

Thomas Herm

Pre-Engineering

CSE 120

Final Project

ROCK PAPER SCISSORS

Table of Contents:

1. Project deliverables and objectives	3
2. Projected Timeline vs Actualized Timeline	4
3. Issues Faced	5
4. Lessons Learned	6
5. Code Breakdown	7 - 11
a. SEC.py	7
b. game_flow.py	7 - 10
c. main.py	10
d. final result	11
6. Presentation link	12

Project Deliverables:

The deliverables are as follows:

Play a certain amount of RPS rounds based on user input. Player should input a case-insensitive (e.g. "rock" == "RoCk") value of rock, paper, or scissors, and the program should handle improper inputs. Computer should randomly generate a value of rock, paper, or scissors.

Rock should beat scissors, scissors should beat paper, and paper should beat rock. If the player and computer pick the same value, the round should restart.

Game should track points, and declare a winner at the end, with the option to go again.

If it's a draw, the player should have the option for a tie breaker. Game should exit when player does not want to restart game.

Project Timelines:

The timeline of the project was very different from the projected timeline. In reality, most of the code was finished shortly after receiving the project. I hit a wall when I ran into interdependency issues, and took a break to focus on other courses. I came back in early April and refactored the interdependent code into one file. This is also when the remaining bugs were squashed. On the last week of classes I created the presentation, and I used the last couple days to assemble the pdf.

Received project – 0 w

Most code complete – 1 w

Break - (rest of unaccounted weeks)

Finish code – 1 w

Make presentation – 1 w

Assemble pdf – ½ w

Issues faced:

I usually try to condense my code into smaller files, but I ran into interdependency issues doing this. I had considered putting it all in one file from the onset of this problem, but I forgot that code order did not matter in python, and so I thought the task was a lot more daunting than it actually was.

I also ran into an issue with exception types. Most error handling in my code can be done with a simple `_` case in the match statement, but inputting a non-integer for `rounds_wanted` does need `try-except`, or it'll crash. I originally tried to use `except: TypeError`, but this did not fix the problem. This is presumably because all input is string, which is the expected type, and I was trying to convert to an integer, which was expected. The problem was that the **value** entered could not be converted, so I had to use `ValueError` instead.

Lessons Learned:

1. Interdependent code should be in the same file
2. Code order does not cause issues in Python
3. `TypeError` is for *unexpected* types
4. `ValueError` is for invalid or unexpected values – including during type conversion

The first two are fairly important, as they caused a lot of headache early on that could have been easily prevented. Had I remembered that code order is largely unimportant, it would have made the prospect of moving everything to one file much less worrying.

The last couple did not give me too much trouble, but was greatly confusing. It only took me searching for different error types to fix and evaluate. It is a very helpful differentiation moving forward.

Code Breakdown:

Now I'll be breaking down the different files of code.

SEC.py:

SEC.py contains all the security related code, which in this case, is error messages. There are error messages for input errors, generation errors, and unknown errors.

```
2
3 def invalidInput(): # invalid input response 5 usages
4
5     print(f"I'm sorry, I don't understand. Please try again...\n")
6
7     #end function
8
9 def generationError(): # generated computer response out of range 3 usages
10
11     print(f"Error: computer response generation of of range\n")
12
13     #end function
14
15 def unknownError(): # default exception 2 usages
16
17     print(f"Unknown error.\n")
18
19     #end function
```

game_flow.py:

This is where most of the code is housed. It uses what I refer to as arg chains instead of pointers, to ensure that any errors or exploits only affect that reference of the value and that process, rather than the original value and any related process.

```

7 def game_round(player_answer, player_points: int = 0, cpu_points: int = 0, current_round: int = 1, rounds_wanted: int = 1): #Start function 1usage
43     case "scissors": # Scissors vs Rock
44         defeat(player_points, cpu_points, current_round, rounds_wanted)
45
46     case "paper": #on cpu answer paper
47         match player_answer:
48             case "rock": # Rock vs Paper
49                 defeat(player_points, cpu_points, current_round, rounds_wanted)
50             case "paper": # Paper vs Paper
51                 draw(player_points, cpu_points, current_round, rounds_wanted)
52             case "scissors": # Scissors vs Paper
53                 victory(player_points, cpu_points, current_round, rounds_wanted)
54
55     case "scissors": # on cpu answer scissors
56         match player_answer:
57             case "rock": # Rock vs Scissors
58                 victory(player_points, cpu_points, current_round, rounds_wanted)
59             case "paper": # Paper vs Scissors
60                 defeat(player_points, cpu_points, current_round, rounds_wanted)
61             case "scissors": # Scissors vs Scissors
62                 draw(player_points, cpu_points, current_round, rounds_wanted)
63
64     case _: # On gen error - verification escape
65         generationError()
66
67
68 def flow(current_round, rounds_wanted, player_points: int = 0, cpu_points: int = 0): # start function 5usages
69
70     isValid = False
71
72     if current_round <= rounds_wanted: # on game continue
73         while not isValid:
74
75             player_answer = input("ROCK - PAPER - SCISSORS! What do you choose? ").casefold()
76
77             match player_answer:
78                 case "rock" | "paper" | "scissors":
79                     isValid = True
80                     game_round(player_answer, player_points, cpu_points, current_round, rounds_wanted)
81                 case _:
82                     invalidInput()
83

```

134:26 CRLF UTF-8 4 spaces Python 3.13 (Code)

```

7 def game_round(player_answer, player_points: int = 0, cpu_points: int = 0, current_round: int = 1, rounds_wanted: int = 1): #Start function 1usage
43     case "scissors": # Scissors vs Rock
44         defeat(player_points, cpu_points, current_round, rounds_wanted)
45
46     case "paper": #on cpu answer paper
47         match player_answer:
48             case "rock": # Rock vs Paper
49                 defeat(player_points, cpu_points, current_round, rounds_wanted)
50             case "paper": # Paper vs Paper
51                 draw(player_points, cpu_points, current_round, rounds_wanted)
52             case "scissors": # Scissors vs Paper
53                 victory(player_points, cpu_points, current_round, rounds_wanted)
54
55     case "scissors": # on cpu answer scissors
56         match player_answer:
57             case "rock": # Rock vs Scissors
58                 victory(player_points, cpu_points, current_round, rounds_wanted)
59             case "paper": # Paper vs Scissors
60                 defeat(player_points, cpu_points, current_round, rounds_wanted)
61             case "scissors": # Scissors vs Scissors
62                 draw(player_points, cpu_points, current_round, rounds_wanted)
63
64     case _: # On gen error - verification escape
65         generationError()
66
67
68 def flow(current_round, rounds_wanted, player_points: int = 0, cpu_points: int = 0): # start function 5usages
69
70     isValid = False
71
72     if current_round <= rounds_wanted: # on game continue
73         while not isValid:
74
75             player_answer = input("ROCK - PAPER - SCISSORS! What do you choose? ").casefold()
76
77             match player_answer:
78                 case "rock" | "paper" | "scissors":
79                     isValid = True
80                     game_round(player_answer, player_points, cpu_points, current_round, rounds_wanted)
81                 case _:
82                     invalidInput()
83

```

134:26 CRLF UTF-8 4 spaces Python 3.13 (Code)


```

68 def flow(current_round, rounds_wanted, player_points: int = 0, cpu_points: int = 0): # start function 5 usages
69
70     else: # on game over
71         final(player_points, cpu_points, current_round, rounds_wanted)
72
73 def start(): # start function 3 usages
74
75     isValid = False # verification bool
76
77     current_round = 1
78
79     while not isValid: # start verification loop
80
81         try: # on proper integer input
82
83             user_input = input("Welcome to Rock Paper Scissors! How many rounds would you like to play? ")
84             rounds_wanted = int(user_input)
85             if rounds_wanted <= 0: # on integer 0 or less
86                 invalidInput()
87             else: # on integer 1 or more
88                 isValid = True
89
90         except ValueError: # on non integer input
91             invalidInput()
92
93         except: # on unknown failure
94             unknownError()
95
96     # end verification loop
97
98     flow(current_round, rounds_wanted)
99 #end function
100
101 def restart(): 1 usage
102
103     isValid = False
104
105     while not isValid:
106         player_restart = input("Do you want to play again? ").casefold()
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123

```

134:26 CRLF UTF-8 4 spaces Python 3.13 (Code)

```

117 def restart(): 1 usage
118
119     match player_restart:
120
121         case "yes":
122             isValid = True
123             print("restarting...")
124             start()
125
126         case "no":
127             isValid = True
128             print("Now closing. Have a great day!")
129             sys.exit()
130
131         case _:
132             invalidInput()
133
134
135
136
137
138
139
140 def final(player_points, cpu_points, current_round, rounds_wanted): #current round required for tie 2 usages
141     print(f"The score is {player_points} to {cpu_points}")
142     if player_points > cpu_points:
143         print("The player wins!")
144     elif player_points == cpu_points:
145         tie_response = (input("It's a draw; do you want to go for a tie breaker? ").casefold())
146
147         match tie_response:
148
149             case "yes":
150                 current_round += 1
151                 flow(current_round, rounds_wanted, player_points, cpu_points)
152
153             case "no":
154                 print("We'll leave it as a draw")
155
156             case _:
157                 print("Error code 3: Unknown response. Please enter yes or no.")
158                 final(player_points, cpu_points, current_round, rounds_wanted)
159
160     else:
161         print("The computer wins!")
162     restart()
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200

```

134:26 CRLF UTF-8 4 spaces Python 3.13 (Code)

```
140 def final(player_points, cpu_points, current_round, rounds_wanted): #current round required for tie 2 usages
141     match tie_response:
142         case "yes":
143             current_round -= 1
144             flow(current_round, rounds_wanted, player_points, cpu_points)
145         case "no":
146             print("We'll leave it as a draw")
147         case _:
148             print("Error code 3: Unknown response. Please enter yes or no.")
149             final(player_points, cpu_points, current_round, rounds_wanted)
150     else:
151         print("The computer wins!")
152         restart()
153
154 def victory(player_points, cpu_points, current_round, rounds_wanted): 3 usages
155     print("You Won this round!")
156     player_points += 1
157     current_round += 1
158     flow(current_round, rounds_wanted, player_points, cpu_points)
159
160 def defeat(player_points, cpu_points, current_round, rounds_wanted): 3 usages
161     print("You lost this round")
162     cpu_points += 1
163     current_round += 1
164     flow(current_round, rounds_wanted, player_points, cpu_points)
165
166 def draw(player_points, cpu_points, current_round, rounds_wanted): 3 usages
167     print("It's a draw; the round will restart.")
168     flow(current_round, rounds_wanted, player_points, cpu_points)
169
170
171
172
173
174
175
176
177
178
179
180
181
182
```

main.py:

Finally, main.py runs the program using the start() function from game_flow.py.

```
from game_flow import start

start()
```

Final Result:

Here is the result:

```
Python 3.8.5 Shell
Welcome to Rock Paper Scissors! How many rounds would you like to play? 0
I'm sorry, I don't understand. Please try again...

Welcome to Rock Paper Scissors! How many rounds would you like to play? g
I'm sorry, I don't understand. Please try again...

Welcome to Rock Paper Scissors! How many rounds would you like to play? 2
ROCK - PAPER - SCISSORS! What do you choose? RoCK
You said "rock", the computer said "scissors"
You Won this round!
ROCK - PAPER - SCISSORS! What do you choose?

main.py 3.8 CRLF UTF-8 4 spaces Python 3.13 (Code)

ROCK - PAPER - SCISSORS! What do you choose? rock
You said "rock", the computer said "scissors"
You Won this round!
ROCK - PAPER - SCISSORS! What do you choose? rock
You said "rock", the computer said "paper"
You lost this round
The score is 1 to 1
It's a draw; do you want to go for a tie breaker? hg
Error code 3: Unknown response. Please enter yes or no.
The score is 1 to 1
It's a draw; do you want to go for a tie breaker? yes
ROCK - PAPER - SCISSORS! What do you choose?

main.py 3.8 CRLF UTF-8 4 spaces Python 3.13 (Code)

ROCK - PAPER - SCISSORS! What do you choose? scIssorS
You said "scissors", the computer said "scissors"
It's a draw; the round will restart.
ROCK - PAPER - SCISSORS! What do you choose? PApEr
You said "paper", the computer said "rock"
You Won this round!
The score is 2 to 1
The player wins!
Do you want to play again? yes
restarting...
Welcome to Rock Paper Scissors! How many rounds would you like to play? |

main.py 3.8 CRLF UTF-8 4 spaces Python 3.13 (Code)

Python 3.8.5 Shell
Welcome to Rock Paper Scissors! How many rounds would you like to play? 1
ROCK - PAPER - SCISSORS! What do you choose? Rock
You said "rock", the computer said "scissors"
You Won this round!
The score is 1 to 0
The player wins!
Do you want to play again? no
Now closing. Have a great day!

Process finished with exit code 0

main.py 3.8 CRLF UTF-8 4 spaces Python 3.13 (Code)
```

Presentation:

