In this project I am building a "rock, paper, scissors" game. This is a hand game for two or more players. Participants say “rock, paper, scissors” and then simultaneously form their hands into the shape of a rock (a fist), a piece of paper (the palm of a hand, or a pair of scissors (two fingers extended). The rules are the following:

* Rock smashes scissors.
* Paper covers rock.
* Scissors cut paper.

The project will be created using python and an image model.

**Python:**

* To arbitrary generate the computer’s choices, I imported the random module:

*import random*

For simplicity, this task and the following will be stored in a function that can be called when necessary.

def get\_computer\_choice():

The computer will randomly choose one of the options:

 possible\_choices = ["rock", "paper", "scissors"]

    computer\_choice = random.choice(possible\_choices)

    print(computer\_choice)

Now, whenever necessary, I can call the function created to perform the task I need, in this case; randomly choose one of the options available:

get\_computer\_choice()

-The user also needs to make choose one of the options available, thus it will be asked to make a choice and the answer will then be stored in a variable for later use.. This task will also be stored in a function:

def get\_user\_choice():

    get\_user\_choice = input("What is your choice?: (rock,paper or scissors) ")

    print(get\_user\_choice)

get\_user\_choice()

-**Determine a winner**

Now that both players have made their choice, it must be decided who wins. This task will be stored in a function with two arguments, “computer choice” and “user choice”:

def get\_winner(computer\_choice, user\_choice):

Using an if, elif, else block we will compare the player’s choices and determine a winner. If both the user and the computer choose the same option it will be a tie.

 if computer\_choice == user\_choice:

       print("Both players have chosen {user\_choice}. It is a tie!")

By comparing the tie condition first, I got rid of quite a few cases. This way, we can know what the computer chose with only two conditional checks of computer\_choice.

* Next, I went over the possible scenarios in which the user could win and vice versa. In case the user chooses “rock”, and the computer chooses “scissors”, the user wins because rock smashes scissors. The computer’s choice is fixed “scissors”, alternatively, if the user chose “paper” it would have lost because, scissors cut paper. This is represented in the below code:

    elif user\_choice == "rock":

        if computer\_choice == "scissors":

            print("Rock smashes scissors, you won!")

        else:

            print("Paper covers rock, You lost!")

* In the next scenario the computer’s choice is “rock”. If the user chooses “paper”, he wins, since paper covers rock. If instead, he chooses scissors, he loses because rock smashes scissors.

elif user\_choice == "paper":

        if computer\_choice == "rock":

            print("Paper covers rock, You won!")

        else:

            print("Rock smashes scissors, You lost!")

* If the computer’s selection is paper and the user chooses “scissors” the user wins since scissors cuts paper. In case the user chooses rock and the computer’s choice remains “paper”, he loses because paper covers rock.

    elif user\_choice == "scissors":

        if computer\_choice == "paper":

            print("Scissors cuts paper, you win!")

        else:

            print("Paper covers rock, You lose!")

Whenever necessary, we can call the function to play the game:

get\_winner(computer\_choice, user\_choice)

**Image model:**

I this case we will be using an image model. An image model stores information about an image such as class, type, range, width, etc.. I took pictures representing each game option and divided them in classes which the computer then got trained to identify.