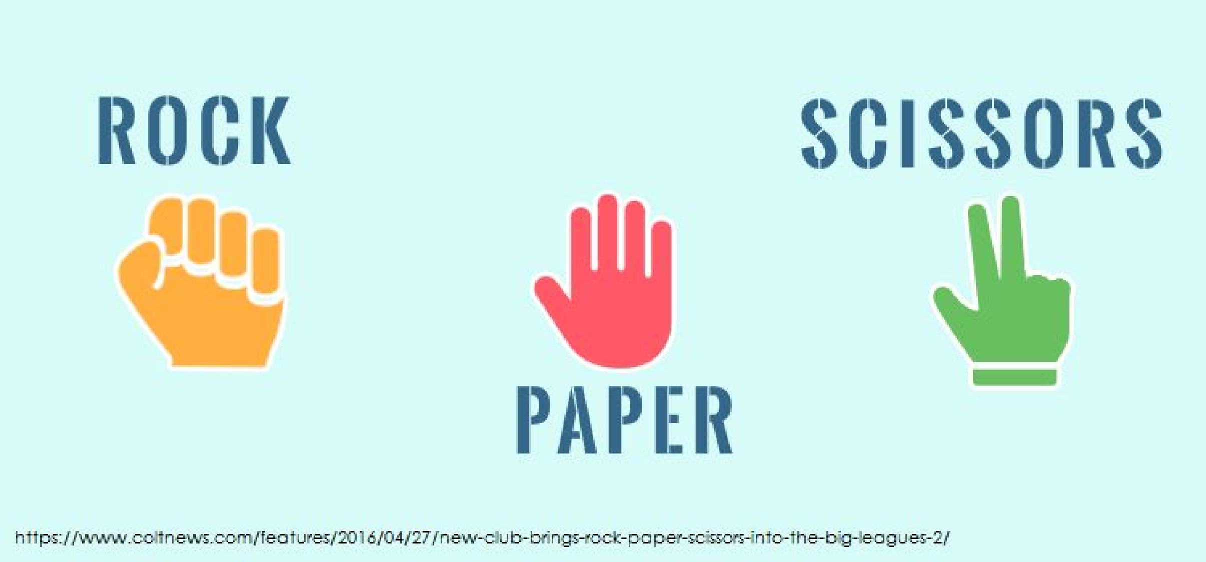
PROJECT 1 - ROCK . PAPER . SCISSORS

By Disha Bhaglal

INTRODUCTION

Rock-paper-scissors is a hand game usually played by two people, where the players simultaneously “throw” one of three hand signals that correspond to rock, paper or scissors as shown below.



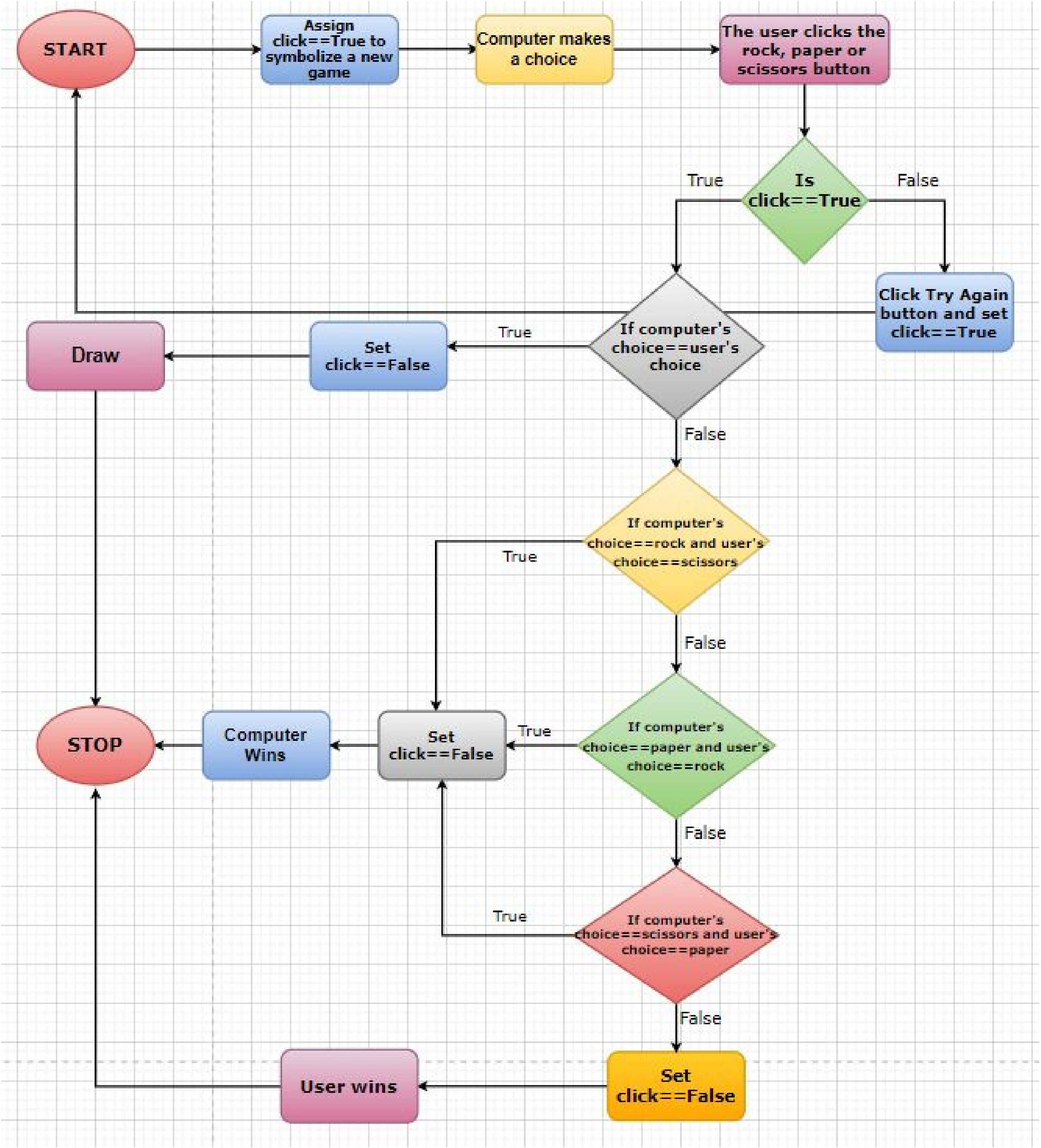
https://www.coltnews.com/features/2016/04/27/new-club-brings-rock-paper-scissors-into-the-big-leagues-2/

The "rock" beats scissors, the "scissors" beat paper and the "paper" beats rock, and if both the players throw the same shape, the game is tied. This project aims at building a computer application for this game. One of the players will be the user, who would be playing against the computer. The main rules will be the same.

TECHNOLOGY USED

We will be programming this game in Python using Tkinter for the Graphical User Interface. Tkinter is the standard GUI library for Python. Python when combined with Tkinter provides a fast and easy way to create GUI applications. Tkinter provides a powerful object-oriented interface to the Tk GUI toolkit. Tkinter also provides various controls, such as buttons, labels and text boxes used in a GUI application which we would be using in our program as well. Thus, a basic understanding of python and tkinter in a prerequisite for this project.

FLOW CHART

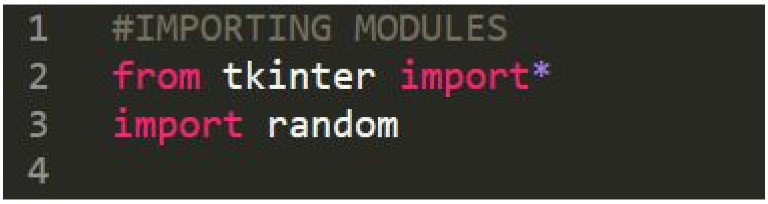

START
Assign click==True to symbolize a new game
Computer makes a choice
The user clicks the rock, paper or scissors button
Is click==True
True
False
Click Try Again button and set click==True
If computer's choice==user's choice
True
Set click==False
Draw
False
If computer's choice==rock and user's choice==scissors
True
False
If computer's choice==paper and user's choice==rock
True
Set click==False
Computer Wins
STOP
False
If computer's choice==scissors and user's choice==paper
True
False
Set click==False
User wins

CODE

The following is the link to the github repository of the code for this project: https://github.com/DishaBhaglal/Rock.Paper.Scissors.git

EXPLANATION FOR THE CODE

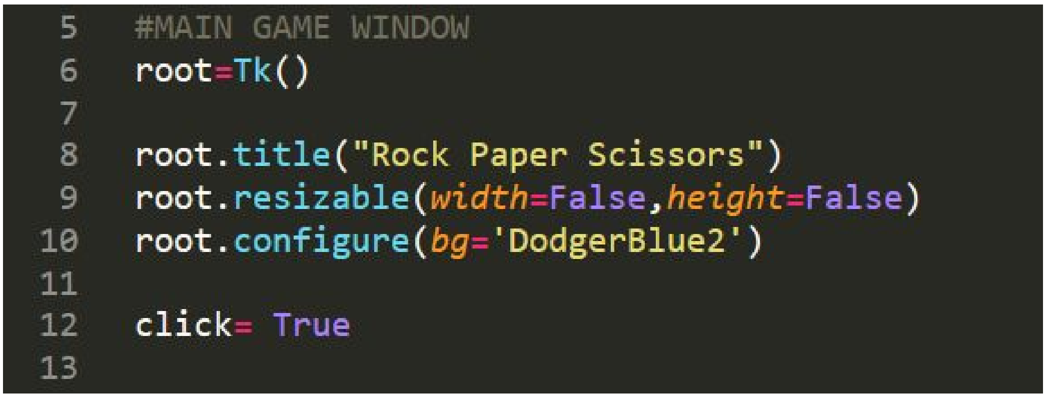
Now, let’s go through the code piece by piece.



```python
#IMPORTING MODULES
from tkinter import*
import random

```

Figure 1

In figure 1, we are importing the various modules that we will be requiring for this project. We first import the tkinter library for GUI and then the random module which will help us to choose rock,paper or scissors for the computer at random.



```python
 5   #MAIN GAME WINDOW
 6   root=Tk()
 7
 8   root.title("Rock Paper Scissors")
 9   root.resizable(width=False,height=False)
10   root.configure(bg='DodgerBlue2')
11
12   click= True
13
```

Figure 2

Here at line 5 we are creating our root() widget, which is basically the main window which will contain our game. In line 8, we are assigning a title to our window. Line 9 is making our window of a fixed size so that we can not adjust the window size and in line 10 we are filling the background colour with ‘DodgerBlue2’ colour. Then we have created a variable click and assigned it the value “True”, which will later help us to know if a new game has started. This code (with root.mainloop() in the end) will give us the below output:

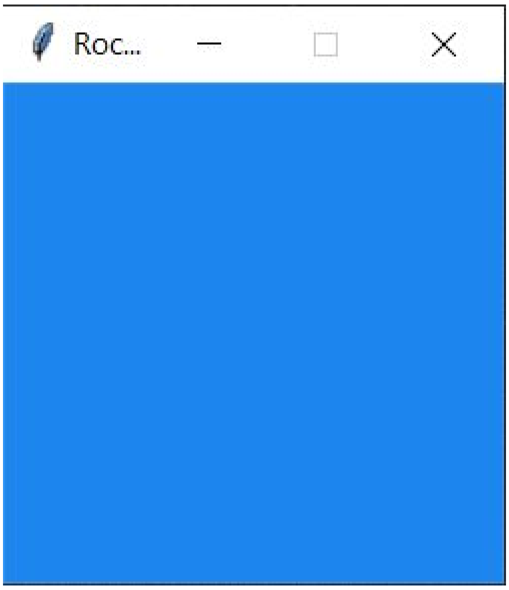
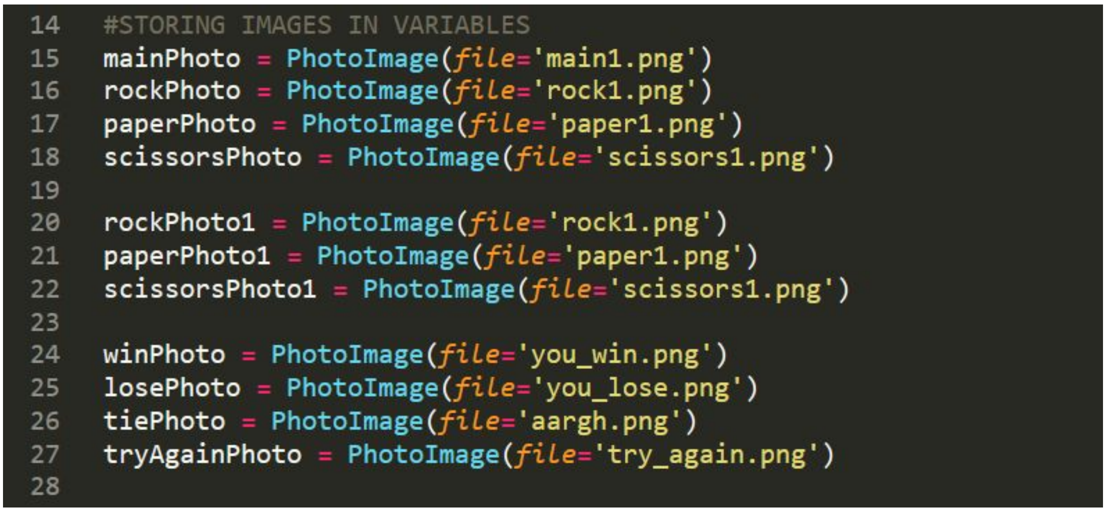


Figure 3

The full title can not be seen due to limiting the size.



```
#STORING IMAGES IN VARIABLES
mainPhoto = PhotoImage(file='main1.png')
rockPhoto = PhotoImage(file='rock1.png')
paperPhoto = PhotoImage(file='paper1.png')
scissorsPhoto = PhotoImage(file='scissors1.png')

rockPhoto1 = PhotoImage(file='rock1.png')
paperPhoto1 = PhotoImage(file='paper1.png')
scissorsPhoto1 = PhotoImage(file='scissors1.png')

winPhoto = PhotoImage(file='you_win.png')
losePhoto = PhotoImage(file='you_lose.png')
tiePhoto = PhotoImage(file='aargh.png')
tryAgainPhoto = PhotoImage(file='try_again.png')

```

Figure 4

In Figure 4, we are storing the various images that we have used in our project into variables. These photos can then be used using the variables later in the code. PhotoImage() function is being used for the purpose which takes the file to be stored as the argument.

The various images used in the project are shown below:

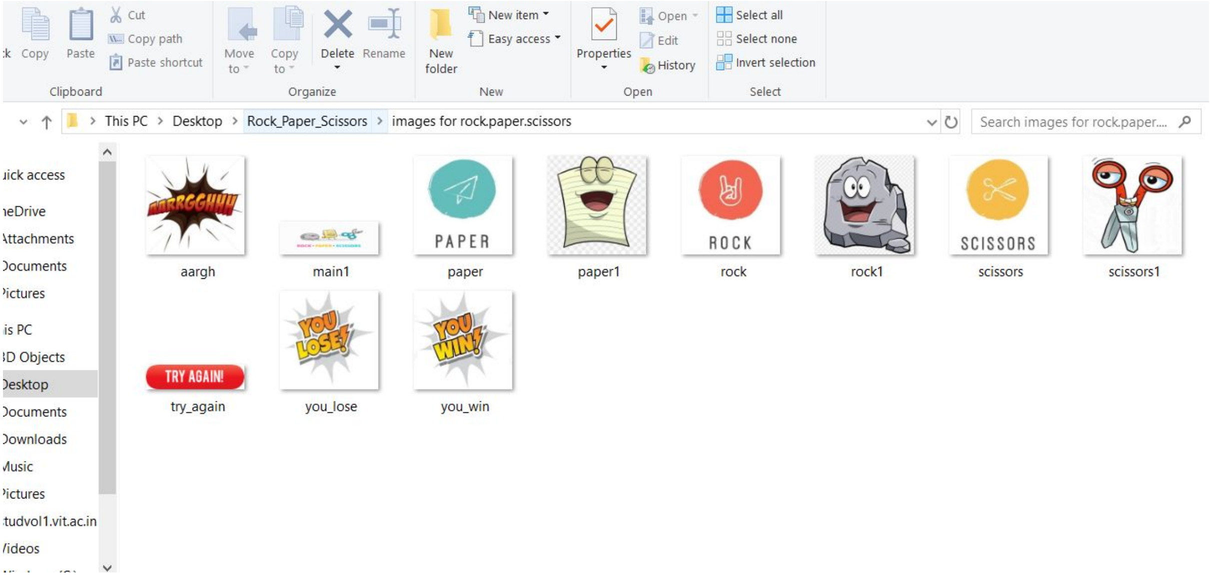
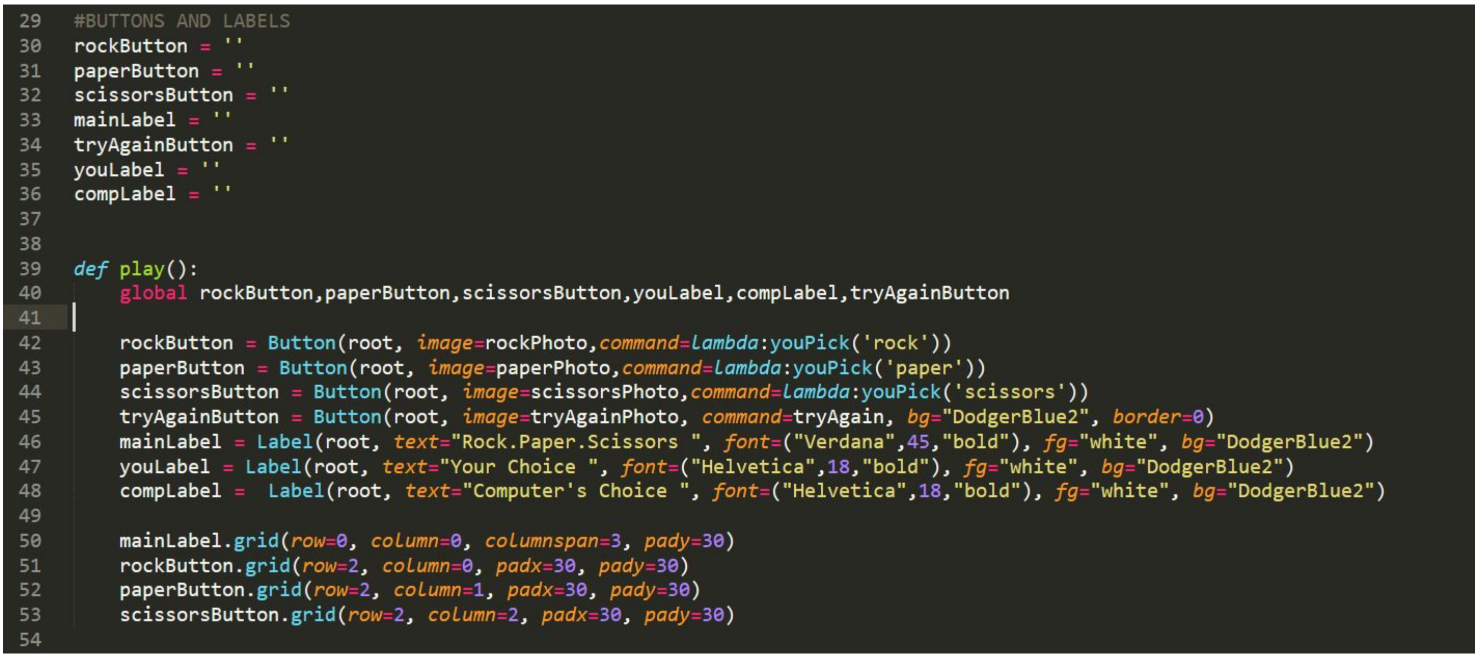


Figure 5

This is the folder containing the images.



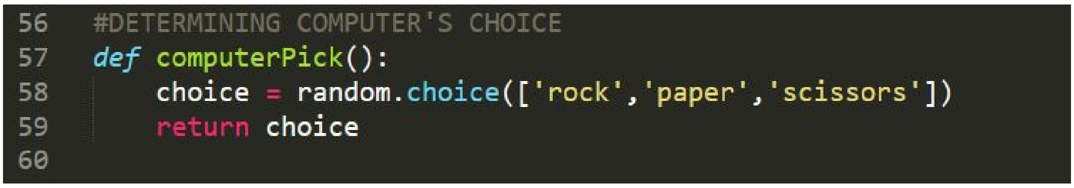
```
29  #BUTTONS AND LABELS
30  rockButton = ''
31  paperButton = ''
32  scissorsButton = ''
33  mainLabel = ''
34  tryAgainButton = ''
35  youLabel = ''
36  compLabel = ''
37
38
39  def play():
40      global rockButton,paperButton,scissorsButton,youLabel,compLabel,tryAgainButton
41
42      rockButton = Button(root, image=rockPhoto,command=lambda:youPick('rock'))
43      paperButton = Button(root, image=paperPhoto,command=lambda:youPick('paper'))
44      scissorsButton = Button(root, image=scissorsPhoto,command=lambda:youPick('scissors'))
45      tryAgainButton = Button(root, image=tryAgainPhoto, command=tryAgain, bg="DodgerBlue2", border=0)
46      mainLabel = Label(root, text="Rock.Paper.Scissors ", font=("Verdana",45,"bold"), fg="white", bg="DodgerBlue2")
47      youLabel = Label(root, text="Your Choice ", font=("Helvetica",18,"bold"), fg="white", bg="DodgerBlue2")
48      compLabel =  Label(root, text="Computer's Choice ", font=("Helvetica",18,"bold"), fg="white", bg="DodgerBlue2")
49
50      mainLabel.grid(row=0, column=0, columnspan=3, pady=30)
51      rockButton.grid(row=2, column=0, padx=30, pady=30)
52      paperButton.grid(row=2, column=1, padx=30, pady=30)
53      scissorsButton.grid(row=2, column=2, padx=30, pady=30)
54
```

Figure 6

In Figure 6, from line 30 to 36, we are creating our various buttons and labels. Buttons are used to perform some functions while labels can store text and images. We have just created them here so that they can be used as global variables in our function.

From line 39, we start defining our play() function. This function will help us give shape to our main window. We first include the buttons and labels required. Then from line 42 to 48 we define our various buttons using Button(). The first argument is root, which means we want our Buttons in the root window. Then the second argument is image which will place an image in our button. Then the third arguments i.e command, mentions the function that needs to be called when he button is clicked. For example the rockButton takes the youPick() function with ‘rock’ as the argument in its command property. This means we will call this function with a ‘rock’ argument whenever the user clicks it. For the labels, we have used the Label() function to give text to them and apply font. We have also given them background and foreground colours using bg and fg respectively.

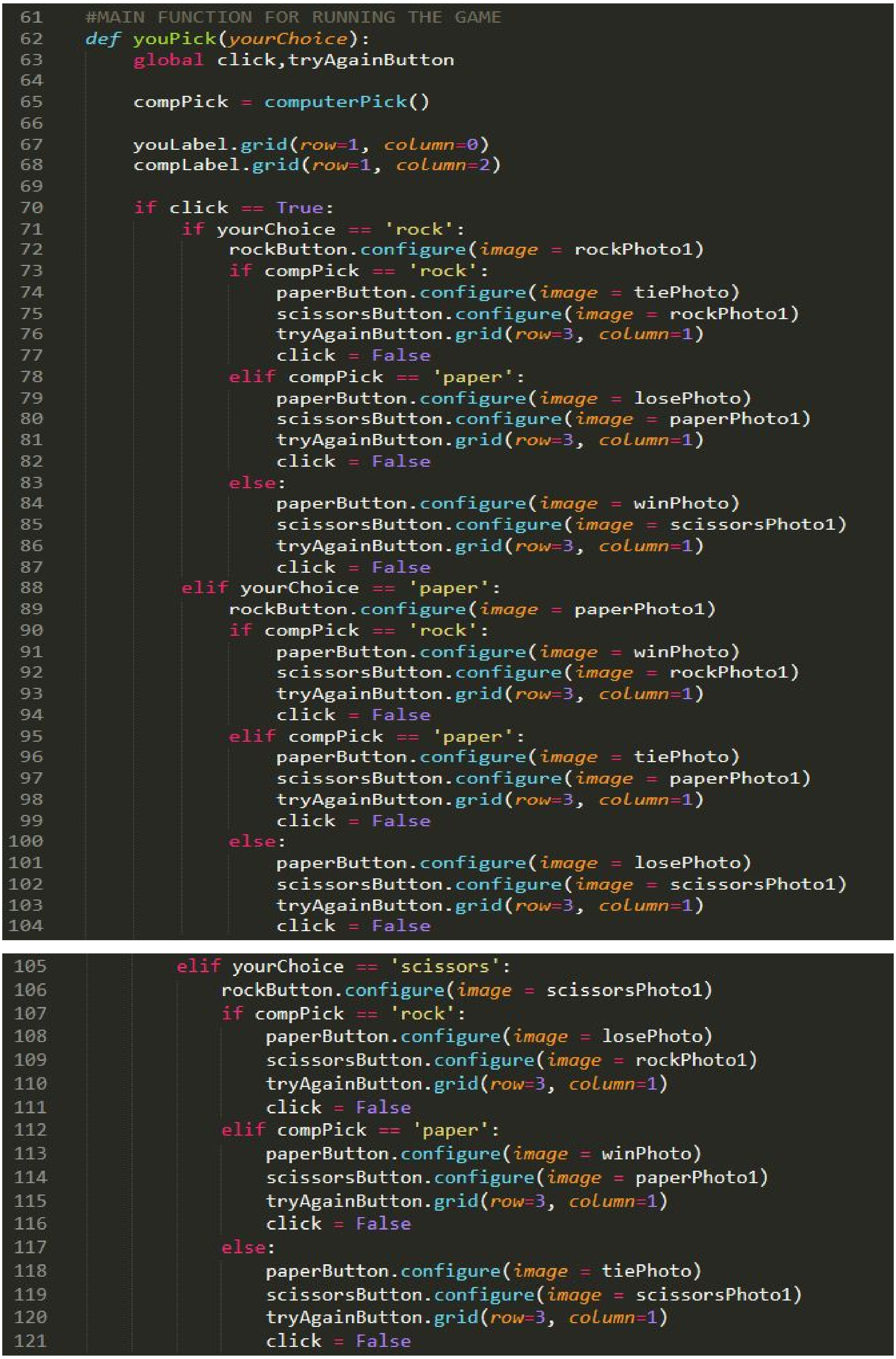
From line 50-53, we have used grid() to actually make the buttons and labels to appear on our game window. We have given the necessary row, column and columnspan values as arguments with some padding using padx and pady



```
#DETERMINING COMPUTER'S CHOICE
def computerPick():
    choice = random.choice(['rock','paper','scissors'])
    return choice

```

Figure 7

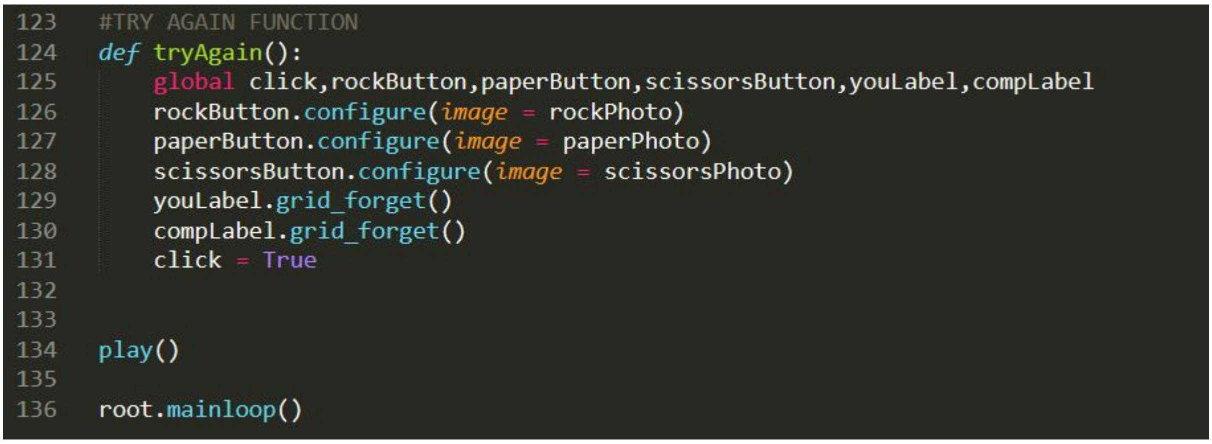
Here we have defined the computerPick() function which simply uses random.choice() on an array containing ‘rock’ , ’paper’ and ‘scissors’ to randomly choose one among them and return it. This function is making the choice for the computer.



```python
#MAIN FUNCTION FOR RUNNING THE GAME
def youPick(yourChoice):
    global click,tryAgainButton

    compPick = computerPick()

    youLabel.grid(row=1, column=0)
    compLabel.grid(row=1, column=2)

    if click == True:
        if yourChoice == 'rock':
            rockButton.configure(image = rockPhoto1)
            if compPick == 'rock':
                paperButton.configure(image = tiePhoto)
                scissorsButton.configure(image = rockPhoto1)
                tryAgainButton.grid(row=3, column=1)
                click = False
            elif compPick == 'paper':
                paperButton.configure(image = losePhoto)
                scissorsButton.configure(image = paperPhoto1)
                tryAgainButton.grid(row=3, column=1)
                click = False
            else:
                paperButton.configure(image = winPhoto)
                scissorsButton.configure(image = scissorsPhoto1)
                tryAgainButton.grid(row=3, column=1)
                click = False
        elif yourChoice == 'paper':
            rockButton.configure(image = paperPhoto1)
            if compPick == 'rock':
                paperButton.configure(image = winPhoto)
                scissorsButton.configure(image = rockPhoto1)
                tryAgainButton.grid(row=3, column=1)
                click = False
            elif compPick == 'paper':
                paperButton.configure(image = tiePhoto)
                scissorsButton.configure(image = paperPhoto1)
                tryAgainButton.grid(row=3, column=1)
                click = False
            else:
                paperButton.configure(image = losePhoto)
                scissorsButton.configure(image = scissorsPhoto1)
                tryAgainButton.grid(row=3, column=1)
                click = False
        elif yourChoice == 'scissors':
            rockButton.configure(image = scissorsPhoto1)
            if compPick == 'rock':
                paperButton.configure(image = losePhoto)
                scissorsButton.configure(image = rockPhoto1)
                tryAgainButton.grid(row=3, column=1)
                click = False
            elif compPick == 'paper':
                paperButton.configure(image = winPhoto)
                scissorsButton.configure(image = paperPhoto1)
                tryAgainButton.grid(row=3, column=1)
                click = False
            else:
                paperButton.configure(image = tiePhoto)
                scissorsButton.configure(image = scissorsPhoto1)
                tryAgainButton.grid(row=3, column=1)
                click = False
```

Figure 8

Figure 8 defines our main function, i.e the youPick() function which takes an argument yourChoice. Earlier in Figure 6 we used this function as the command for various buttons. The yourChoice variable stores our choice in it which is assigned a value once we click the rock, paper or scissors button. Now, in this function we first include the global variables to be used. We then call the computerPick function and return its value in the compPick variable. So this means compPick stores the choice of the computer. We also use youLabel and compLabel to add the text “Your choice” and “Computer’s Choice” on the window. Then we check if click==True which is used to know if we are starting a new game. If click == True, then we check if yourChoice (user’s choice) == ‘rock’ and if yes we will make the changes in the rockButton. Then we will check the value of compPick. If compPick==’rock’ then it would be tie and thus we will display the tie image. Similarly, if compPick==’paper’ then the user will lose and we will display the you lose image and if compPick==’scissors’ then the user will win and thus we will display the you win image. We will also make some other changes to the buttons. At the end of every result shown, we will change click to False, so that the user can not make a choice again until a new game is started. We will also display the Try Again button using grid to allow the user to start a new game by clicking it.
Similarly, we will check for yourChoice == ‘paper’ and ‘scissors’ as well and check the various conditions.



```
123   #TRY AGAIN FUNCTION
124   def tryAgain():
125       global click,rockButton,paperButton,scissorsButton,youLabel,compLabel
126       rockButton.configure(image = rockPhoto)
127       paperButton.configure(image = paperPhoto)
128       scissorsButton.configure(image = scissorsPhoto)
129       youLabel.grid_forget()
130       compLabel.grid_forget()
131       click = True
132
133
134   play()
135
136   root.mainloop()
```

Figure 9

In Figure 9, we define the tryAgain() function which will help the user to start a new game. We first include all required global variables. Then form line 126 to 128, we configure the images on all the buttons back to the ones they had in the start of the game. We use grid_forget() to remove some text added later in

the game from the game window. Last but not the least, we set click back to True which symbolizes that we want to start a new game.
In line 134 we call the play() function to add all the buttons and labels to the window. We end our code with root.mainloop(), a method which is executed when we want to run our application.

OUTPUTS:

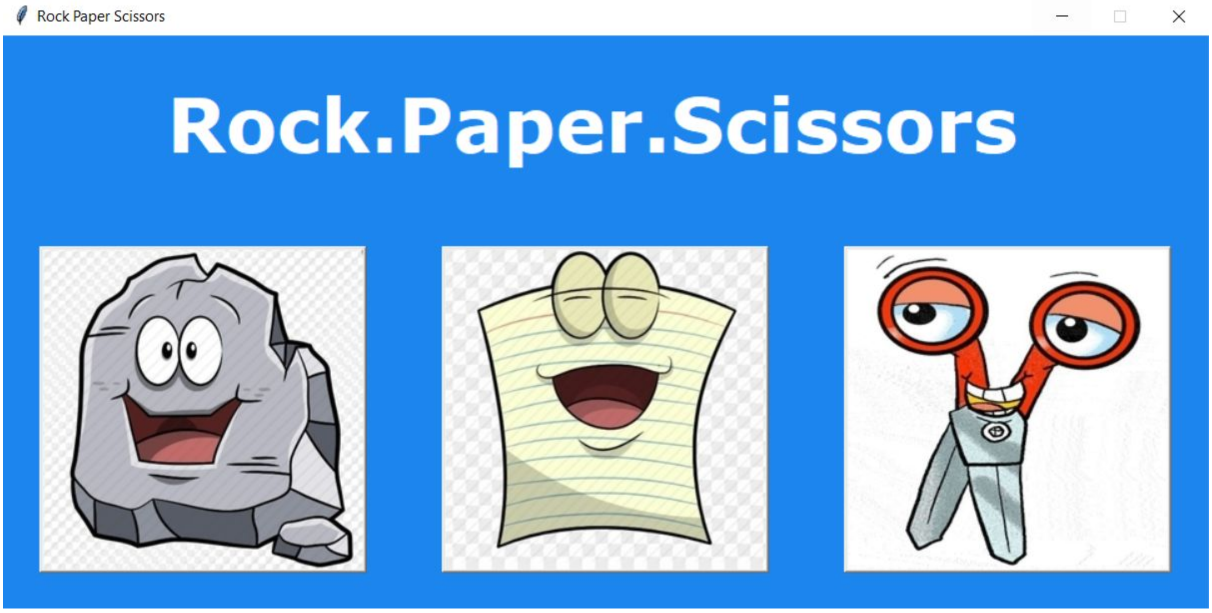


Figure 10 (Game Starts)

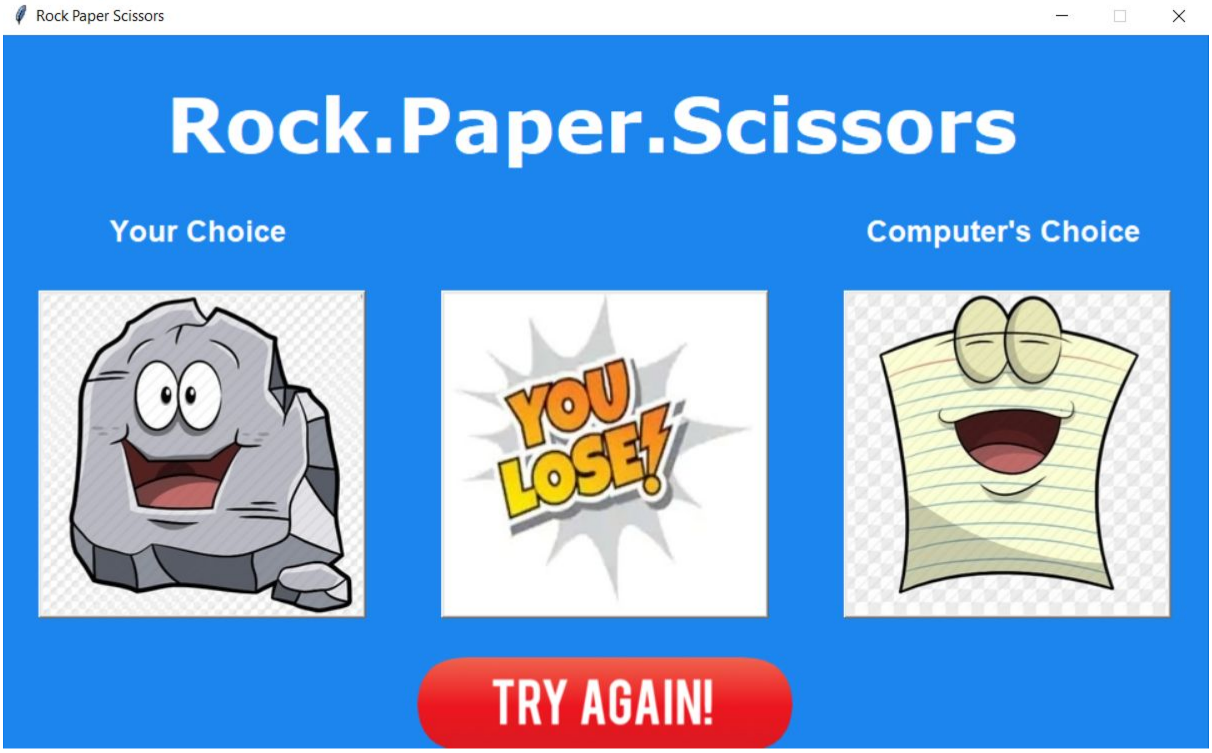


Figure 11 (User loses)

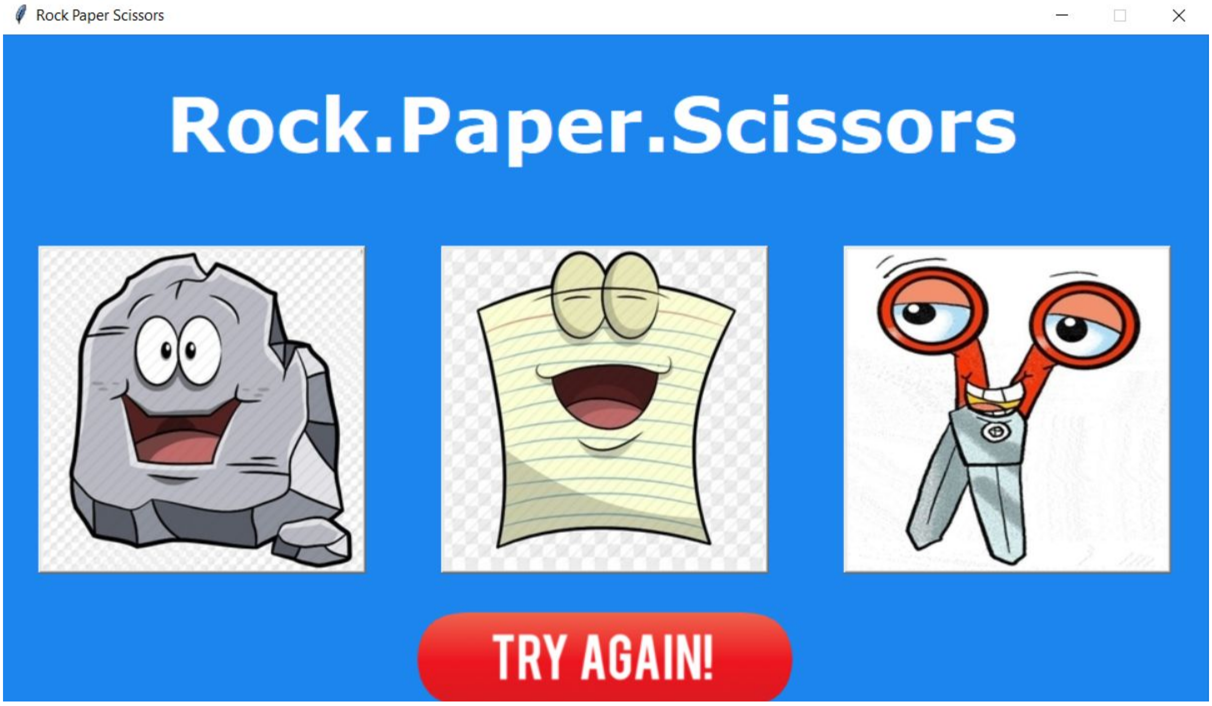


Figure 12 (Try Again button is clicked)

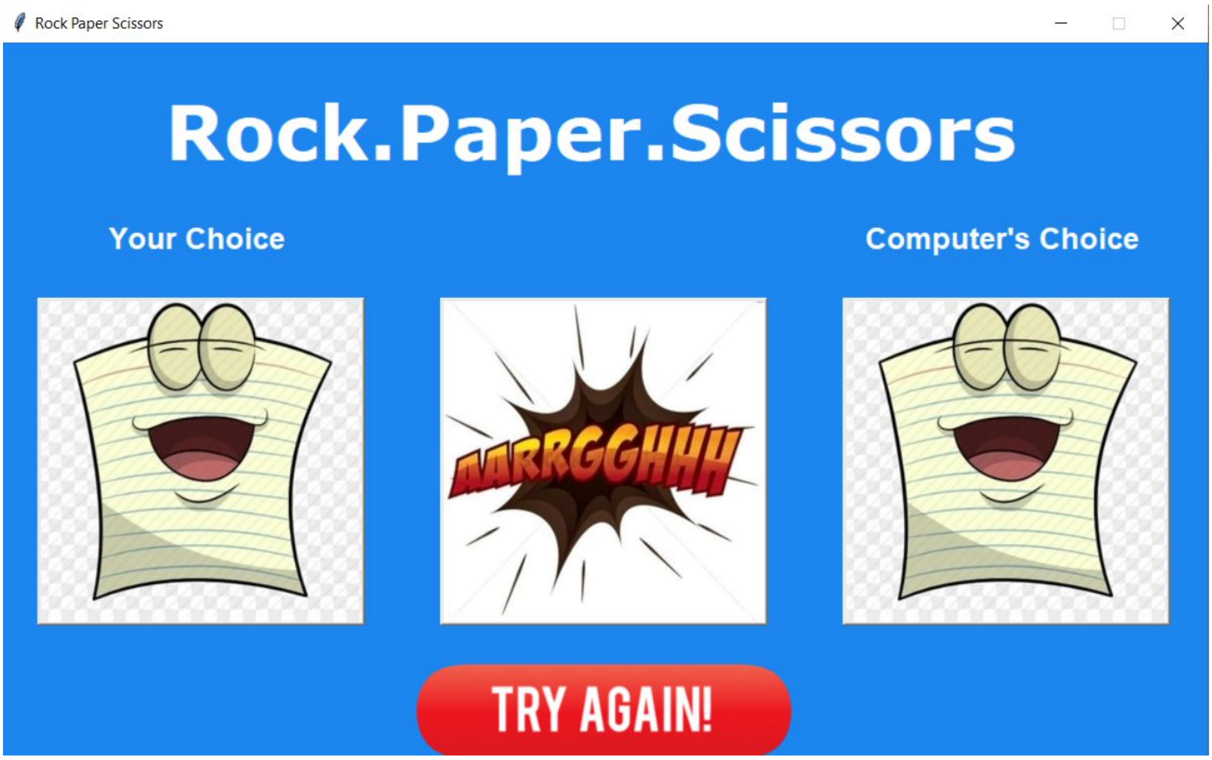


Figure 13 (It's a tie)

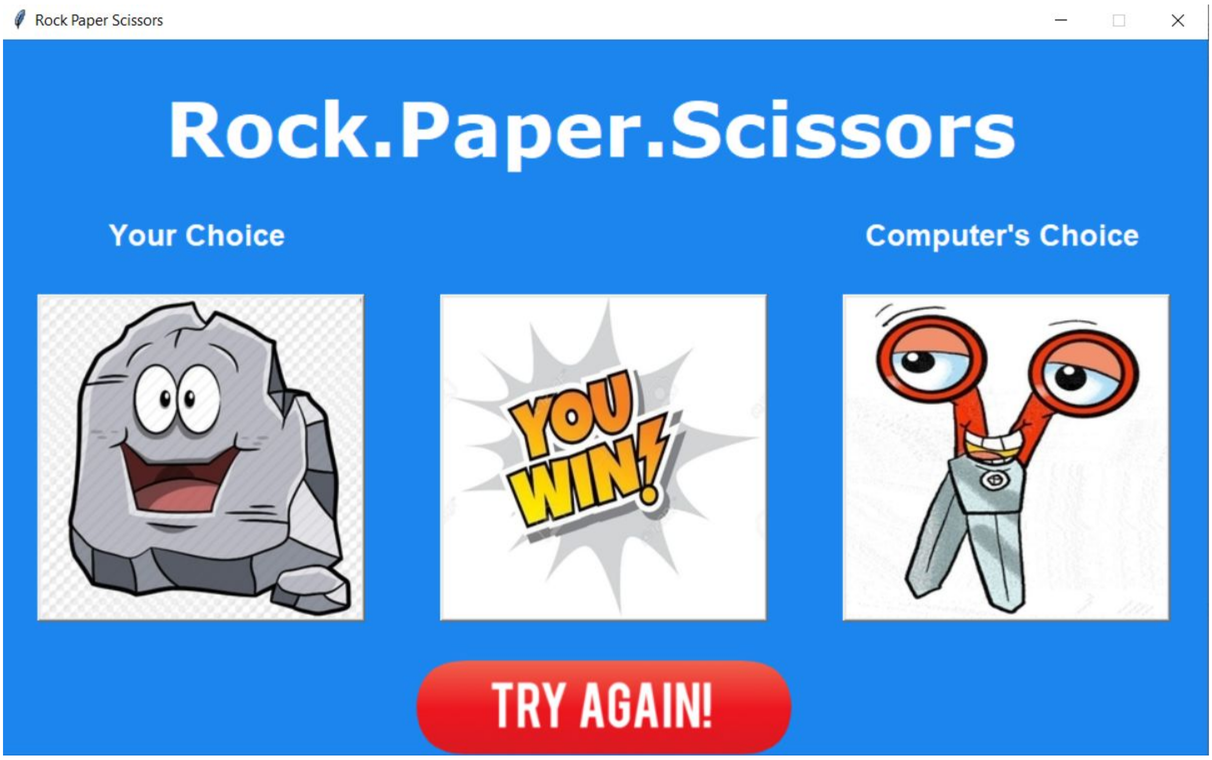


Figure 14 (User Wins)