Contents

1 Basic Test Results 2

2 aaa expected autograde 3

3 aaa hint result.png 4

4 README 5

5 ex2 rpsls.py 6

6 ex2 square.py 9

1 Basic Test Results

```
Starting tests...
Wed Oct 30 12:43:14 IST 2013
4f0c988778fe98f3000c6bbf987692e9e4f22a40  -


README
ex2_square.py
ex2_rpsls.py

Testing README...
Done testing README...

Testing ex2_square.py...
result_code    Squares    50    1
Done testing ex2_square.py

Testing ex2_rpsls.py...
result_code    RPSLSgame    160    1
result_code    RPSLSillegal    correct    1
result_code    RPSLSsingleset    20    1
result_code    RPSLSsetseries    20    1
result_code    RPSLSsetreset    20    1
Done testing ex2_rpsls.py

Grading summary
---------------
****** Square:
Number of failed tests: 0
Total number of tests : 50
Penalty: 0.0
****** RPSLS Game:
Number of failed tests: 0
Total number of tests : 160
Penalty: 0.0
****** RPSLS Single set:
Number of failed tests: 0
Total number of tests : 20
Penalty: 0.0
****** RPSLS Set series:
Number of failed tests: 0
Total number of tests : 20
Penalty: 0.0
****** RPSLS Set reset:
Number of failed tests: 0
Total number of tests : 20
Penalty: 0.0
******
Expected automatic grade: 100.0
******
Submission passed!
Tests completed
```

2 aaa expected autograde

```
Grading summary
---------------
****** Square:
Number of failed tests: 0
Total number of tests : 50
Penalty: 0.0
****** RPSLS Game:
Number of failed tests: 0
Total number of tests : 160
Penalty: 0.0
****** RPSLS Single set:
Number of failed tests: 0
Total number of tests : 20
Penalty: 0.0
****** RPSLS Set series:
Number of failed tests: 0
Total number of tests : 20
Penalty: 0.0
****** RPSLS Set reset:
Number of failed tests: 0
Total number of tests : 20
Penalty: 0.0
******
Expected automatic grade: 100.0
******
Submission passed!
```

3 aaa hint result.png

4 README

```
roigreenberg
305571234
roi greenberg

==================================
=  README for ex2: Flow Control  =
==================================


usage:  1. python ex2_square.py
    2. python ex2_rpsls.py


==================
=  Description:  =
==================

In Task 1 I asked the user for number 'n' and printed a square of '#'
    in size of (2n+1)*(2n+1) with a rhombus inside with edge
    of n stars inside.
In task 2 I asked the user to choose number of rounds and then play
    Rock-Paper-Scissors-Lizard-Spock until someone win.

============================
=  List of submitted files: =
============================

README        This file
ex2_square.py    print square with a rhombus inside it
ex2_rpsls.py    playing Rock-Paper-Scissors-Lizard-Spock Game

======================
=  Special Comments  =
======================
```

You should run your programs with python3 or python 3.3

5 ex2 rpsls.py

```
1  ######################################################################
2  # FILE : ex2_rpsls.py
3  # WRITER : Roi Greenberg + roigreenberg + 305571234
4  # EXERCISE : intro2cs ex2 20013-2014
5  # DESCRIPTION: Playing Rock-Paper-Scissors-Lizard-Spock game against
6  # for set of rounds the player want until one's win.
7  ######################################################################
8
9  #!/usr/bin/env python3
10 import random
11 from ex2_rpsls_helper import get_selection
12
13 def who_win(player, comp):  # find who win
14     if player == comp:
15         return ("draw")
16     elif player == 1 and (comp == 4 or comp == 3):
17         return ("Player")
18     elif player == 2 and (comp == 1 or comp == 5):
19         return ("Player")
20     elif player == 3 and (comp == 2 or comp == 4):
21         return ("Player")
22     elif player == 4 and (comp == 5 or comp == 2):
23         return ("Player")
24     elif player == 5 and (comp == 3 or comp == 1):
25         return ("Player")
26     else:
27         return ("Computer")
28
29 def rpsls_game():  # the game function
30     player_win = comp_win = draws = 0  # reset counters
31     while -2<(player_win - comp_win)<2:
32         # player input
33         player_sel = int(input("    Please enter your selection:"\
34                                + " 1 (Rock), 2 (Paper), 3 (Scissors),"\
35                                + " 4 (Lizard) or 5 (Spock): "))
36         # make sure the number is correct
37         while player_sel < 1 or player_sel > 5:
38             # player input
39             print("    Please select one of the available options.\n")
40             player_sel = int(input("    Please enter your selection:"\
41                                + " 1 (Rock), 2 (Paper), 3 (Scissors),"\
42                                + " 4 (Lizard) or 5 (Spock): "))
43         print ("    Player has selected: "
44                +"{}.".format(get_selection(player_sel)))
45         comp_sel = random.randint(1,5)  # computer selection
46         print ("    Computer has selected: "\
47                +"{}.".format(get_selection(comp_sel)))
48         winner = who_win(player_sel, comp_sel)  # find who won
49         if winner != "draw":  # declare the result
50             print ("    The winner for this round is
51         else:
52             print ("    This round was drawn\n")
53         if winner == "Player":  # count the sum of t
54             player_win += 1
55         elif winner == "Computer":
56             comp_win += 1
57         else:
58             draws += 1
59     # find out who win and announce the winner
```

See style guidelines for format of comments describing functions.

It would be helpful to write comments explaining which case (in the game) each elif refers to.

It would be more elegant (and clear) to write abs(player_win - comp_win) < 2.

This comment describes a number of lines of code, so it should appear before them (not in the same line. The same goes for the comment in line 53.

```
60      game_winner = "Player" if player_win > comp_win else "Computer"
61      print ("The winner for this game is:", game_winner)
62      print ("Game score: Player {},".format(player_win)\
63             +" Computer {}, draws {}".format(comp_win, draws))
64      return (1 if game_winner == "Player" else -1)
65
66
67  def rpsls_play():
68      choice = 2
69      # welcoming line
70      print ("Welcome to the Rock-Scissors-Paper-Lizard-Spock game!")
71      while choice == 2:  # run until player choose to quit
72          sets = sets_won = 0
73          choice = 3
74          N = int(input("Select set length: "))
75          while choice == 3:  # run until player choose to reset
76              player_game = comp_game = 0  # reset counters
77              sets += 1
78              for ii in range(1, N+1):  # run for N rounds
79                  print ("Now beginning game", ii)
80                  game_winner = rpsls_game()  # get the round's winner
81                  if game_winner == 1:  # count the winning
82                      player_game += 1
83                  else:
84                      comp_game += 1
85                  print ("Set score: Player {},".format(player_game)\
86                         +" Computer {}".format(comp_game))
87                  # break the loop if one have won more than N/2 games
88                  if (player_game > (N/2)) or (comp_game > (N/2)):
89                      break
90              if player_game == comp_game:  # run if draw after N games
91                  num = N +1
92                  # run until one have 2 wins more than the other
93                  while -2<(player_game - comp_game)<2:
94                      print ("Now beginning game", num)
95                      num += 1
96                      game_winner = rpsls_game()  # get the round's winner
97                      if game_winner == 1:  # count the winning
98                          player_game += 1
99                      else:
100                         comp_game += 1
101                     print ("Set score: Player {}, ".format(player_game)\
102                            +"Computer {}".format(comp_game))
103             # announce the player status
104             if player_game > comp_game:
105                 print ("Congratulations! You have won "\
106                        +"in", player_game + comp_game, "games.")
107                 sets_won += 1
108             else:
109                 print("Too bad! You have lost"\
110                       +" in", player_game + comp_game, "games.")
111             # sum the set score
112             print ("You have played {} sets, and ".format(sets)
113                    + "won {}!\n".format(sets_won))
114             # ask the player how he want to continue
115             print ("Do you want to: 1 - quit, 2 - reset "
116                    +"scores or 3 - continue? ", end="")
117             while True:  # confirm the right input
118                 try:
119                     choice = int(input())
120                     break
121                 except ValueError:
122                     pass
123             if choice == 1:  # if player want to quit
124                 break
125             if choice == 2:  # if player want to reset
126                 print ("Resetting scores")
127
```

There should be a comment describing the function.

```
128 #Here to help you test your code.
129 #When debugging, it is helpful to be able to replay with the computer
130 # repeating the same choices.
131 if __name__=="__main__": #If we are the main script, and not imported
132     from sys import argv
133     try:
134         random.seed(argv[1]) # as a string is good enough
135     except:
136         pass
137 
138     rpsls_play()
```

6 ex2 square.py

```
1  ######################################################################
2  # FILE : ex2_square.py
3  # WRITER : Roi Greenberg + roigreenberg + 305571234
4  # EXERCISE : intro2cs ex2 20013-2014
5  # DESCRIPTION: Print square of '#' with  rhombus of '*' inside it
6  ######################################################################
7
8  #!/usr/bin/env python3
9  def square_printing(n):
10     number = n
11     index = 1
12     for i in range(1, 2*n+2):  # for all the lines
13         if i==1 or i==2*n+1:  # check for the first and last lines
14             print ("#" * (2 * n + 1))
15         else:
16             print ("#", end="")
17             for k in range(1, 2*n):  # for every line
18                 if k == number or k == (2*n - number):
19                     print ("*", end="")
20                 else:
21                     print (" ", end="")
22             print ("#", end="")
23             number -= index  # change the indicator for '*'
24             if number == 1:
25                 index = -1
26             print ("")  # start in new line
27
28 #Here to help you test your code.
29 if __name__=="__main__":  #If we are the main script, and not imported
30     from sys import argv
31     try:
32         n = int(argv[1])
33     except:
34         n = int(input("Please enter a positive integer: "))
35     square_printing(n)
```