Girls' Programming Network

Guess Who!

Extensions

7. Extension: Which questions?

So far, we've had to ask each question every time we want to guess - even if you only needed to find out what their accessory is. Now, we want to let the human player decide which question they want to ask.

Task 7.1: Eye colour, hair colour, accessory, or name?

Once your `while` loop has started, add a question that asks them what kind of question they would like to ask (eye colour, hair colour, accessory, or name?).

Task 7.2: Question time!

You have 4 kinds of questions and if statement checks. Put each of these 4 chunks inside an if statement that means that it will only occur if this was the type of question the user wanted to ask.

Hint

Remember that `input` returns a **`string`**. Make sure that your type of `input` (string, int, etc.) matches the `if` statement!

8. Extension: How many questions?

Now, let's track how many (or how few) questions it takes you each game to guess correctly!

Task 8.1: Counter!

Before your loop create a `variable`, this will be your guess counter. Start by setting it to `0`.

Task 8.2: Add 1!

Every time the user makes a guess (a name guess or any other feature guess), add one to this counter.

Hint

You'll need to add to the counter at the beginning of your `while` loop!

Task 8.3: How many questions?

At the end of the game, `print` out how many questions the user has asked.

9. Extension: I give up!

What if you're sick of guessing, and just want to find out who it is? Let's now add to our code so that we can decide to give up and finish the game.

Task 9.1: I give up!

When we check if the user has guessed the correct name we see if it is correct, in which case they win. Otherwise we tell them to try again.

Let's let them give up, ending the game and revealing the answer.

Add an `elif` to your `if-else` statement(s) so the user can give up and find out who it was.

Hint

Create an `elif` statement to check if the user entered `"give up"` for their guess when trying to guess the name, and in this statement end the game and `print` out the answer.

★ Challenge 9.2: Cases and "

What we've written so far will only give the user the answer and quit the game if they input `"give up"` - but what if we want our game to be more robust, and understand that someone typing `"give up"` or `"Give Up"` wants the same result?

You might need to modify your `elif` statement(s), or maybe re-write how you check the user input.

★ Challenge 9.3: Even more options!

What if we want the user to be able to input even more options, like 'I QUIT', or 'WHY CANT I GUESS THIS', and have the computer know that this means the person does not want to play any more and to end the game and tell them the answer?

10. Extension: List the Info

It's hard to remember everything you have learned so far about the character. Let's store it!

Task 10.1 List to store the info

At the start of the game create an empty list called **`info`** to store all of the information we have about the mystery person.

It will look something like this, depending on what questions you've asked and what the answer were:
[["eyes", "blue", "no"],
["hair", "brown", "yes"],
["accessory", "hat", "no"],
["name", "Tim", "no"],
["eyes", "brown", "yes"]]

Task 10.2 List to store the info

Every time the computer says 'yes' or 'no' to a guess, we want to store that information!

Create a variable called **`info`** and store a list of the information you just learned.

Hint

The different kinds of options (eyes, hair, accessory and name) will need different parts to be hard coded. Here's what it would look like for eyes if it wasn't correct:

```
info = ["eyes", eye_guess, "no"]
```

Task 10.3 List to store the info

Each time we get a new bit of data, append it to the **`info`** list.

Hint

This is how we add something to a list using append:

```
pets = ['dog', 'cat']
pets.append('axolotl')
```

Task 10.4 Printing the info

Print the info at the start of every turn.