TUTOR NOTES

Girls' Programming Network

Scissors Paper Rock!

Part 1: Welcome Message

```
# <student's name>

welcome_message = """
-----------------------------------------------------
Welcome to Human vs. Computer in Scissors, Paper, Rock!
-----------------------------------------------------
Moves: choose scissors, paper or rock by typing in your
selection.
Rules: scissors cuts paper, paper covers rock and rock crushes
scissors.
Good luck!
------------
"""

# Task 1.1
print(welcome_message)
```

2. Who played what?

```
# Task 2.1
class Player:
    def __init__(self, name):
        self.name = name
        self.move = None

print(welcome_message)

computer = Player("Computer")
human = Player("Human")
```

```
# Task 2.2
class Player:
    def __init__(self, name):
        self.name = name
        self.move = None

    def set_move(self, move):
        self.move = move

print(welcome_message)

computer = Player("Computer")
computer.set_move("rock")

human = Player("Human")
```

```
# Task 2.3
print(welcome_message)

computer = Player("Computer")
computer.set_move("rock")

human = Player("Human")
human.set_move(input("What is your move? scissors, paper or rock?
"))
```

```
# Task 2.4
print(welcome_message)

computer = Player("Computer")
computer.set_move("rock")

human = Player("Human")
human.set_move(input("What is your move? scissors, paper or rock?
"))

print(computer.name, "played:", computer.move)
print(human.name, "played:", human.move)
```

```
# Bonus 2.5
print(welcome_message)

computer = Player("Computer")
computer.set_move("rock")

human = Player(input("What is your name? "))
human.set_move(input("What is your move? scissors, paper or rock?
"))

print(computer.name, "played:", computer.move)
print(human.name, "played:", human.move)
```

3. Win, lose or tie?

```
# Task 3.2
class Game:
    # Student can use whatever variable names they like
    def __init__(self, player1, player2):
        self.player1 = player1
        self.player2 = player2
```

```
# Task 3.3
class Game:
    win_moves = {
        "paper": "rock",
        "scissors": "paper",
        "rock": "scissors"}

    def __init__(self, player1, player2):
        self.player1 = player1
        self.player2 = player2
```

```
# Task 3.4
class Game:
    win_moves = {
        "paper": "rock",
        "scissors": "paper",
        "rock": "scissors"}

    def __init__(self, player1, player2):
        self.player1 = player1
        self.player2 = player2

    def get_match_winner(self):
        if self.player1.move == self.player2.move:
            return None
        if Game.win_moves[self.player1.move] == self.player2.move:
            return self.player1
        else:
            return self.player2
```

```
# Task 3.5
print(welcome_message)

human = Player(input("What is your name? "))
human.set_move(input("What is your move? scissors, paper or rock?
"))

computer = Player("Computer")
computer.set_move("rock")

print(computer.name, "played:", computer.move)
print(human.name, "played:", human.move)

game = Game(human, computer)
winner = game.get_match_winner()
if not winner:
    print("It's a tie!\n")
else:
    print(winner.name, "won the match\n")
```

```
# Bonus 3.6
human.set_move(input("What is your move? scissors, paper or rock?
").lower())
```

4. Smarter Computer

```
# Task 4.1
# <student's name>

import random

welcome_message = """
...
```

```
# Task 4.2
print(welcome_message)

computer = Player("Computer")
computer.set_move(random.choice(["scissors", "paper", "rock"]))

human = Player(input("What is your name? "))
human.set_move(input("What is your move? scissors, paper or rock?
"))

print(computer.name, "played:", computer.move)
print(human.name, "played:", human.move)

game = Game(human, computer)
winner = game.get_match_winner()
if not winner:
    print("It's a tie!\n")
else:
    print(winner.name, "won the match\n")
```

5. Again, Again, Again!

```
# Task 5.1
print(welcome_message)

number_matches = int(input("How many matches would you like to
play? "))
...
```

```
# Task 5.2
print(welcome_message)

number_matches = int(input("How many matches would you like to
play? "))

# These lines haven't changed and stay outside the loop
computer = Player("Computer")
human = Player(input("What is your name? "))
game = Game(human, computer)

for i in range(number_matches):
    # These lines haven't changed either but move inside the loop
    computer.set_move(random.choice(["scissors", "paper",
"rock"]))
    human.set_move(input("What is your move? scissors, paper or
rock? "))

    print(computer.name, "played:", computer.move)
    print(human.name, "played:", human.move)

    winner = game.get_match_winner()
    if not winner:
        print("It's a tie!\n")
    else:
        print(winner.name, "won the match\n")
```

```
# Task 5.3
for i in range(number_matches):
    # The indented loop code goes here...

print("GAME OVER!")
```

```
# Task 5.4
print(welcome_message)

while True:
    try:
        number_matches = int(input("How many matches would you like
to play? "))
        break
    except ValueError:
        print("Oops! That was no valid number. Try again...")
```

6. Keeping Score!

```
# Task 6.1
class Player:
    def __init__(self, name):
        self.name = name
        self.move = None
        self.score = 0

    def set_move(self, move):
        self.move = move
```

```
# Task 6.2
class Player:
    def __init__(self, name):
        self.name = name
        self.move = None
        self.score = 0

    def set_move(self, move):
        self.move = move

    def add_win(self):
        self.score += 1

# The rest of the game setup code goes here...

for i in range(number_matches):
    # The rest of the loop code goes here...

    winner = game.get_match_winner()
    if not winner:
        print("It's a tie!\n")
    else:
        winner.add_win()
        print(winner.name, "won the match\n")
```

```
# Task 6.3
class Game:
    win_moves = {
        "paper": "rock",
        "scissors": "paper",
        "rock": "scissors"}
```

```
    def __init__(self, player1, player2):
        self.player1 = player1
        self.player2 = player2

    def get_match_winner(self):
        if self.player1.move == self.player2.move:
            return None
        if Game.win_moves[self.player1.move] ==
self.player2.move:
            return self.player1
        else:
            return self.player2

    def get_game_winner(self):
        if self.player1.score > self.player2.score:
            return self.player1
        elif self.player1.score < self.player2.score:
            return self.player2
        else:
            return None
```

```
# Task 6.4
for i in range(number_matches):
    # The loop code goes here…

print("------------------------------------------------------")
print("GAME OVER!")
print(human.name, "won", human.score, "matches")
print(computer.name, "won", computer.score, "matches")

winner = game.get_game_winner()
print(winner.name if winner else "No one", "is the winner!!")
print("------------------------------------------------------")
```

7. That's not a real move!

Task 7.1: Check the move is valid!

Create a `while` loop that runs until the user enters a valid move of "scissors", "paper" or "rock".

If the move isn't valid, ask the user for their move again!

★ CHALLENGE 7.2: Game Over! Shut Down! ★

Sometime the user might say they want to play a certain number of rounds, but has to leave before the rounds are finished.

Create an `if` statement that checks to see if the user entered "quit" as their move, and close the game down.

Don't forget to tell the user who the overall winner was!

7.1

```
class Player:
    def __init__(self, name):
        self.name = name
        self.move = None
        self.score = 0

    def add_win(self):
        self.score += 1

    def print_move(self):
        print(self.name, "played:", Game.translator[self.move])

    def inputGame(self):
        user_hand = input( "Please input your hand (R, P, or S) (or Quit): ")
        user_hand = user_hand[:1].upper()
        while user_hand not in ['Q','R','P','S']:
            user_hand = input( "Please input your hand (R, P, or S) (or Quit): ")
            user_hand = user_hand[:1].upper()
        self.move = user_hand

    def print_score(self):
        print(self.name, "won", self.score, "matches")
```

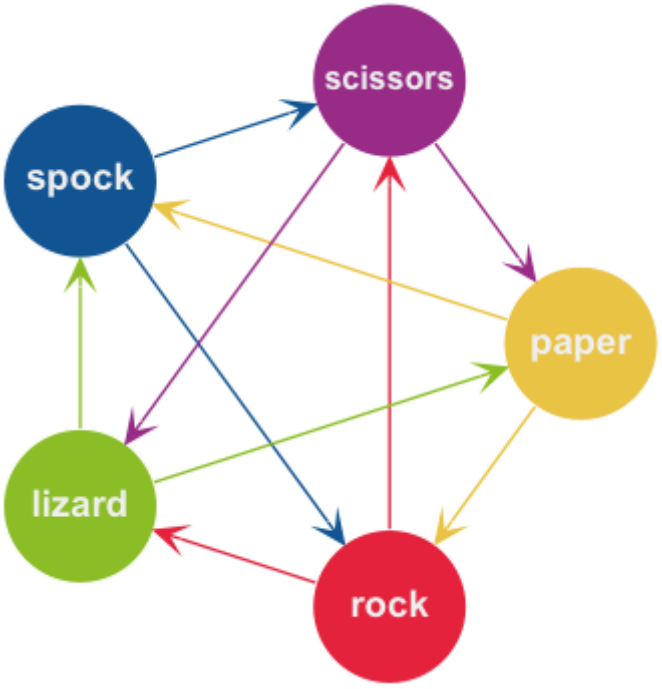
7.2

```
if __name__=="__main__":

    # Task 1.1
    print(welcome_message)

    human = Player(input("What is your name? "))
    print("Hello", human.name, "\n")

    computer = Player("Computer")

    game = Game(human, computer)



    while True:
        try:
            target_score = int(input("How many matches should win the game? "))
```

```
        break
    except ValueError:
        print("Oops! That was not a value number. Try again...")
print()

while human.score < target_score and computer.score < target_score:

    human.inputGame()
    print(human.move)

    if human.move == "Q":
        print(human.name, "quits the game.\n")
        break
```

8. Extension: Scissors, Paper, Rock, Lizard, Spock!

Let's add some more moves and play Scissors, Paper, Rock, Lizard, Spock! Follow the arrows in the picture to see who wins!



Task 8.1 Updated moves!

When you ask the user what move they want to play, include lizard and spock!

Make sure you give the computer the same options!

Task 8.2 Updated combos!

Update the moves dictionary to include all of the new combinations!

```
def inputGame(self):
        user_hand = input( "Please input your hand (Rock, Paper, Scissors, Lizard, or spocK)
(or Quit): ")
        if user_hand[-1].upper()=="K":
            user_hand = "K"
        else:
            user_hand = user_hand[:1].upper()
        while user_hand not in Game.move_choices:
            user_hand = input( "Please input your hand (Rock, Paper, Scissors, Lizard, or
spocK) (or Quit): ")
            if user_hand[-1].upper()=="K":
                user_hand = "K"
            else:
                user_hand = user_hand[:1].upper()
        self.move = user_hand


class Game:
    move_choices = ["S","P","R","L","K"]
    win_moves = {"P":["R","K"],
                 "S":["P","L"],
                 "R":["S","L"],
                 "L":["K","P"],
                 "K":["S","R"]}

    translator= {"P":"Paper","S":"Scissors","R":"Rock","L":"Lizard","K":"Spock"}

def get_match_winner(self):
        if self.player1.move == self.player2.move:
            return None
        if self.player2.move in self.win_moves[self.player1.move]:
            return self.player1
        else:
            return self.player2
```