Girls' Programming Network

Scissors Paper Rock!

This project was created by GPN Australia for GPN sites all around Australia!

This workbook and related materials were created by tutors at:

Sydney, Canberra and Perth

Girls' Programming Network

If you see any of the following tutors don't forget to thank them!!

Writers

Sarah Mac
Renee Noble
Vivian Dang
Courtney Ross

Testers

Catherine Murdoch
Maddie Jones
Sheree Pudney

A massive thanks to our sponsors for supporting us!

ATLASSIAN

Part 0: Setting up

Intro to Python

Task 0.1: Making a python file

1. Go to https://replit.com/
2. Sign up or log in
   (we recommend signing in with Google if you have a Google account)

Task 0.2: Making a python file

1. **Create** a new project
2. Select **Python** for the template
3. Name your project **scissors_paper_rock**

Task 0.3: You've got a blank space, so write your name!

A main.py file will have been created for you!

1. At the top of the file use a comment to write your name!

   Any line starting with # is a comment.

   ```
   # This is a comment
   ```

2. Run your code using the ▶️ **Run** button. It **won't** do anything yet!

☑ CHECKPOINT ☑

If you can tick all of these off you can go to Part 1:

- ☐ You should have a file called main.py
- ☐ Your file has your name at the top in a comment
- ☐ Run your file and it does nothing!

Part 1: Welcome Message

Task 1.1: Print a welcome and the rules

Welcome the player and print the rules!

Use a print to make it happen when you run your code:

```
------------------------------------------------------
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!
------------
```

Hint

Want to print multiple lines at a time? You can use three sets of quotes instead of one, to make your strings go over multiple lines

```
print("""
Print
Three
Lines
""")
```

☑ CHECKPOINT ☑

If you can tick all of these off you can go to Part 2:

- ☐ Print a welcome
- ☐ Print the rules
- ☐ Try running your code!

2. Who played what?

Task 2.1: Make the computer play the same move every time!!

Make a variable for the computer's move such as **`computer_move`**, set it to `"scissors"`, `"paper"` or `"rock"`.

Task 2.2: Ask the human for their move

Use `input` to ask the human for their move and save their answer in a variable, name it something like **`human_move`**.

It should look like this when you run your code:

```
------------------------------------------------------
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!
------------

What is your move? scissors, paper or rock?
```

Task 2.3: Print out the moves

Print out the moves the computer and the human have played.

It should look like this when you run your code:

```
------------------------------------------------------
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!
------------

What is your move? scissors, paper or rock? scissors

Computer Played: paper
Human Played: scissors
```

☑ CHECKPOINT ☑

If you can tick all of these off you can go to Part 3:

- ☐ Set a move for the computer
- ☐ Ask the human to type in their move and store it in a variable
- ☐ Print out the human and computers moves
- ☐ Run your code!

★ BONUS 2.4: Not so fast!!

This would look cooler if the computer paused before it said each line!

1) At the top of your file write `import time`
This will let us use what we need to use to make our program sleep for a few seconds.

2) Before any `print`, add a line that says `time.sleep(0.1)`
This will make our program 'sleep' for a tenth of a second! You can adjust it to any time you want. **Try putting sleep between your print statements!**

★ BONUS 2.5: Personalise the game

Waiting for the next lecture? Try adding this bonus feature!!

1. At the start of the game ask the human to enter their name. Store it in a variable (maybe use `player_name`)

2. Change your other code so that every time it says “Human” it prints the player’s name instead!

Remember you can add a variable to some text like this:

```
"Hello " + player_name
```

3. Win, lose or tie?

Let's figure out who won the game!

Task 3.1: What are the different ways to win, lose and tie?

What are all the combinations of how the game could go? Finish this table

Human Move	Computer Move	Who Wins?
scissors	scissors	draw
scissors	paper	human
scissors	rock	
paper		

Dictionaries

Task 3.2: Store the combinations in a dictionary

Create a new dictionary called results, it will store all the data from the table above!

In this dictionary, each key should be a tuple of the human and computer move. The corresponding value will be who wins!

Hint

Make sure the keys are created using (human_move, computer_move), just like in the table above.

```
results = {("paper", "rock"): "human"}
```

Task 3.3: Get the value using the key!

Using our dictionary, store who wins in a variable such as winner. Use the combination of **`human_move`** and **`computer_move`** to create the key to get the value!

Then `print` out the winner to the screen!

Hint

Remember we can look things up in the dictionary with a *tuple*. Tuples go inside round brackets and can be multiple items.

If we had a dictionary of holidays by month and date like this:

```
holidays = {("December", 25): "Christmas", ("April", 25): "ANZAC Day",
            ("January", 1): "New Year's Day"}
```

we might look things up like this: `holidays[(month, date)]`

`holidays[("January", 1)]` would return `"New Year's Day"`

Hint

Remember that (human_move, computer_move) is not the same as (computer_move, human_move)! Order is important.

☑ CHECKPOINT ☑

If you can tick all of these off you can go to Part 4:

- ☐ Create a dictionary containing every combination of moves
- ☐ Store who won in a variable
- ☐ Print out the winner
- ☐ Run your code and test different moves!

★ BONUS 3.4: ROCK Rock rOcK!

Waiting for the next lecture? Try adding this bonus feature!!

We see that `"Rock"` is not the same as `"rock"` and our game only works when it's all in lowercase. Make your game work when player input a move with capital letters such as `"Rock"` or `"sCissors"`.

`"FrOg".lower()` will return `"frog"`. Try use `.lower()` on your variables to make sure the human players move is converted to lowercase!

4. Winner, Winner!

It's time to tell the user who won the game!

Task 4.1: If it's a tie!

Use an `if` `elif` and `else` statements to print out the winner!

You should print different messages based on whether:

- It's a tie
- The human won
- The computer won

☑ CHECKPOINT ☑

If you can tick all of these off you can go to Part 5:

- ☐ Your if statement prints "it's a tie" if the moves are the same
- ☐ Your elif statement prints "human won the game"
- ☐ Your else statement prints "computer won the game"
- ☐ Run your code and test different moves!

★ BONUS 4.2: Name the winner!

Waiting for the next lecture? Try adding this bonus feature!!

Update your code so that instead of saying "The winner is human" refer to the human by name, using the name you collect in Bonus 2.5.

5. Smarter Computer

The computer keeps playing the same move! That's no fun! Let's make the computer choose a random move!

Random

Task 5.1: Import Random Library

To get access to cool random things we need to import random!

At the top of your file add this line:

```
import random
```

Task 5.2: Choose a random move!

Find your line of code where you set your computer move, improve this line by choosing a random move.

Use chose a random move for the computer using **`random.choice`** from a list of `"scissors"`, `"paper"` or `"rock"`.

Hint

If I wanted to choose a random food for dinner I could use code like this:

```
dinner = random.choice(["pizza", "chocolate", "nutella", "lemon"])
```

☑ CHECKPOINT ☑

If you can tick all of these off you can go to Part 6:

- ☐ The computer plays a random move every time.
- ☐ The line "Computer played: …." prints different things out!
- ☐ Try different moves against the computer, does the the correct winner print?

★ BONUS 5.3: A picture says a thousand words!

Waiting for the next lecture? Try adding this bonus feature!!

Instead of printing "The human played paper" it would be much cooler to print a picture of a paper! Use ascii art to print images for what the human and computer played!

```
Human plays paper:

 _________
|         |
|  I AM   |
|A SHEET  |
|OF PAPER |
|_________|
```

1. Go to this link: girlsprogramming.network/ascii And get the pictures for paper, scissors and rock
2. At the top of your code, store each of these ascii images as a string in different variables (maybe rock_pic, paper_pic, etc …)
3. Instead of just printing out the word the human or computer played, also print out the correct picture to match what they played. You might need to use an if statement to figure out which picture to print!

6. Again, Again, Again!

We want to play Scissors-Paper-Rock more than once! Let's add a loop to play on repeat!

For Loops

Task 6.1: How many games?

Find out how many games the user wants to play at the start of the game!
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!

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 to start your loop:
`for i in range(number_of_games):`

Task 6.3: Indenting your code

Things we want to do every game need to be indented inside the 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
```

You can indent many lines at once by highlighting them and then hitting the tab key. Make sure you highlight the whole line though!

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!

☑ CHECKPOINT ☑

If you can tick all of these off you can go to the Extensions:

- ☐ Ask the user how many games they want to play
- ☐ Your game repeats the number of times the user asked for
- ☐ GAME OVER prints once, after all of the rounds!