "failed"; // default value
    if (result.equals("hit") || result.equals("kill")) {
      testResult = "passed";
    }
    System.out.println(testResult);
  }
}
```

First version of SimpleDotCom class

```
public class SimpleDotCom {
  private int[] locationCells;
  private int numberOfHits = 0;

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

  public String checkYourself(String stringGuess) {
    int guess = Integer.parseInt(stringGuess);
    String result = "miss"; // default value
    for (int i : this.locationCells) {
      if (guess == i) {
        result = "hit";
        this.numberOfHits++;
        break;
      }
    }
    if (this.numberOfHits == this.locationCells.length) { result = "kill"; }
    System.out.println(result);
    return result;
  }
} // end of SimpleDotCom
```

equivalent to

```
for (int i=0; i<locationCells.length; i++) {
  if (guess == locationCells[i]) {
    // rest of the code as before
  }
}
```

The `main()` method in the `SimpleDotComGame` class

DECLARE: an int variable to hold the number of user guesses, named **`numOfGuesses`**. **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 incremented by 1, and that number incremented by 2 (e.g. 3,4,5).

INVOKE the **`setLocationCells()`** method on the SimpleDotCom instance.

DECLARE: a boolean variable representing the state of the game, named **`isAlive`**. **SET** it to true.

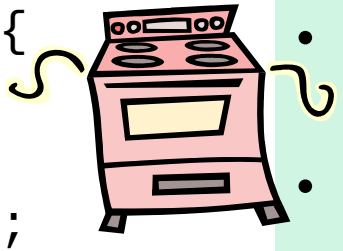
The `main()` method in the `SimpleDotComGame` class *(cont.)*

```
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 numberOfGuesses variable
     //CHECK for dot com death
     IF result is "kill"
        SET isAlive to false (no more looping)
        PRINT the number of guesses
     END IF
  END WHILE
END METHOD
```

The game's `main()` method

```
public static void main(String[] args) {
  int numOfGuesses = 0;
  GameHelper helper = new GameHelper();
  SimpleDotCom dot = new SimpleDotCom();
  int randomNum = (int) (Math.random() * 5);
  int[] locations = {randomNum, randomNum + 1,
                     randomNum + 2};
  dot.setLocationCells(locations);
  boolean isAlive = true;

  while (isAlive == true) {
    String guess = helper.getUserInput("Enter a number ");
    String result = dot.checkYourself(guess);
    numOfGuesses++;
    if (result.equals("kill")) {
      isAlive = false;
      System.out.println("You took " + numOfGuesses + " guesses");
    }
  }
}
```



`GameHelper` = pre-written "ready baked" class.

- We are not going to worry about how it works!
- It gets the input from the user; that is all we need to know (for now).

Testing the first version of the game …

A complete game interaction:

```
>java SimpleDotComGame
enter a number   3
hit
enter a number   4
hit
enter a number   2
miss
enter a number   5
kill
You took 4 guesses
```

This may be different, depending on **`Math.random()`**.

Another complete game interaction:

```
>java SimpleDotComGame
enter a number   3
hit
enter a number   3
hit
enter a number   3
kill
You took 3 guesses
```

Bug!

… and things for you to try out!