



Girls' Programming Network

Guess Who!

*Create a program that picks a person at random
and gives you hints to help you Guess Who!*

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

Jeannette Tran
Fiona Lin
Renee Noble
Alex McCulloch
Annie Liu
Bryony Lanigan
Heather Catchpole
Sheree Pudney

Testers

Vivian Dang
Courtney Ross
Rachel Alger
Libby Berrie
Caitlin Macleod
Rashmica Gupta

Part 0: Setting up

Task 0.1: Making a python file

Open the start menu, and type 'IDLE'. Select IDLE.

1. Go to the File menu and select 'New File'. This opens a new window.
2. Go to the File menu, select 'Save As'
3. Go to the Desktop and save the file as 'guess_who.py'

Task 0.2: You've got a blank space, so write your name!

At the top of the file use a comment to write your name!
Any line starting with # is a comment.

```
# This is a comment
```

Use F5 key to run your code (you can also Go to the Run menu and select 'Run Module').
Click OK when it asks you to Save.
Your code does nothing yet!

Task 0.3: What are we going to build today?

We're building a game of Guess Who. You'll need your 'Guess Who People' sheet.

The computer will randomly pick a person from the sheet then the player can guess eye colour, hair colour and accessory to help them work out who the computer picked.

How many goes will it take you to Guess Who?

✓ CHECKPOINT ✓

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

- ☐ You should have a file called guess_who.py
- ☐ Your file has your name at the top in a comment
- ☐ Run your file with F5 key and it does nothing!!
- ☐ You understand what you're going to build

Part 1: Welcome to 'Guess Who'

Task 1.1: Welcome to 'Guess Who'

Let's `print` out a welcome message to the player. You can make the computer say anything you want!

```
Welcome to Guess Who
```

Hint

To print `Hello` use `print("Hello")`

Task 1.2: Who is playing my game?

Let's find out who's playing!

Use `input` to ask the user for their name. Store their answer in a variable called `player_name` so we can use it in our code!

Hint

To find out someone's favourite ice-cream and store it in a variable called `favourite`

```
favourite = input("What is your favourite ice-cream? ")
```

Task 1.3: Let's play!

Now that we know the player's name, let's `print` out a customised message to them.

For example, if the player typed in Annie, we might `print`:

```
Let us start playing Annie
```

Hint

Remember to use the `player_name` variable that you made in Task 1.2!

To print `Hello Annie` we would use `print("Hello " + player_name)`

CHECKPOINT

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

- ☐ Print a welcome message to the player
- ☐ Ask for the player's name
- ☐ Print a customised message to the player
- ☐ Try running your code!

Part 2: Picking a person!

Lists

Task 2.1: Creating a person

Let's create a list for one character on the Guess Who People sheet and store it in a variable called `character`. You can choose any character on the sheet you like.

We want the list to store the character's name, eye colour, hair colour, and accessory.

We'll store them in this order:

```
[<name>, <eye colour>, <hair colour>, <accessory>]
```

Accessory can be a 'hat' or 'glasses' (or 'none' if they don't have a hat or glasses).

Hint

For example:

```
character = ['Aleisha', 'brown', 'black', 'hat']
```

Task 2.2: Print out the character

Let's `print` out the character that you have created. It should look like this:

```
['Aleisha', 'brown', 'black', 'hat']
```

Task 2.3: Splitting up the list!

It's easier to access the character's individual features if we store each one in its own variable. Let's do that now!

Make a variable called `name`. Get the name from the character list you created in Task 2.1 and store it in this variable!

Hint

To get something out of a list, you can use something like this in your program:

```
pets = ['Fluffy', 'Oscar', 'Audrey', 'Molly']  
cat = pets[3]
```

Don't forget that lists start at index 0, so `pets[0]` = Fluffy and `pets[1]` = Oscar.

Task 2.4: More features!

Let's also get the other features of our person out and store them in variables.

1. Make a different variable called `eye_colour`. Get the character's eye colour from the list and store it in here.
2. Make another variable - this time for the hair colour. You can decide what to call it (`hair_colour` could be good). Store the character's hair colour in here.
3. The last feature we need is the character's accessory, so make a variable for this!

Task 2.5: Print out each of the features

Use print statements to print out the name, eye colour, hair colour and accessory of our character.

It should look like this when you run your code:

```
Name is: Aleisha  
Eye colour is: brown  
Hair colour is: black  
Accessory is: hat
```

✓ CHECKPOINT ✓

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

- ☐ Created a person
- ☐ Print out the person
- ☐ Split up the features of the person and store them in variables
- ☐ Print out the features of the person
- ☐ Run your code!

Part 3: Guess who?

Task 3.1: Guessing someone's name

Use `input` to ask the player to guess the name of your character. Save their answer in a variable - name it something like `guess`.

Comment out the code you added in 2.2 and 2.5 where we `print` out the character we created and their features! It would be a pretty boring game if we just told the player who the character is!

Hint

You can comment out code by just adding `#` at the start of the line.

```
#print(character)
```

If
Statements

Task 3.2: Check if they have guessed correctly!

Use `if` and `else` statements to tell the player whether they have made the right guess.

You should also congratulate them if they have guessed it right:

```
Guess who? Aleisha  
You got it right!
```

You should print out the correct name if they have guessed wrong, like below:

```
Guess who? Mary  
Nope, sorry, it was Aleisha!
```

Hint

In the `if` statement, compare the player's guess with the name of the character you chose. Don't forget to use `==`.

To check if someone guessed my favourite ice-cream

```
favourite = "mango"  
guess = input("Guess my favourite icecream? ")  
if guess == favourite:  
    print("You are right")  
else:  
    print("You are wrong")
```


✓ CHECKPOINT ✓

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

- ☐ Ask the player to guess who and store it in a variable
- ☐ Congratulate them if they guess it right
- ☐ Tell them what the correct name was if they are wrong
- ☐ Run your code and test different names

★ BONUS 3.3: ALEX alex or AEx

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

We can use `word = word.title()` to change what the player entered to title case. Title case is when the first letter is upper case and all the rest are lower case, like a name!

Update your code so we're always using the title case version of what your player entered!

Part 4: Let's get more information!

We are just guessing blindly at the moment, which isn't very fun! Let's let the player get more information about the person before they have to Guess Who.

Task 4.1: Asking about eyes

Find your code you wrote in 3.1 that asks the player to guess the name of the character.

BEFORE that code, ask the player to guess what the person's eye colour is!

Store their answer in a variable - call it something like: `eye_guess`

Your question should look something like this if the user guesses brown eyes.

```
Guess their eye colour? brown
```

Task 4.2: Check the eyes

Check to see if the eye colour that the player guessed is the correct one using `if` and `else` statements. If they are right, tell them "Yes", otherwise tell them "No".

For example, if your character's eye colour is brown and the user guesses right:

```
Guess their eye colour? brown
Yes
```

Task 4.3: What's their hair colour?

Do the same thing again that you did for 4.1 and 4.2, but for the person's hair colour!

Put this after you ask them to guess the eye colour, but before you ask them to guess the name.

Task 4.4: What's their accessory?

Do the same thing again that you did for 4.1 and 4.2, but for the person's accessory!

Put this after you ask them to guess the hair colour, but before you ask them to guess the name.

✓ CHECKPOINT ✓

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

- ☐ Ask the user for their eye colour guess
- ☐ Print out if their eye colour guess was right
- ☐ Ask the user for their hair colour guess
- ☐ Print out if their hair colour guess was right
- ☐ Ask the user for their accessory guess
- ☐ Print out if their accessory guess was right
- ☐ Run your code!

★ BONUS 4.4: BLUE Blue bLuE

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

We can use `word = word.lower()` to change what the player entered to lowercase. Update your code so we're always using the lowercase version of what your player entered for their eye colour, hair colour and accessory guesses.

★ BONUS 4.5: Not so fast!

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

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 a command to make our program sleep for a few seconds.
- 2) Before we tell the user if they guessed correctly, add a line that says `time.sleep(1)`
This will make our program 'sleep' for a second! You can adjust it to any time you want.

Part 5: Choose a random person.

Task 5.1: One character is not enough

Comment out the line where you created a `character` in Task 2.1. This should look like:

```
# character = ['Aleisha', 'brown', 'black', 'hat']
```

We're about to get the whole bunch of characters and we'll choose a character randomly from that. You can add your own characters to the group later.

Task 5.2: Copy the list of people

Go to this link: https://bit.ly/GPN_22 and copy the `list` of the people on the Guess Who People sheet and all of their features. This is `Guess_Who_People_List`.

Paste the list at the top of your file so that it looks like this:

```
people = [['Aleisha', 'brown', 'black', 'hat'],
          ['Brittany', 'blue', 'red', 'glasses'],
          ['Charlie', 'green', 'brown', 'glasses'],
          ['Dave', 'blue', 'red', 'glasses'],
          ['Eve', 'green', 'brown', 'glasses'],
          ['Frankie', 'hazel', 'black', 'hat'],
          ['George', 'brown', 'black', 'glasses'],
          ['Hannah', 'brown', 'black', 'glasses'],
          ['Isla', 'brown', 'brown', 'none'],
          ['Jackie', 'hazel', 'blonde', 'hat'],
          ['Kevin', 'brown', 'black', 'hat'],
          ['Luka', 'blue', 'brown', 'none']]
```

Random

Task 5.3: 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 5.4: Choose a random person

Let's make the computer pick a random person from the list of people we just copied in.

After your commented out character line from 5.1, use `random.choice` to pick from the list of people. Store the random choice in a variable called `character`.

Hint

If I wanted to choose a random food and store it in a variable called `dinner`, I could use code like this:

```
food = ["pizza", "curry", "nutella", "omelette"]
dinner = random.choice(food)
```

Task 5.5: Print out the character

Print out the `character` that the computer has chosen.

Try running your code a couple of times! You should get a random person each time.

Does your game give you the right answer for your eye colour, hair colour and accessory guess for the chosen character?

```
['Hannah', 'brown', 'black', 'glasses']
```

Task 5.6: Don't give them the answer!

Now you know your game is working, comment out the line you added in 5.5 to print the chosen character. You don't want to give the player the answer!

Task 5.7: Play your game!

Run your code and play Guess Who?

As you get the answers to your questions, use counters to cover up people who you know haven't been chosen as the character by the computer.

For example, if you guess `eye_colour = green` and the game tells you No, you know it can't be Charlie or Eve as they have green eyes. Good luck!

✓ CHECKPOINT ✓

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

- ☐ Comment out the character you made in Task 2.1
- ☐ Copied and pasted the list
- ☐ Chosen a random person from the list
- ☐ Print out the randomly chosen person
- ☐ Comment out printing the randomly chosen person
- ☐ Run your code! Play the game!

★ BONUS 5.6: Get creative!

You can add your own characters!

We've left room for you to draw your own people at the bottom of the Guess Who People character sheet.

Give them a name, eye colour, hair colour and accessory add them into the list of people at the top of your code!

Feel free to add yourself, your friends or one of the wonderful tutors at GPN!

Part 6: Again, Again, Again!

We want to play 'Guess Who' until we guess the correct person! Let's add a loop to guess on repeat!

While
Loops

Task 6.1: Loop time!

Create a while loop that runs forever so you can have as many guesses as you want!

After you've randomly selected the character and separated the features, but before you make any guesses, add this line so the guesses repeat.

```
while True:
```

Task 6.2: Indenting your code

Things we want to do repeatedly need to be indented under the `while` loop. Only those things indented will get repeated.

Indent all the code under the `while True:` line so it repeats and we get lots of guesses.

Hint

Indented lines have a tab (the big empty space) at the start like this:

```
while True:
    # THIS IS INDENTED
```

You can indent many lines of code by selecting all the lines you want to indent and then clicking on TAB.

Task 6.3: Don't tell them the answer!

When the player guesses the wrong name, update your `else` statement so that they aren't told who the correct person is!

Task 6.3: Stopping

We want our program to stop when we guess the right name!

When the player guesses the right name and we have told them, add a **break** to stop the loop. Make sure it is indented under the **if**.

✓ CHECKPOINT ✓

If you can tick all of these off you can go to Part 7:

- ☐ Create a while loop that lets your game keep going!
- ☐ Your game code for guessing is inside the while loop
- ☐ Don't print the name of the person if you guess incorrectly.
- ☐ The game only ends when you guess the right person!

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.