Final Project Report

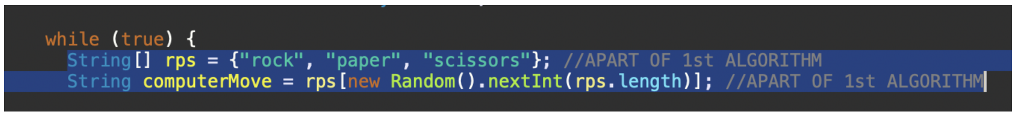
Adrian Martinez
CPSC-39

Rock Paper Scissors Shoot! Game

The code I decided to implement was a simple game of rock, paper, scissors, and shoot. This game allows you to play against the computer essentially, because each choice is randomized and you have multiple chances to either win or lose. Further down in this report I will explain the different algorithms, methods, and data structures I used to create this game.

Explaining 3 Algorithms in my code:

(1st Algorithm) - Randomized move selection



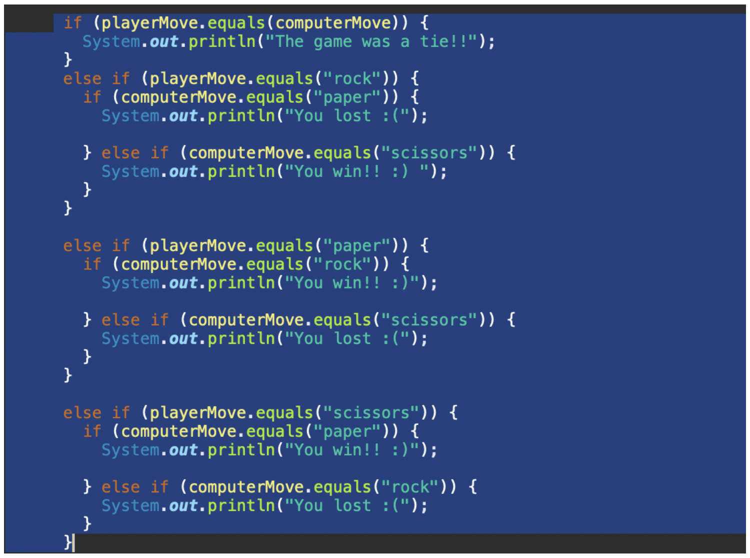
```
while (true) {
  String[] rps = {"rock", "paper", "scissors"}; //APART OF 1st ALGORITHM
  String computerMove = rps[new Random().nextInt(rps.length)]; //APART OF 1st ALGORITHM
```

The photo above displays highlighted code displaying the main source of this specific algorithm. What this code does is, it essentially randomly selects a move for the computer from the array we have of possible moves. As you can see above, there are 3 terms in parentheses, those being “rock”, “paper”, and “scissors”. The use of “Random().nextInt()” allows the code to generate a random index correlated with one of the array elements.

Big O Time: O(1). Algorithm is very short and not very complicated with only 2 lines

---

(2nd Algorithm) - Outcome of Game Decision

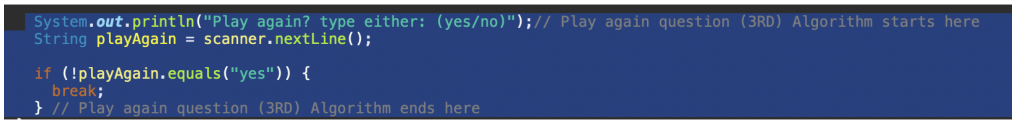


```java
if (playerMove.equals(computerMove)) {
  System.out.println("The game was a tie!!");
}
else if (playerMove.equals("rock")) {
  if (computerMove.equals("paper")) {
    System.out.println("You lost :(");

  } else if (computerMove.equals("scissors")) {
    System.out.println("You win!! :) ");
  }
}

else if (playerMove.equals("paper")) {
  if (computerMove.equals("rock")) {
    System.out.println("You win!! :)");

  } else if (computerMove.equals("scissors")) {
    System.out.println("You lost :(");
  }
}

else if (playerMove.equals("scissors")) {
  if (computerMove.equals("paper")) {
    System.out.println("You win!! :)");

  } else if (computerMove.equals("rock")) {
    System.out.println("You lost :(");
  }
}
```

The code above displays the algorithm that decides how the game ends, whether you win, lose, or tie. As we all know, the rules of this game are very simple: rock beats scissors, paper beats rock, scissors beats paper, and if you end up doing the same move as your opponent the game ends up in a tie, or you try again. Above you can see the different messages I included to match up with the decision you went with, for example, if you chose rock and the computer chose paper, you will see the message "You lost :(". As for the code itself it is very simple, with the help of the if-else statements the process of making this a reality is very clear and easy.

Big O time: O(1).

(3rd Algorithm) - Play again question?



```java
System.out.println("Play again? type either: (yes/no)");// Play again question (3RD) Algorithm starts here
String playAgain = scanner.nextLine();

if (!playAgain.equals("yes")) {
  break;
} // Play again question (3RD) Algorithm ends here
```

As you can see above in the blue highlighted section, this specific algorithm deals with the end of the game and whether it ends or goes on depending on what the player chooses. Using a simple println command I am able to ask the player whether they want to play again or not saying "yes/no". The command below initiates your decision so for example, if you type "yes" the game will start again.

Big O Time: O(1) Also very short and simple line of code, only involves 1 string comparison

Explaining the Data Structures in my code:

1. The first and most obvious data structure I used in my code was an array. I used an array because it is a simple topic that I have covered since my earliest days learning how to code. The specific array is “String[] rps = {“rock”, “paper”, “scissors”};. The array is used to store the three possible choices for the game. Furthermore, the code selects its move randomly by accessing one of the elements in this array using the code “rps[new Random().nextInt(rps.length)].”

---

2. The second data structure used in my code is a string. The purpose of the strings were to hold and compare player input, the auto generated moves made by the computer,, and the player’s decision to either replay the game or end it. For example, one string was “playAgain” and that string essentially stores the player’s response to whether they want to play or end the game.

My thought process when encountering errors:

To be quite frank, I did not encounter very huge errors when making my code. I thought about an assignment I did in one of my previous computer science courses here at merced and remember my professor showing us a lecture on him utilizing the if statements and furthermore else if statements. When creating the outcome of the game algorithm I had a little bit of trouble in organizing that specific algorithm due to the immense amount of code and at times I would either forget a letter, a semicolon, or forget an entire step in the game. You can scroll up and look at the “2nd Algorithm” and see how it still looks compact and a little confusing but I found going one by one with patience and testing every 3 lines rather than 15 helped me a lot.

What I would change or add in another version of this project:

In my mind while making this, I thought about a way to visually show the actions of the game. Frankly, I think right now that is way beyond my ability but I kept thinking about how cool it would be to see the certain hand gestures in the console area while you play. Rather than just seeing text, I think everyone would enjoy a visual of cool animations along with it.

Thank you Professor for your dedication and commitment to helping us students learn more about our future careers and dreams. All students whether they may say it or not appreciate the time you put in for us!