Java
John Marchand – jhmarchand@gmail.com
Team 4381 – Twisted Devils
https://github.com/jhmarchand/java4381

Week 3

Guessing Game

Lets Review the solution from last week. Heres a few questions to consider. Why do I set the guess to -1? Why dont I have a if statement before I tell them they got the right answer?

```
java.util.Scanner input = new java.util.Scanner(System.in);
int answer = (int) (Math.random() * 10 );
int guess = -1;

System.out.println("Pick a number between 0 and 9");

while ( guess != answer)
{
	// get the users guess
	System.out.println("Enter your guess");
	guess = input.nextInt();

	// check if too high or too low
	if ( guess > answer )
		System.out.println("Too high");

	if ( guess < answer )
		System.out.println("Too low");
}

System.out.println("Great Job, you got the right answer of " + answer);

input.close();
```

Who all kept track of how many guesses it took them? How I solved it. I wanted to keep track of their guesses which is a number. So I created my own int called guesses. Since in the beginning they haven't guessed any, I set it equal to 0. Then after each time I read their guess, I used the ++ to increase it by one. Then, at then end, I just printed it out.

Types or Else!

There are many other base types that we should know about. Lets take a look at a few of them.

Booleans – these hold either true or false. You can see how to declare them from the following examples. You can even set them equal to the result of a conditional expression like 5 > 6! Or use them in a if statement.

Lots of time, if the result of your if condition is false, you want to do something. You could always say if true do this, if false do that, but its easier to say if true do that, else do this. Look at the following syntax.

```
boolean roboticsIsCool = true;
boolean thisClassIsBoring = false;

if ( thisClassIsBoring )
	System.out.println("I want to go home");
else
	System.out.println("Lets Learn some Java");

boolean fiveIsLessThanSix = 5 < 6;
```

Doubles and floats - All of our numbers have been whole numbers so far. No decimal places. To use decimal places, we want to use doubles. Floats are like doubles, but only have half the precision. In general, just use doubles. Mixing ints, floats and doubles sometimes causes an error, and you may need to tell the compiler what you want. This is called casting.

```
double pi = 3.14;
double diameter = 3.0;
double circumference = pi * diameter;

// 10.7 is consider a double, so going down to
float floatVariable = (float) 10.7;

int radius = 5;
double area = pi * radius * radius;
System.out.println("Area = " + area );

//int intArea = pi * radius * radius;

int intPi = (int) pi;
int intArea = intPi * radius * radius;
System.out.println("Area = " + intArea
```

Chars - These are a single character and we use the single quote ‘ for a char and the “ for a string. We can add some chars together to make a string. There is a trick involved, and we’ll get to that later.

```
char myChar = 'J';
String message = "My Name is ";
message = message + myChar + 'o' + 'h' + 'n';
System.out.println(message);
```

Arrays - A array is a list of items. So if you want to store 3 ints together, you can. Then you can work with all of them in loops. One trick, arrays start with 0, so the first item in an array is the 0 item not the 1 item. Lets look at the following example that creates an array of three numbers and then sums them.

```
int[] intArray = new int[3];

intArray[0] = 3;
intArray[1] = 2;
intArray[2] = 7;

int sum = 0;

for ( int i = 0; i < intArray.length; i++ )
       sum = sum + intArray[i];

System.out.println(sum);
```

Strings

A string is what? A string of characters. A list of characters. An array of char. Here’s the start, lets work through this together. Given this string, how can we print out the 3 numbers?

```
String myString = "55,87,436";
char[] charArray = myString.toCharArray();
```

Tic Tac Toe

Consider these ways of holding a tic tac toe board. Which looks best to you? Pick one, and enter some default values. Then print out your board!

```
// here 0 is O, 1 is blank, 2 is X
int[] tic = new int[3]; // top row
int[] tac = new int[3]; // middle row
int[] toe = new int[3]; // bottom row

tic[0] = 0;
tic[1] = 2;
tic[2] = 1;
tac[0] = 2;
tac[1] = 0;
tac[2] = 1;
toe[0] = 2;
toe[1] = 1;
toe[2] = 0;

char[] topRow = new char[3];
char[] middleRow = new char[3];
char[] bottomRow = new char[3];

topRow[0] = 'X';
topRow[1] = ' ';
topRow[2] = 'O';
// and on and on.

char[] board = new char[9];
for( int i = 0; i < board.length ; i++)
      board[i] = ' ';
```

Extras

Start out your board blank, and let the user enter a value of 2,2 or something to put an x in the center. Then keep asking them so they can play again. Print out the board after each turn.

Don't let the user change a square – IE if they enter a square to mark, make sure its blank.

Can you tell if someone has won the game?

What do you think of multi-dimensional arrays? Could that be useful here?

```
char[][] tttBoard = new char[3][3];
tttBoard[1][1] = 'X';
```