Week 10: Computer Science 1

Loops, Methods, Arrays Review and Problems

Let's work through a few examples to review loops and arrays.

One of the best ways to get comfortable with these tools is to walk through the logic of a problem or even better a game.

You can break down the rules (or logic) of a game into a series of steps. Then you can convert those steps into code.

Let's create the classic game of **Rock, Paper, Scissors.**

The first step is to break down the rules of the game. What are they?

1. Two players choose either rock, paper, or scissors.
2. Rock beats scissors.
3. Scissors beats paper.
4. Paper beats rock.

Let's convert the rules into code.

First, let's encode the choices. We have three discrete options: rock, paper, and scissors. What would be the best method to encode these choices?

You may initially think it would be easier to just use the names of the choices, but it's better to encode them as integers. The computer natively works with numbers so it's easier to compare numbers than strings.

- Rock = 0
- Paper = 1
- Scissors = 2

```
int rock = 0;
int paper = 1;
int scissors = 2;
```

Now, how do we decide who wins?

1. Rock beats scissors.
2. Scissors beats paper.
3. Paper beats rock.

We can use a series of if / else if statements to determine the winner.

```
if (player1 == rock && player2 == scissors) {
    System.out.println("Player 1 wins!");
} else if (player1 == scissors && player2 == paper) {
    System.out.println("Player 1 wins!");
} else if (player1 == paper && player2 == rock) {
    System.out.println("Player 1 wins!");
} else if (player1 == player2) {
    System.out.println("It's a tie!");
} else {
    System.out.println("Player 2 wins!");
}
```

Let's convert or code into a method. Why should we use a method?

What do we need to pass into the method? What should the return type be?

```
public static void determineWinner(int player1, int player2) {
    if (player1 == rock && player2 == scissors) {
        System.out.println("Player 1 wins!");
    } else if (player1 == scissors && player2 == paper) {
        System.out.println("Player 1 wins!");
    } else if (player1 == paper && player2 == rock) {
        System.out.println("Player 1 wins!");
    } else if (player1 == player2) {
        System.out.println("It's a tie!");
    } else {
        System.out.println("Player 2 wins!");
    }
}
```

Let's leave the method as `void` for now. We can change it to return a value later. Can you think of a return type that would be useful?

We now have a method to determine the winner of a round of Rock, Paper, Scissors.

Let's think about how we can design a complete program to utilize this method for your game and think about the `main()` method.

When you are designing a program, your `main()` method should have a higher level view. It should call other methods to perform specific tasks.

The `main()` method should be easy to read and understand. It should be clear what the program is doing.

Let's create our `main()` method to test our `determineWinner` method and prompt two players to enter their choices.

```
Player 1, enter your choice (0 = rock, 1 = paper, 2 = scissors): 0
Player 2, enter your choice (0 = rock, 1 = paper, 2 = scissors): 1
Player 2 wins!
```

```java
import java.util.Scanner;

public class RockPaperScissors {

    public static void main(String[] args) {

        Scanner input = new Scanner(System.in);
        System.out.print("Player 1, enter your choice (0 = rock, 1 = paper, 2 = scissors): ");
        int player1 = input.nextInt();
        System.out.print("Player 2, enter your choice (0 = rock, 1 = paper, 2 = scissors): ");
        int player2 = input.nextInt();
        determineWinner(player1, player2);

    }

    public static void determineWinner(int player1, int player2) {
        int rock = 0;
        int paper = 1;
        int scissors = 2;
        if (player1 == rock && player2 == scissors) {
            System.out.println("Player 1 wins!");
        } else if (player1 == scissors && player2 == paper) {
            System.out.println("Player 1 wins!");
        } else if (player1 == paper && player2 == rock) {
            System.out.println("Player 1 wins!");
        } else if (player1 == player2) {
            System.out.println("It's a tie!");
        } else {
            System.out.println("Player 2 wins!");
        }
    }
}
```

Can we improve our code?

How about creating a method for the player to enter their choice?

```
public static int getPlayerChoice() {
    Scanner input = new Scanner(System.in);
    System.out.print("Enter your choice (0 = rock, 1 = paper, 2 = scissors): ");
    return input.nextInt();
}
```

Now we can use this method in our `main()` method.

```
import java.util.Scanner;

public class RockPaperScissors {

    public static void main(String[] args) {

        int player1 = getPlayerChoice("Player 1: ");
        int player2 = getPlayerChoice("Player 2: ");
        determineWinner(player1, player2);

    }

    public static void determineWinner(int player1, int player2) {
        int rock = 0;
        int paper = 1;
        int scissors = 2;
        if (player1 == rock && player2 == scissors) {
            System.out.println("Player 1 wins!");
        } else if (player1 == scissors && player2 == paper) {
            System.out.println("Player 1 wins!");
        } else if (player1 == paper && player2 == rock) {
            System.out.println("Player 1 wins!");
        } else if (player1 == player2) {
            System.out.println("It's a tie!");
        } else {
            System.out.println("Player 2 wins!");
        }
    }

    public static int getPlayerChoice(String player) {
        Scanner input = new Scanner(System.in);
        System.out.print(player + "Enter your choice (0 = rock, 1 = paper, 2 = scissors): ");
        return input.nextInt();
    }
}
```

Do you see how much easier it is to read the `main()` method now?

We can call the `getPlayerChoice()` method for each player and then call the `determineWinner()` method. All the functionality and logic is contained in the methods.

What if we wanted to have five rounds and count how many rounds each player has won?

```java
import java.util.Scanner;

public class RockPaperScissors {

public static void main(String[] args) {

        int player1Wins = 0;
        int player2Wins = 0;

        for (int i = 0; i < 5; i++) {
            int player1 = getPlayerChoice("Player 1: ");
            int player2 = getPlayerChoice("Player 2: ");
            int result = determineWinner(player1, player2);
            if(result == 1){
                player1Wins++;
            }else if(result == 2){
                player2Wins++;
            }
        }

        System.out.println("Player 1 wins: " + player1Wins);
        System.out.println("Player 2 wins: " + player2Wins);

    }

    public static int determineWinner(int player1, int player2) {
        int rock = 0;
        int paper = 1;
        int scissors = 2;
        if (player1 == rock && player2 == scissors) {
            System.out.println("Player 1 wins!");
            return 1;
        } else if (player1 == scissors && player2 == paper) {
            System.out.println("Player 1 wins!");
            return 1;
        } else if (player1 == paper && player2 == rock) {
            System.out.println("Player 1 wins!");
            return 1;
        } else if (player1 == player2) {
            System.out.println("It's a tie!");
            return 0;
        } else {
            System.out.println("Player 2 wins!");
            return 2;
        }
    }

    public static int getPlayerChoice(String player) {
        Scanner input = new Scanner(System.in);
        System.out.print(player + "Enter your choice (0 = rock, 1 = paper, 2 = scissors): ");
        return input.nextInt();
    }
}
```

Let's create a new game, Hangman.

What are the rules of hangman?

1. A word is chosen at random.
2. The player guesses a letter.
3. If the letter is in the word, the letter is revealed.
4. If the letter is not in the word, the player loses a life.
5. The player has 6 lives to guess the word.

How do we get a random number?

Remember the Math.random() method?

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

This code generates a random number between 0 and 4.

Let's use this to choose a random word from an array of words.

```
String[] words = {"apple", "banana", "cherry", "date", "elderberry"};

String word = words[(int) (Math.random() * words.length)];
```

Now let's break down the other parts of the code we need.

1. We need to create a char array to store the guessed word.
2. We need to get the player's guess.
3. We need to create a loop to check if the guessed word is correct.
4. If the guessed word is correct, we need to reveal the letter.
5. If the guessed word is incorrect, we need to decrement the number of lives.

Work out the logic in comments first.

Then create a method to play the game. You should start with two methods, your main method and a method to play the game.

I'll get you started and then you can finish the code.

```
import java.util.Scanner;

public class Hangman {

    public static void main(String[] args) {
        String[] words = {"apple", "banana", "cherry", "date", "elderberry"};
        playGame(words);
    }

    public static void playGame(String[] words) {
        String word = words[(int) (Math.random() * words.length)];
        char[] guessedWord = new char[word.length()];
        for (int i = 0; i < guessedWord.length; i++) {
            guessedWord[i] = '*';
        }

        Scanner scanner = new Scanner(System.in);
        int guesses = 0;
        while (hasStars(guessedWord) && guesses < 6) {
            System.out.println(guessedWord);
            System.out.println("Guess a letter:");
            char guess = scanner.next().charAt(0);
            boolean correctGuess = false;
            for (int i = 0; i < word.length(); i++) {
                if (word.charAt(i) == guess) {
                    guessedWord[i] = guess;
                    correctGuess = true;
                }
            }
            if (!correctGuess) {
                guesses++;
            }
        }

        if (hasStars(guessedWord)) {
            System.out.println("Sorry, you didn't guess the word in 6 attempts.");
        } else {
            System.out.println(guessedWord);
            System.out.println("Congratulations, you guessed the word!");
        }
    }

    public static boolean hasStars(char[] array) {
        for (int i = 0; i < array.length; i++) {
            if (array[i] == '*') {
                return true;
            }
        }
        return false;
    }
}
---
```