Girls' Programming Network

Scissors Paper Rock!

Extensions!

Congratulations!

You finished your Scissors Paper Rock game!

Here's a few extensions you can do to make your game even better!

Try one (or more!) of the extensions in this book!

How Many Games?
Keeping Score
That's not a real move
Scissors Paper Rock Lizard Spock

Where do I start?

You can do these in any order, so choose your favourites!

6. Extension: How Many Games?

Instead of running infinite times, we now want to run as many times as the user wants!

For Loops

Task 6.1: How many games?

Find out how many games the user wants to play at the start of the game! Store this in a variable called **`number_of_games`**. Put this after your welcome message!

Hint

Input returns a **string**. Make sure you **convert it to an int** and store it in a variable!

`int("57")` will give you back 57. You can use `int(...)` on a variable too!

Task 6.2: Loop time!

REMOVE the while loop you made in section 5. Instead, create a for loop that runs as many times as the user asked for!

You'll need to use:

- A `for` loop
- `range(number_of_games)`

Use this line after you have asked how many games they want to play:

```
for i in range(number_games):
```

Task 6.3: Indenting your code

Things we want to do every game need to be indented inside the loop.
Make sure all your code is still indented in the FOR loop like it was for the WHILE loop.
We want to ask for a move and check the winner every round!

Hint

Indented lines have a tab at the start like this, they look this:

```
for blah in something:
    THIS IS INDENTED
```

Task 6.4: GAME OVER!

After all the rounds are played, print out "GAME OVER!".
Make sure this is after your loop and doesn't print every round!

7. Extension: Keeping Score!

Do "Extension 6: How Many Games" first, then you can do this extension
Why play lots of games if we're not even keeping count of who wins?? Let's keep score!

Task 7.1: Counter!

Before your loop, create 2 variables, these are going to be your human and computer counters. Start by setting them both to 0.
These will keep track of the human and computer scores throughout the game!

Task 7.2: Add 1!

Every time the computer or human wins we need to add one to the appropriate counter If it's a tie neither player gets a point!

Hint

You'll need to add to a counter inside your if/elif statements whenever someone wins!

To add 5 to a score_count you would use:

```
score_count = score_count+5
```

Task 7.3: And the winner is!

After all the games are played we need to report the overall winner.

Print out how many games the human can computer won each and the overall winner!

```
---------------------------------
GAME OVER!
Human won 5 games
Computer won 2 games
Human is the winner!!
---------------------------------
```

Hint

Use an `if` statement to compare the scores to calculate the overall winner!

★ CHALLENGE 7.4: First to X

Right now we play a set number of games. But can you figure out how you could change your program to keep playing until a player gets a certain number of points?

You might need to use a while loop, or a break, or something else you can think of!

8. Extension: That's not a real move!

What happens if the human plays a wrong move, like Batman? Or what happens if the human doesn't write their move in lowercase letters and plays ROCK, Rock or ROcK? Test your code and find out!

There are a few little issues with our code so far:

- If the human inputs an incorrect move the program doesn't notice!
- If the human inputs a move which is not written in lowercase letters, they will lose the game. (Unless you already did the bonus task 3.3!)

We need to make our code more robust! Let's see what we can do to fix these issues!

Task 8.1: 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!

Task 8.2: ROCK Rock rOcK!

To compare the human's move to the list moves, the strings need to look the same.

We can use `word = word.lower()` to change what the user entered to lower case. Update your code so we're always using the lowercase version of what your user entered when comparing the strings!

Notice the dot, this is important!

★ CHALLENGE 8.3: Game Over! Shut Down! ★

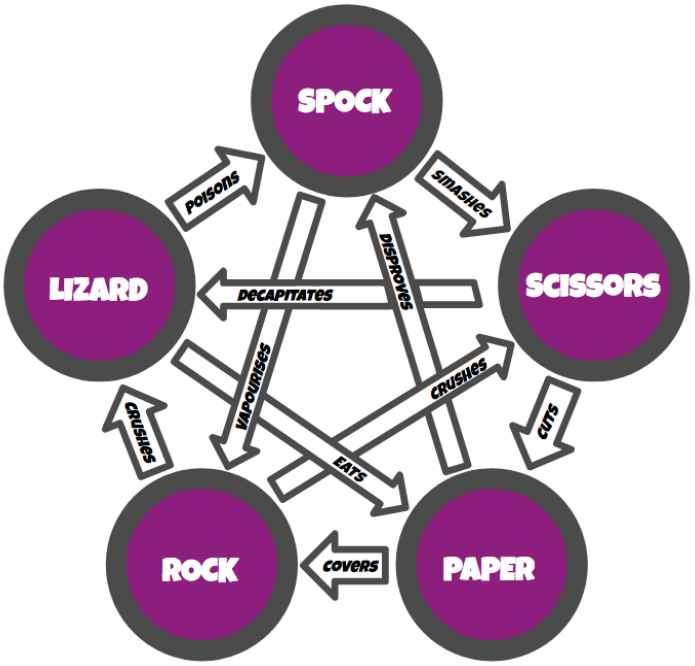
Sometimes the user might say they want to play a certain number of rounds, but has to leave before the rounds are finished.

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

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

9. 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 9.1 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 9.2 Updated combos!

Add more elif statements to make all your extra options possible!
(There's a table on the back of this sheet you can use to figure out the options)

★ CHALLENGE 9.3: Too many elifs!!

Woah, that dictionary got big! It's got 25 combinations. But what if we dealt with all the ties in one if statement!

Use a single if statement to find all the ties, by comparing the human computer moves!

Human Move	Computer Move	Who Wins?
scissors	scissors	
scissors	paper	
scissors	rock	
scissors	lizard	
scissors	spock	
paper	scissors	
paper	paper	
paper	rock	
paper	lizard	
paper	spock	
rock	scissors	
rock	paper	
rock	rock	
rock	lizard	
rock	spock	
lizard	scissors	
lizard	paper	
lizard	rock	
lizard	lizard	
lizard	spock	
spock	scissors	
spock	paper	
spock	rock	
spock	lizard	
spock	spock