

The joke machine

Introduction

This is a **pair programming** activity. You will be working in pairs to create a joke machine program. Remember to switch driver and navigator roles every five minutes.

Scenario

The joke machine is a program that tests your skills in joke punchlines. It gives you the opening line to a joke and you must guess the punchline. If you are correct, then you win a point!

Use the jokes provided to make a program that will:

- Include an introduction to the game
- Tell the start of a joke
- Allow the user to guess the punchline
- Check if the user is correct
 - Give the user a point if they guess correctly
 - Provide feedback if they guess correctly
 - Provide feedback if they are incorrect
- Reveal the final score at the end

Jokes

What is pink and fluffy?

Pink fluff

What is brown and sticky?

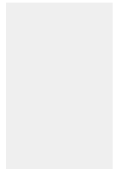
A brown stick

What is black and white and red all over?

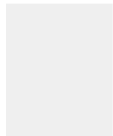
A newspaper

Task 1: Introductions

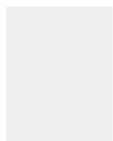
Tick ✓ off the subtasks as you go:



Write text that will output a message that introduces the user to the game. It could include a title and some simple instructions.



Use the `print` statement to display this text to the user.



Test your code.

Sample code block:

```
print("Guess the punchline...")
```

Common errors (use this checklist to help you debug your code):

- ☐ Capital P used for print
- ☐ Brackets missing from start or end of text
- ☐ Speech marks missing from start or end of text

Task 2: Ask for the punchline

Tick ✓ off the subtasks as you go:

☐

Write the opening statement to the first joke

☐

Create a variable to hold the user's guess

☐

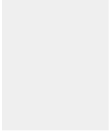
Decide if you want the data to be converted to uppercase or lower case, and use the appropriate function for this

☐

Write an `if` statement that includes a condition to check if their punchline guess is correct

☐

Provide some text to display if they are correct



Test your code

Sample code block:

```
print("Here is the start of my joke")  
  
punchline = input().upper()  
  
if punchline == "THE PUNCHLINE":  
    print("Well done, you were correct!")
```

Common errors (use this checklist to help you debug your code):

- ☐ Upper case I is used for If
- ☐ One = sign is used instead of ==
- ☐ Colon : missing at the end of the if
- ☐ Indents/spaces have been missed
- ☐ Quotations missed around the punchline in the condition
- ☐ Punchline in the condition is written in uppercase but .lower () has been used

Task 3: Keep score

Tick ✓ off the subtasks as you go:

Create a variable to track the score

Initialize the variable at the top of the code `score = 0`

Increment the score within the if statement `score = score + 1`

Test your code by placing `print(score)` on a new line

Delete the `print(score)` line of code once testing is complete.

Common errors (use this checklist to help you debug your code):

- ☐ Score hasn't been initialized at the top of the code
- ☐ Incorrect spelling of score variable
- ☐ Score hasn't been incremented in the correct place (it should be directly underneath the `well done` statement, inside the if statement)

Task 4: Feedback if they are incorrect

Tick ✓ off the subtasks as you go:

Add an `else:` underneath the `if`

Add a `print` statement that provides feedback on the joke

Test your code

Sample code block:

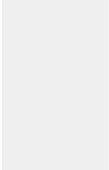
```
else:  
    print("Wrong, it was this punchline")
```

Common errors (use this checklist to help you debug your code):

- ☐ Else has a capital E
- ☐ Colon `:` missing after the `else`
- ☐ `print` statement not indented
- ☐ `else` isn't inline with the `if`

Task 5: Add more jokes

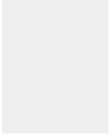
Tick ✓ off the subtasks as you go:



Double check that your code is working correctly. It should give feedback if the user is correct or incorrect. Execute the program to check this, try each scenario.



Add the other two jokes to the program.



Make sure that you test regularly.

Common errors (use this checklist to help you debug your code):

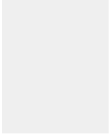
- ☐ Indents in the wrong place (refer back to your original working code block to check if you indents are lined up)
- ☐ Colons missing
- ☐ Capital letters used at the start of key terms: else, if, print

Task 6: Reveal the final score

Tick ✓ off the subtasks as you go:



Use a `print` statement to reveal the final score to the user



Test your code

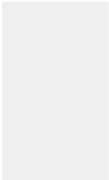
Common errors (use this checklist to help you debug your code):

- ☐ `score = score + 1` has not been correctly placed inside each `if`
- ☐ Incorrect spelling of score variable

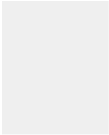
Task 7: Improve your game

Your user might type something that is correct but is worded slightly differently. For example, instead of writing a `brown stick` they might just write `brown stick`. This would still technically be a correct answer.

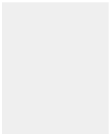
Tick ✓ off the subtasks as you go:



Add an `elif` block to the `brown stick` and the newspaper jokes so that the user will still get a point if they miss the `a`



Remember that your user will also earn a point for their correct answer



Test your code

Sample code:


```
elif punchline == "BROWN STICK":
```

```
    print("You got it!")
```

```
    score = score + 1
```

Common errors (use this checklist to help you debug your code):

- ☐ Elif has a capital E instead of lower case
- ☐ The colon : is missing from the end of the elif
- ☐ print and score aren't indented within the elif
- ☐ elif written in the wrong place (it should be after the if and before the else)

Explorer tasks

Tick ✓ off the subtasks as you go:

Improve the game further by thinking of other possible answers that users might give and provide feedback for those

Use another set of conditions to provide feedback to the user based on their score. There should be feedback for 0, 1, 2, or 3