LESSON : 4

Making Decisions: Part I

ANNOUNCEMENTS

Before we start, let's see if we have any codewizards announcements.

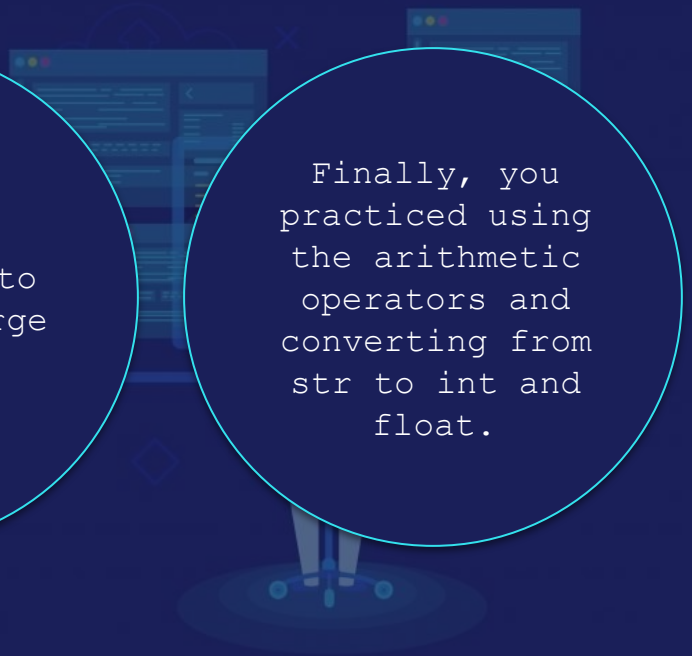
Click here to go to the notice board.

WHAT WE DID IN LAST CLASS

You learned how to perform string interpolation as an alternative to string concatenation.

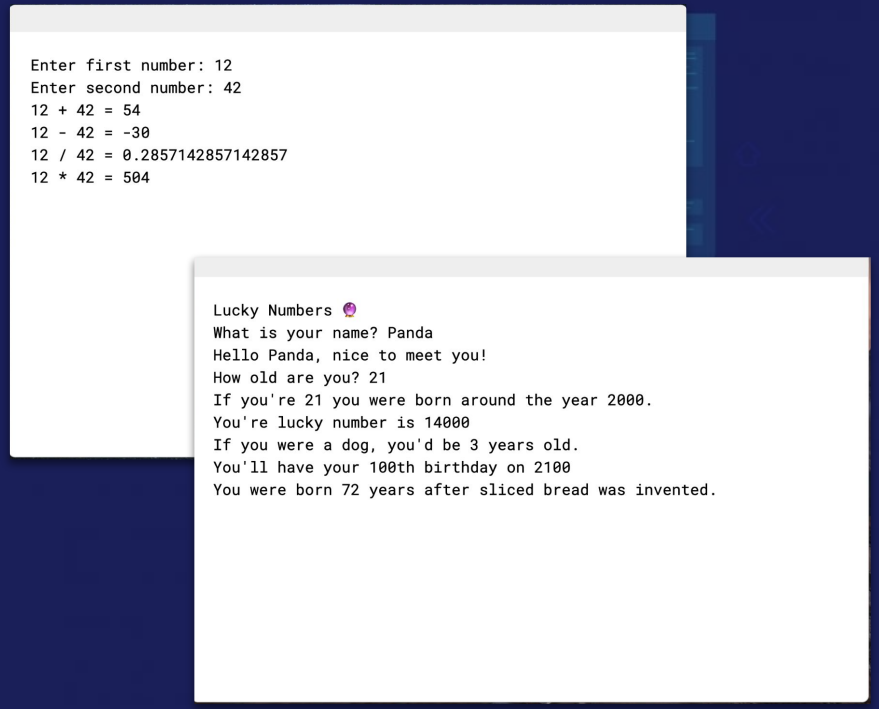
You also learned how to deal with large strings.

Finally, you practiced using the arithmetic operators and converting from str to int and float.



HOMEWORK SHOWCASE

Any doubts about homework or from the previous class?



```
Which option is a lie?
    (1) The platypus has 3 stomaches
    (2) Dolphins sleep with one eye open
2
That's incorrect! 'Dolphins sleep with one eye open' is true!
Which option is a lie?
    (1) A cat served as mayor of an Alaskan town for 20 years.
    (2) Dogs are distant relatives of earthworms
2
That's correct! 'Dogs are distant relatives of earthworms' is a lie!
You get a point added to your score!
You got 1 out of 2 questions correct.
You only missed one question, that's pretty good!
```

Project of the day

QUIZ APP

TODAY, WE'LL CREATE A SMALL QUIZ APP BY COMPLETING THE FOLLOWING STEPS:

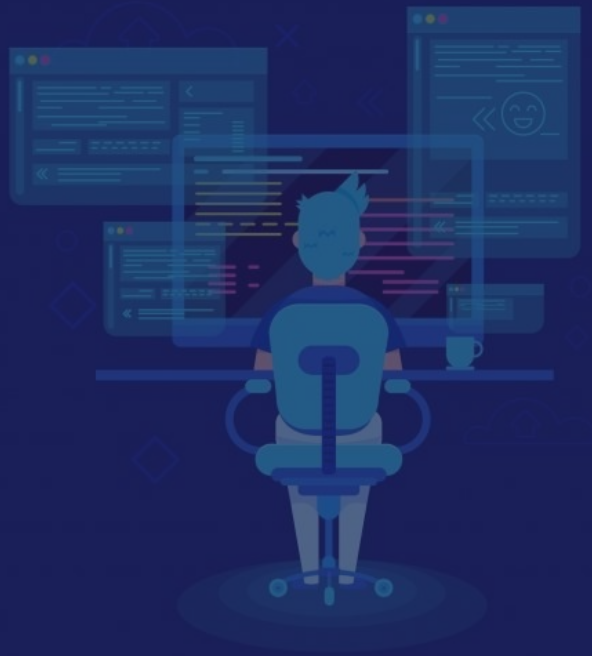
Step 01: Display Options And Get The User's Choice

Step 02: Use Conditional Statements To Check The User's Choice

Step 03: Track The Number Of Correct Answers

EXERCISE: 1/3

Display Options And Get The User's Choice



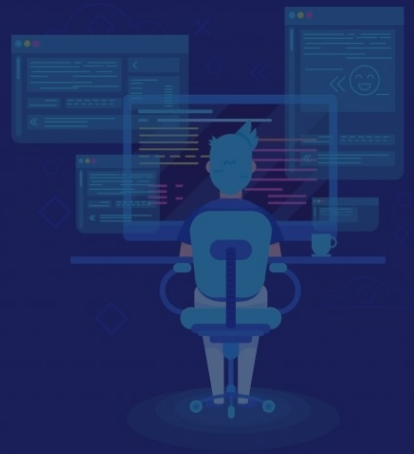
2) *Use Conditional Statements To Check The User's Choice*

3) *Track The Number Of Correct Answers*

1.1 - Display Options And Get The User's Choice

- Create two options for your quiz and store them in variables.
- Add these options to a multi-line f-string.
- For example:

```
(1) The platypus has 3 stomachs
(2) Dolphins sleep with one eye open
```



CHEAT SHEET

Step 1

In the file **index.py**,

- ❏ Create two options for your quiz and store them in variables.
- ❏ Add these options to a multi-line f-string.

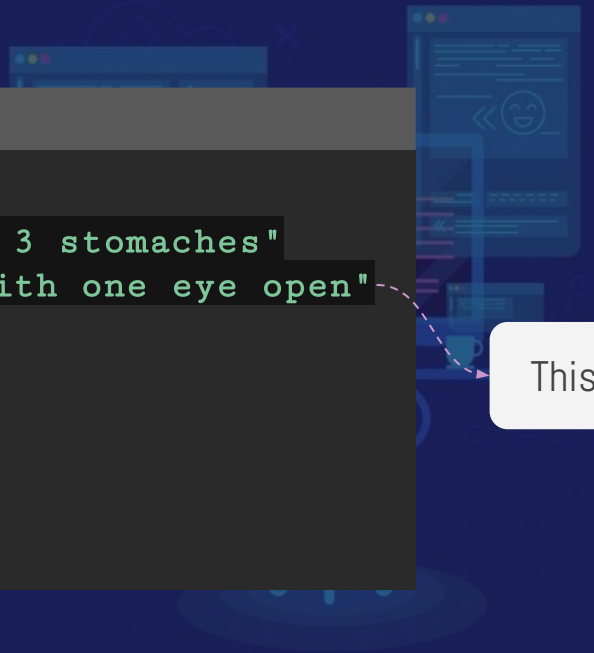
SOLUTION

This is a lie

index.py

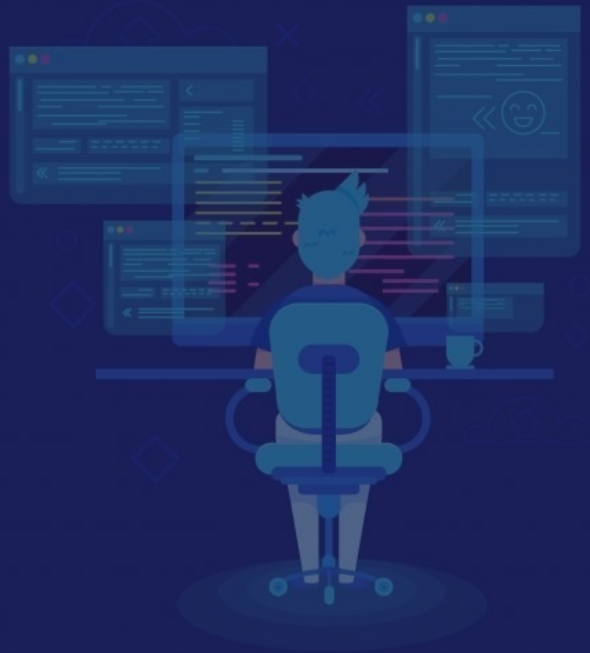
```
option1 = "The platypus has 3 stomaches"
option2 = "Dolphins sleep with one eye open"

options = f"""
    (1) {option1}
    (2) {option2}
"""
```

This is true



1.2 - Display Options And Get The User's Choice

- Prompt the user for a guess about which option is a lie.
  - You should display the possible options along with this prompt.
- Store the user's choice in a variable.
  - Remember to convert to `int()`



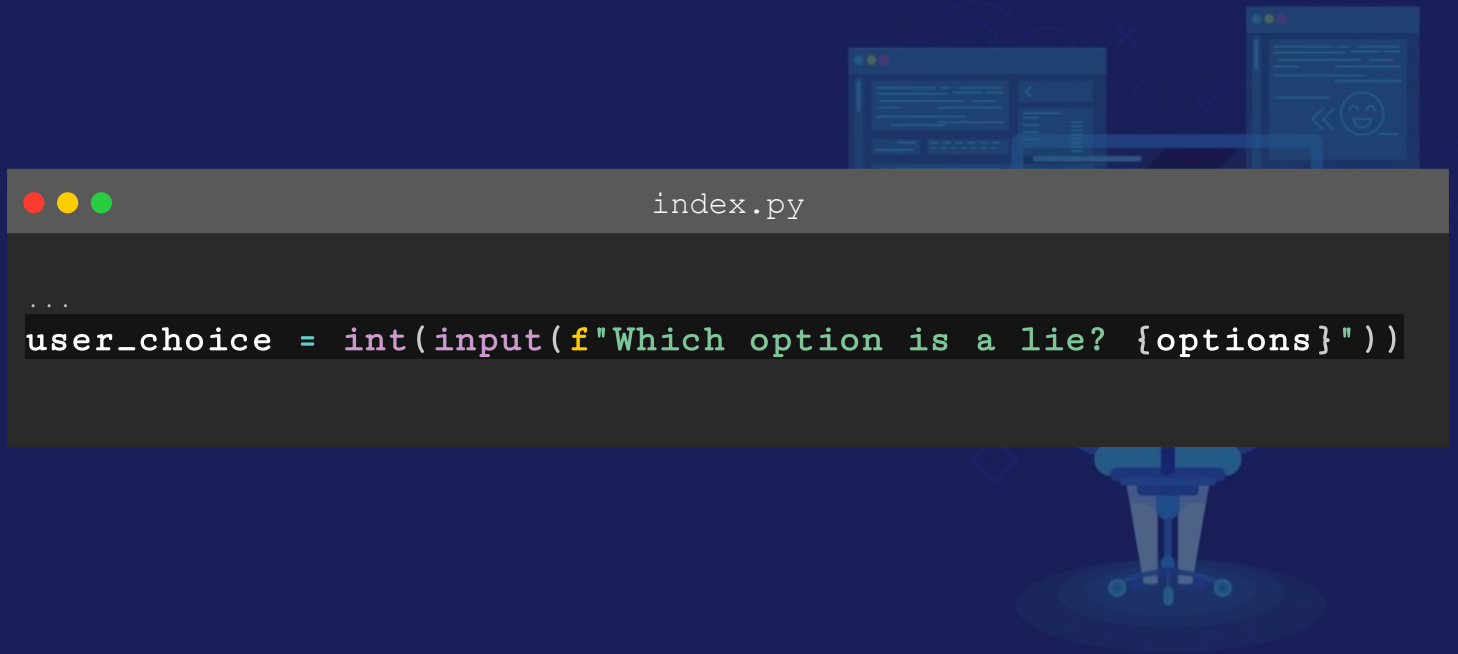
CHEAT SHEET

Step 2

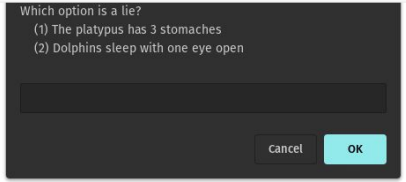
In the file **`index.py`**,

- ❏ Prompt the user for a guess about which option is a lie.
- ❏ Store the user's choice in a variable.
  - ❏ Remember to convert to **`int()`**

SOLUTION


index.py
...
user_choice = int(input(f"Which option is a lie? {options}"))

GETTING THE USER'S CHOICE

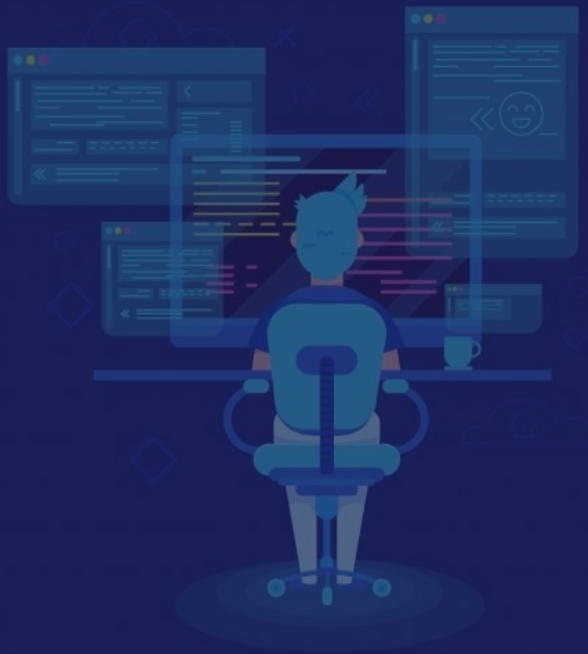


EXERCISE: 2/3

~~1) *Display Options And Get The User's Choice*~~

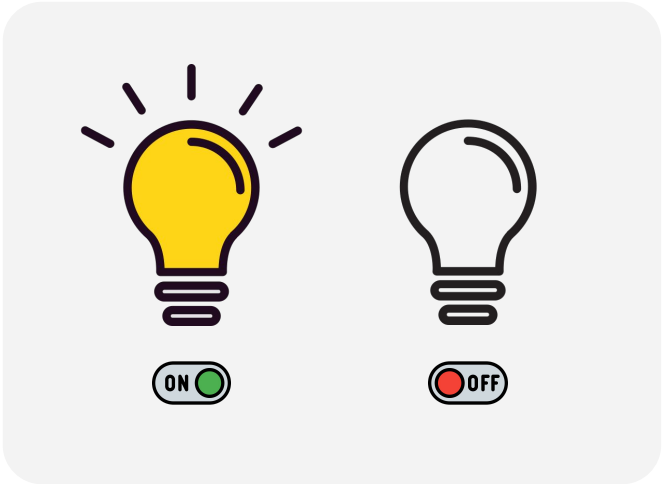
Use Conditional Statements To Check The User's Choice

3) *Track The Number Of Correct Answers*



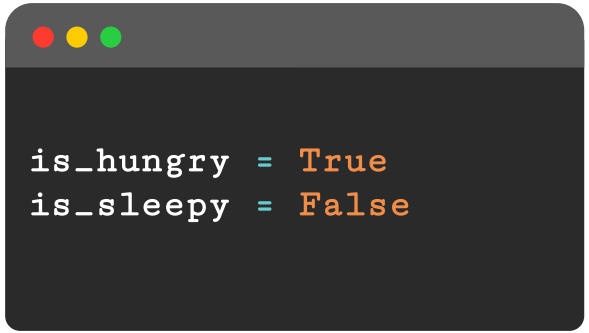
BOOLEAN VALUES

- Sometimes we need to examine the state of certain variables or expressions in our code. Is it on or off, valid or invalid, yes or no, a one or zero?
- The state we evaluate these variables or expressions to is known as a boolean (`bool`) value.
- Boolean values can be `True` or `False`.



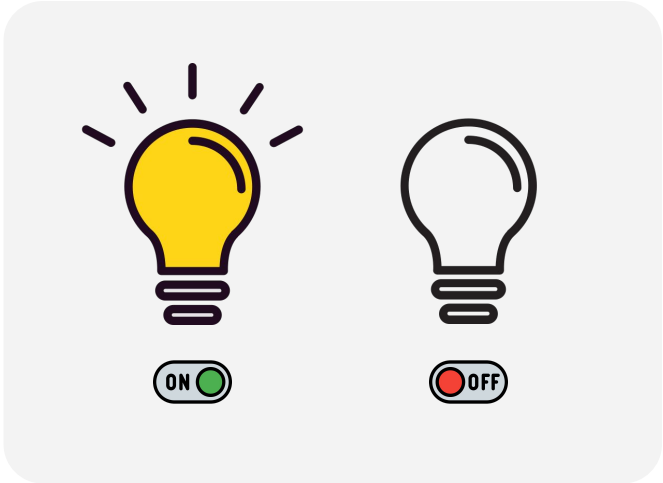
SYNTAX: BOOLEAN (BOOL) VALUES

You can directly assign the values `True` and `False` to `bool` variables.



GENERATING BOOLEAN VALUES FROM EXPRESSIONS

- Usually, you won't use boolean values directly. Instead, you'll create expressions that result in a boolean value.
- These expressions generally take the form of a comparison between an unknown value (like some input from the user) and a known value.



SYNTAX: COMPARISON OPERATORS

There are **6** comparison operators that you can use to generate boolean values in an expression.

```
age = 21
age > 18   # True
age < 18   # False
```

```
height = 6.2
height <= 5.5   # False
height >= 6.2   # True
```

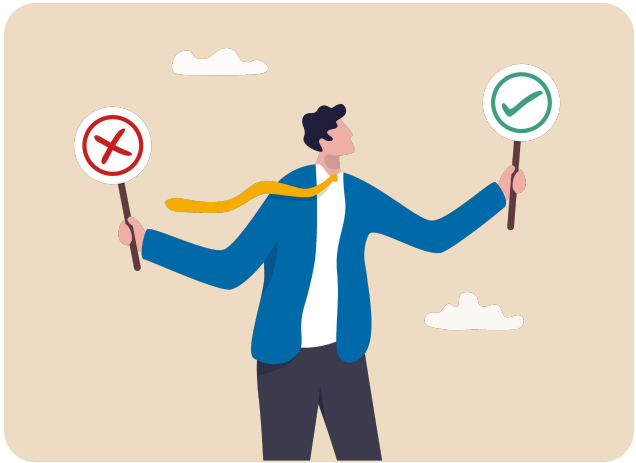
```
favorite_food = "Tacos"
favorite_food == "Pizza"   # False
favorite_food != "Pizza"   # True
```

A single = sign is an assignment of a value to a variable.

The == and != operators are comparisons of equality.

CONDITIONAL STATEMENTS

- Often, you'll need to make some sort of decision in your programs based off unknown values.
- Conditional statements allow you to perform different actions based on the value of a boolean comparison.



SYNTAX: CONDITIONAL STATEMENTS - INTRODUCING "IF"

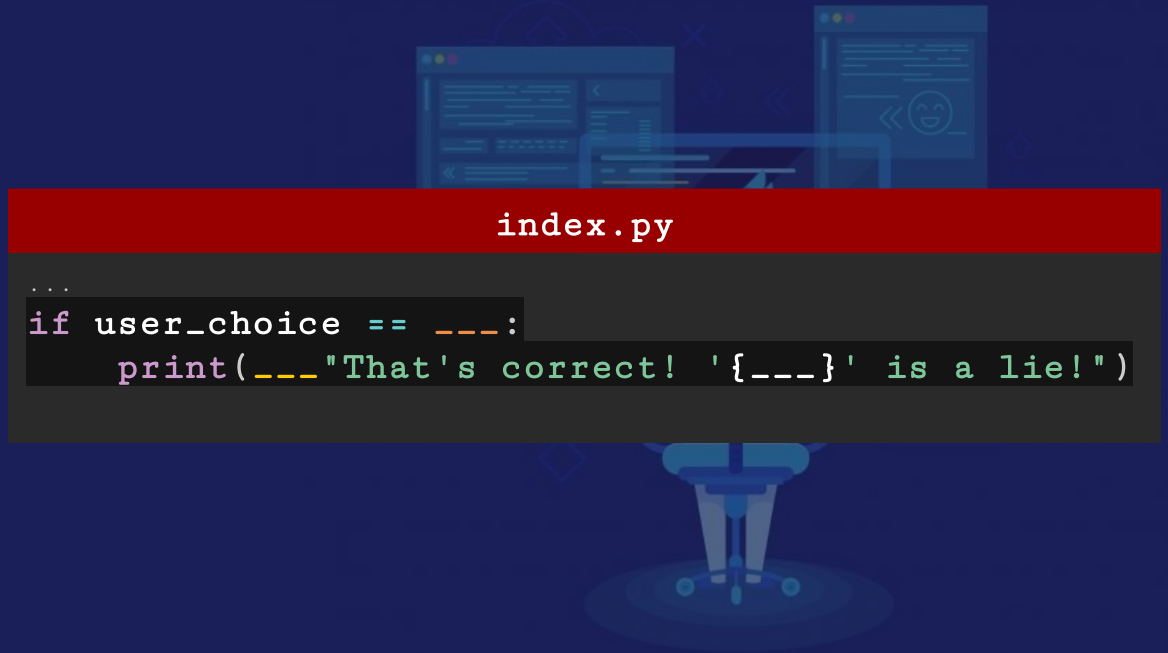
- The "**`if`**" conditional statement allows you to run a block of code if the expression evaluates to **`True`**.
- Everything that should run if the conditional statement is **`True`** should be indented 4 spaces.

CHEAT SHEET

Step 1

In the file **index.py**,

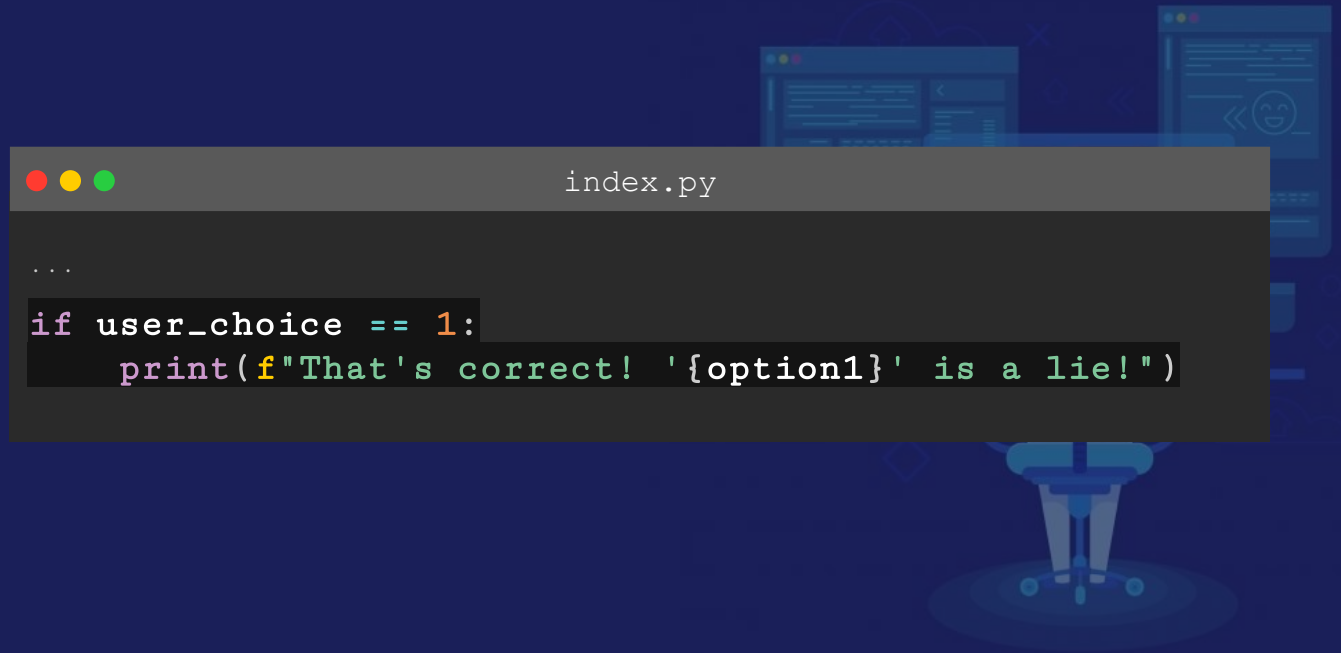
- ❏ See if the **user_choice** is the correct value and print a message.



index.py

```
...
if user_choice == ___:
    print(___"That's correct! '{___}' is a lie!")
```

SOLUTION


index.py
...
if user_choice == 1:
print(f"That's correct! '{option1}' is a lie!")

CHECKING IF THE USER MADE THE CORRECT CHOICE

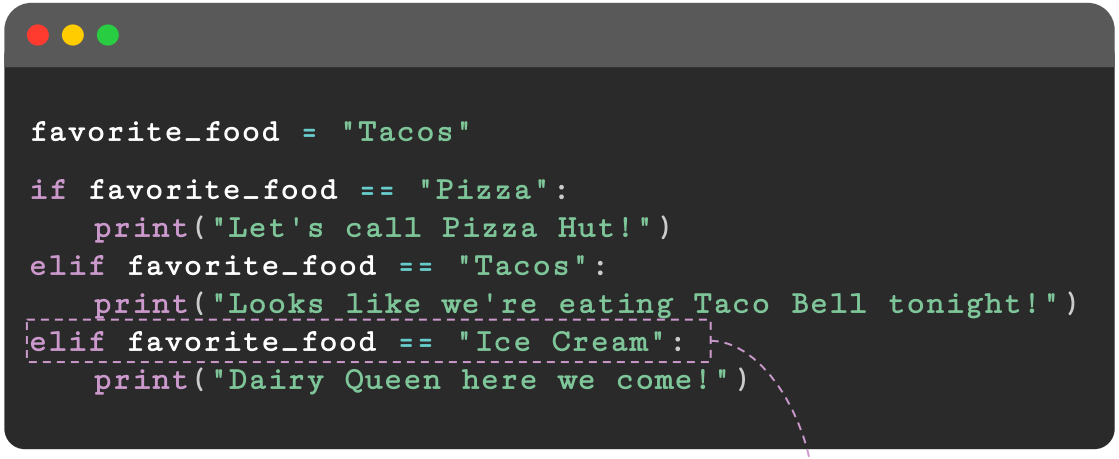
```
Which option is a lie?
    (1) The platypus has 3 stomaches
    (2) Dolphins sleep with one eye open
1
That's correct! 'The platypus has 3 stomaches' is a lie!
```

MULTIPLE CONDITIONAL STATEMENTS

- What happens if you need to ask multiple related questions?
- Creating multiple "`if`" statements works, but there's a better way.

SYNTAX: INTRODUCING ELIF STATEMENTS

- The **`elif`** conditional statement allows you to group **`multiple`** conditional statements together.
- The first conditional statement that is **`True`** will stop the rest from running.

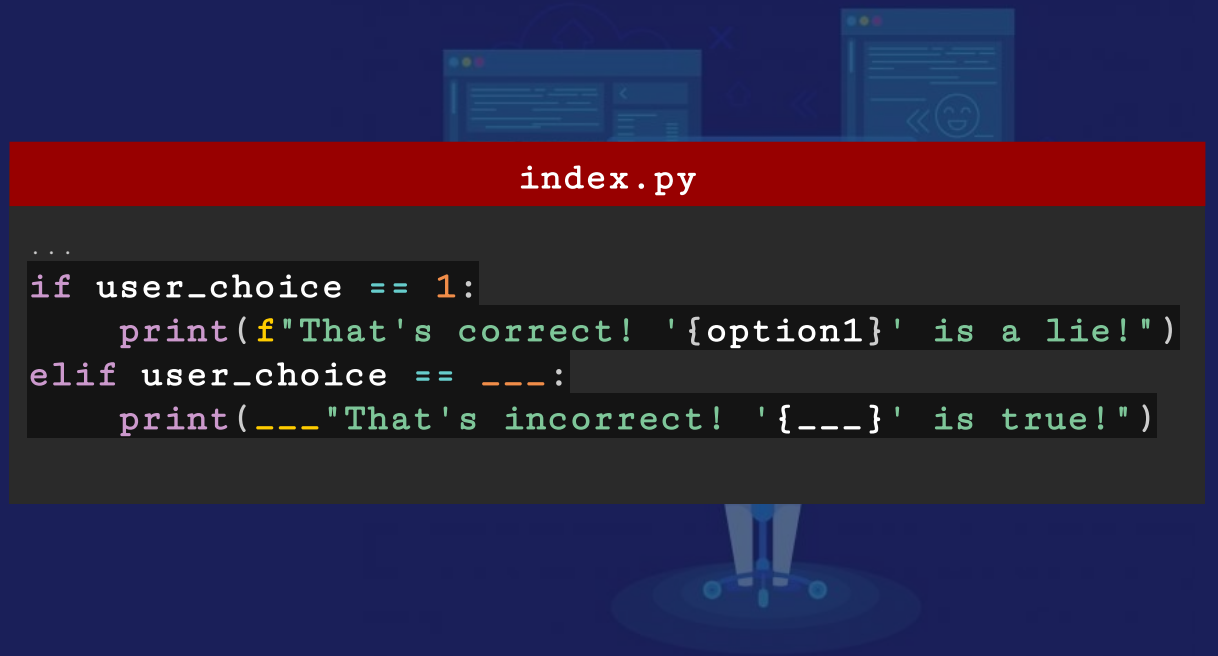


CHEAT SHEET

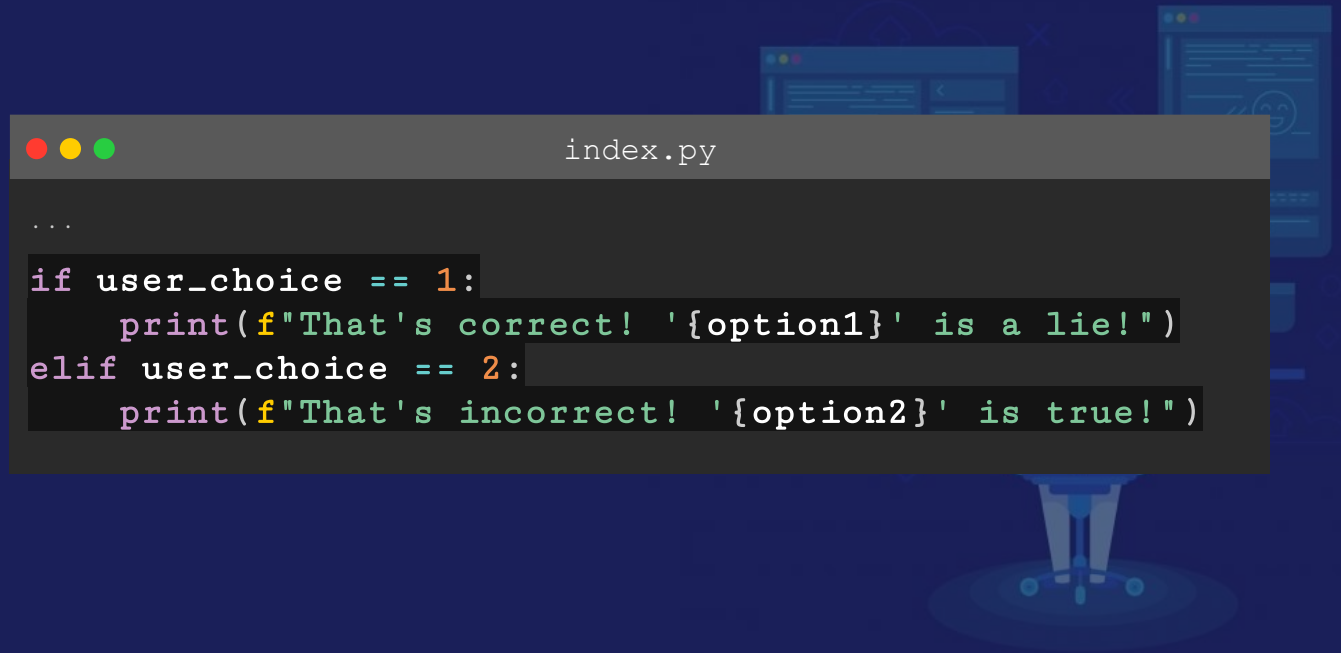
Step 2

In the file **index.py**,

- ❏ See if the **user_choice** is the incorrect option and print a message.



SOLUTION


index.py
...
if user_choice == 1:
print(f"That's correct! '{option1}' is a lie!")
elif user_choice == 2:
print(f"That's incorrect! '{option2}' is true!")

CHECKING IF THE USER MADE THE INCORRECT CHOICE

```
Which option is a lie?
    (1) The platypus has 3 stomaches
    (2) Dolphins sleep with one eye open
2
That's incorrect! 'Dolphins sleep with one eye open' is true!
```

THE DEFAULT CONDITIONAL STATEMENTS

Sometimes, you'll want to execute a block of code when all other conditional statements are **`False`**.

SYNTAX: - INTRODUCING THE ELSE CONDITIONAL STATEMENTS

The "**`else`**" conditional statement runs when all other conditional statements are **`False`**.

```
favorite_food = "Sushi"
if favorite_food == "Pizza":
    print("Let's call Pizza Hut!")
elif favorite_food == "Tacos":
    print("Looks like we're eating Taco Bell tonight!")
elif favorite_food == "Ice Cream":
    print("Dairy Queen here we come!")
else:
    print("I have no idea what you like!")
```

This code block is the one that runs.

CHEAT SHEET

Step 3

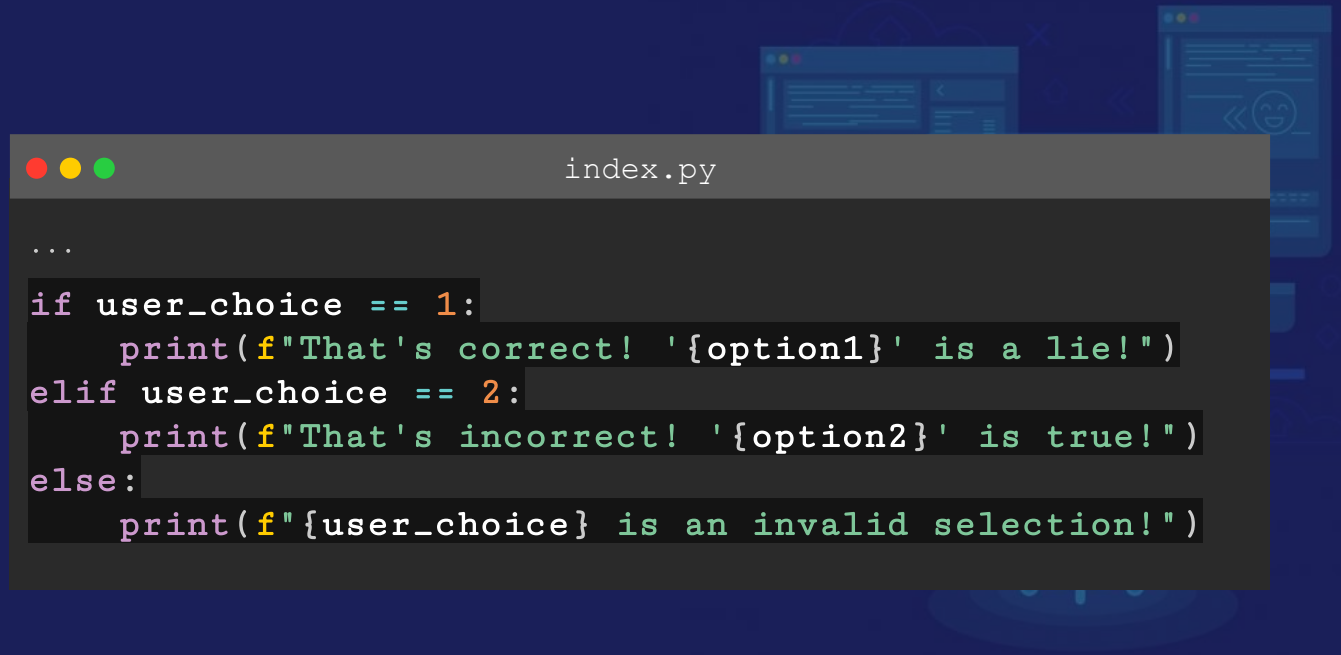
In the file **index.py**,

- Print a message if the **user_choice** is an unknown value.

index.py

```
...
if user_choice == 1:
    print(f"That's correct! '{option1}' is a lie!")
elif user_choice == 2:
    print(f"That's incorrect! '{option2}' is true!")
else:
    print(f"{___} is an invalid selection!")
```

SOLUTION


index.py
...
if user_choice == 1:
print(f"That's correct! '{option1}' is a lie!")
elif user_choice == 2:
print(f"That's incorrect! '{option2}' is true!")
else:
print(f"{user_choice} is an invalid selection!")

CHECKING IF THE USER MADE ANY INVALID CHOICE

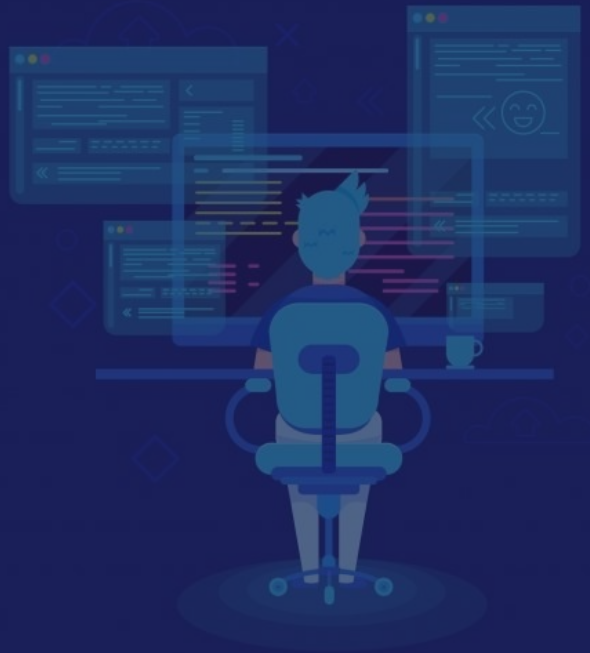
```
Which option is a lie?
    (1) The platypus has 3 stomaches
    (2) Dolphins sleep with one eye open
3
3 is an invalid selection!
```

EXERCISE: 3/3

~~1) Display Options And Get The User's Choice~~

~~2) Use Conditional Statements To Check The User's Choice~~

Track The Number Of Correct Answers



UPDATING VALUES IN VARIABLES

- We can use variables to keep track of a value that will change throughout our program.
- Imagine we want to track the number of tacos a user eats.
- They would start with 0 tacos

```
tacos_eaten = 0
```

UPDATING VALUES IN VARIABLES (CONTINUED)

- Our `tacos_eaten` variable assigns a name to a value, but that value doesn't always have to stay the same.
- We can update a variable's value at any time and with any new value we wish.

```
tacos_eaten = 1
```

UPDATING VALUES IN VARIABLES (CONTINUED)

- That works well for the first taco, but what about the next one?
- How would we know that 2 should be the next value?
- To add (or subtract) a value from `tacos_eaten`, we can use the variable itself to calculate the new value.

```
tacos_eaten = 1
tacos_eaten = tacos_eaten + 1
print(tacos_eaten)  # 2
tacos_eaten = tacos_eaten + 1
print(tacos_eaten)  # 3
```

CHEAT SHEET

Step 1

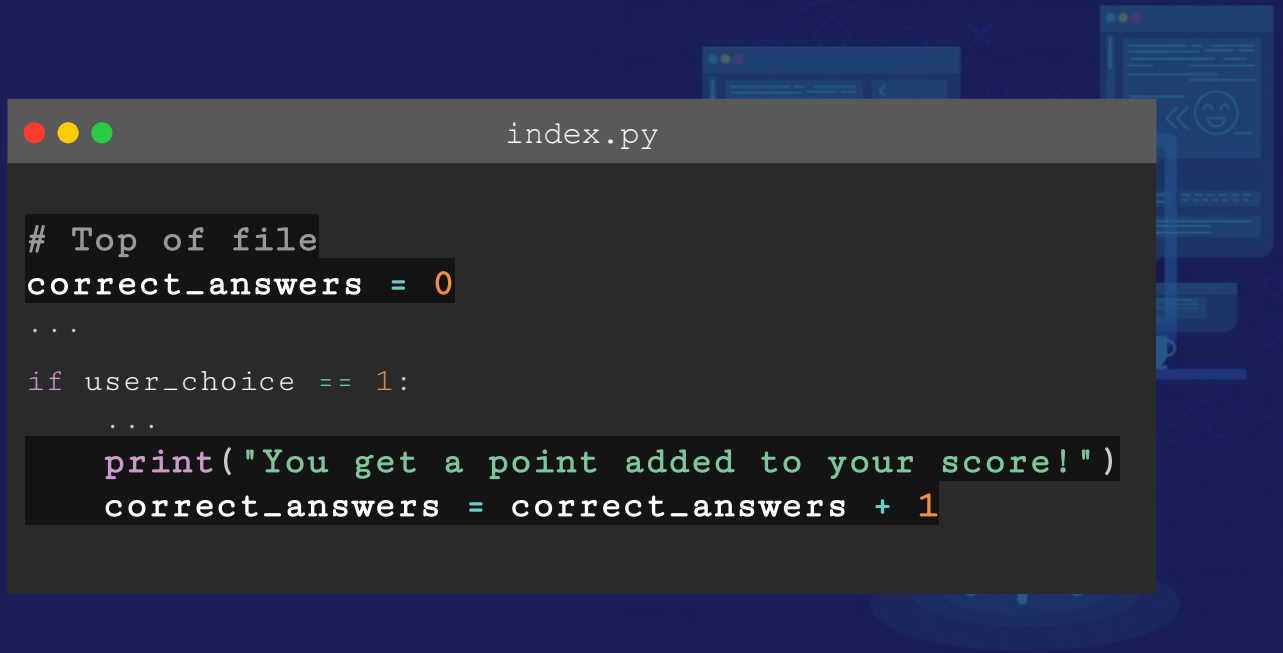
In the file **`index.py`**,

- ❏ At the top of your file, create a variable to track correct answers
- ❏ Increment this variable when the user guesses correctly and print a message telling them so.

index.py

```
# Top of file
correct_answers = ___

...

if user_choice == 1:
    ...
    print("You get a point added to your score!")
    correct_answers = ___ + 1
```

SOLUTION


index.py
# Top of file
correct_answers = 0
...
if user_choice == 1:
...
print("You get a point added to your score!")
correct_answers = correct_answers + 1

ADDING A POINT FOR CORRECT ANSWERS

```
Which option is a lie?
    (1) The platypus has 3 stomaches
    (2) Dolphins sleep with one eye open
1
That's correct! 'The platypus has 3 stomaches' is a lie!
You get a point added to your score!
```

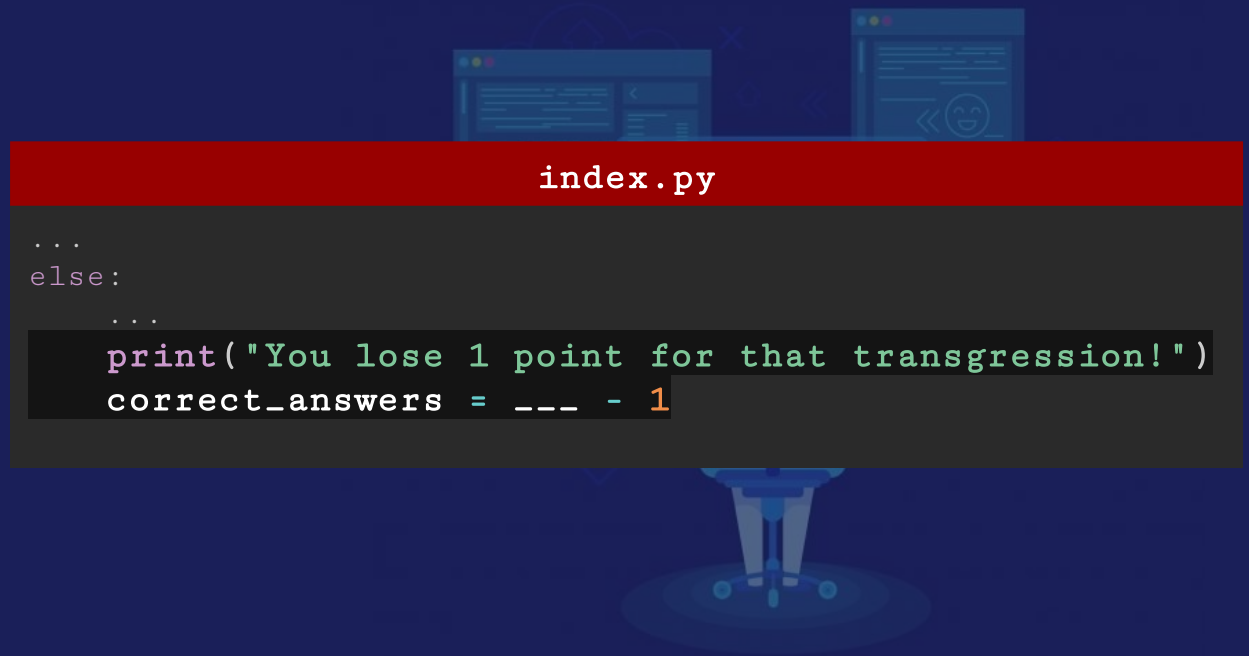
- Let's take away a point for an invalid answer next.
- Do you know where we'd do this?

CHEAT SHEET

Step 2

In the file **`index.py`**,

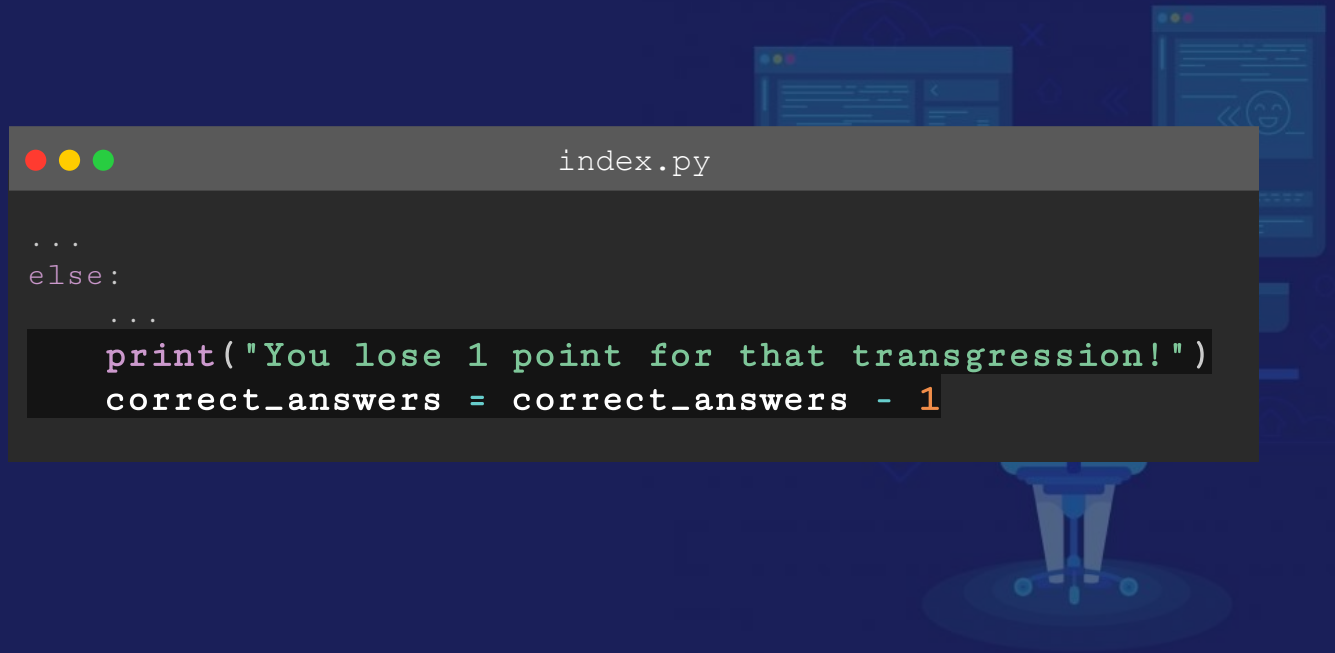
- Decrement the **`correct_answers`** when the user makes an invalid guess and print a message telling them so.



index.py

```
...
else:
    ...
    print("You lose 1 point for that transgression!")
    correct_answers = ___ - 1
```

SOLUTION


index.py
...
else:
...
print("You lose 1 point for that transgression!")
correct_answers = correct_answers - 1

REMOVING A POINT FOR INVALID ANSWERS

```
Which option is a lie?
    (1) The platypus has 3 stomaches
    (2) Dolphins sleep with one eye open
3
3 is an invalid selection!
You lose 1 point for that transgression!
```

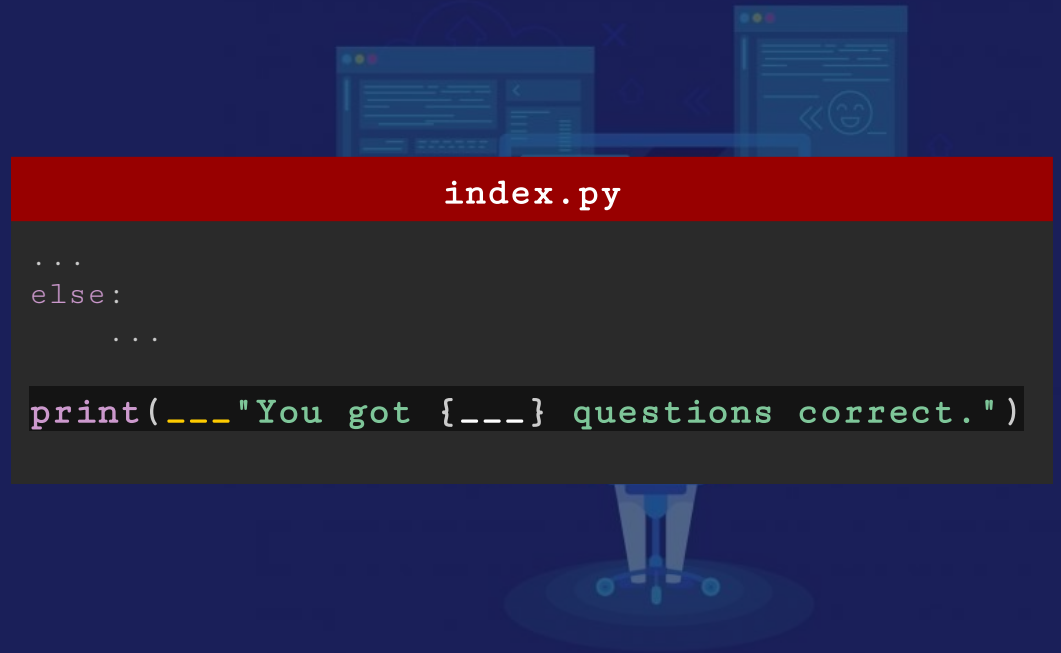
- Let's print the correct answers at the end of our program.
- Does anyone know how to do this?

CHEAT SHEET

Step 3

In the file **`index.py`**,

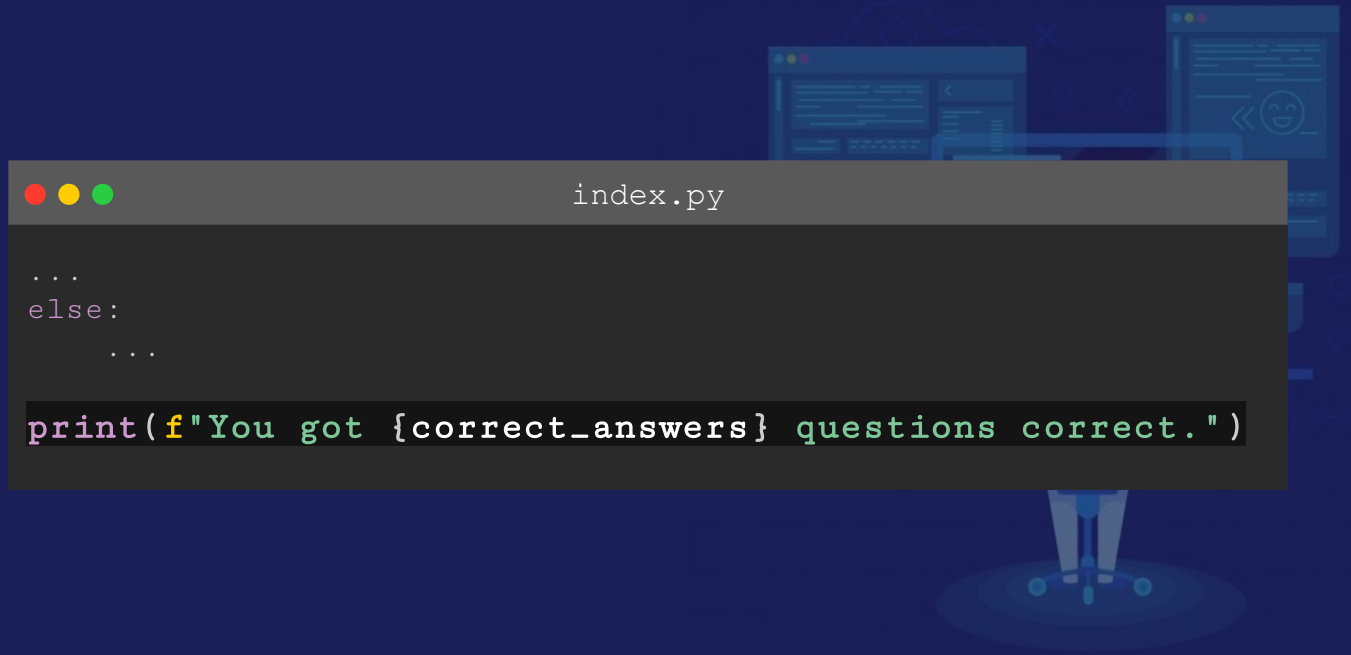
- Print the **`correct_answers`** at the end of the program.
- Make sure this **`print()`** statement is *outside* of the conditional statements.



index.py

```
...
else:
    ...

print(___"You got {___} questions correct.")
```

SOLUTION


index.py
...
else:
    ...
print(f"You got {correct_answers} questions correct.")

DISPLAYING THE NUMBER OF CORRECT ANSWERS

```
Which option is a lie?
    (1) The platypus has 3 stomaches
    (2) Dolphins sleep with one eye open
1
That's correct! 'The platypus has 3 stomaches' is a lie!
You get a point added to your score!
You got 1 questions correct.
```

TODAY, WE CREATED A SMALL QUIZ APP BY COMPLETING THE FOLLOWING STEPS:

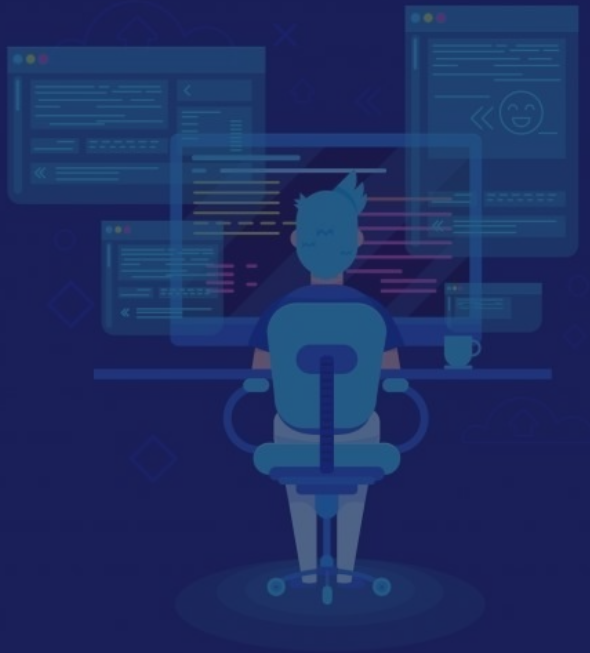
Step 01: Display Options And Get The User's Choice.

Step 02: Use Conditional Statements To Check The User's Choice.

Step 03: Track The Number Of Correct Answers

BONUS: 1

Shorthand Increment/Decrement Syntax



SHORTHAND INCREMENT OPERATORS

You can use `+=` to **`increment`** a value stored in a variable.

```
tacos_eaten = 0

tacos_eaten += 1
print(tacos_eaten) # 1

tacos_eaten += 1
print(tacos_eaten) # 2

tacos_eaten += 1
print(tacos_eaten) # 3
```

=

```
tacos_eaten = 0

tacos_eaten = tacos_eaten + 1
print(tacos_eaten) # 1

tacos_eaten = tacos_eaten + 1
print(tacos_eaten) # 2

tacos_eaten = tacos_eaten + 1
print(tacos_eaten) # 3
```

SHORTHAND DECREMENT OPERATORS

You can use `-=` to **`decrement`** a value stored in a variable.

```
tacos_eaten = 3

tacos_eaten -= 1
print(tacos_eaten) # 2

tacos_eaten -= 1
print(tacos_eaten) # 1

tacos_eaten -= 1
print(tacos_eaten) # 0
```

=

```
tacos_eaten = 3

tacos_eaten = tacos_eaten - 1
print(tacos_eaten) # 2

tacos_eaten = tacos_eaten - 1
print(tacos_eaten) # 1

tacos_eaten = tacos_eaten - 1
print(tacos_eaten) # 0
```

CHEAT SHEET

Step 1

In the file **index.py**,

- ❏ Refactor **correct_answers** increment/decrement to use shorthand operators.

index.py

```
...
if user_choice == 1:
    ...
    print("You get a point added to your score!")
    correct_answers ___= 1
...
else:
    ...
    print("You lose 1 point for that transgression!")
    correct_answers ___= 1
```

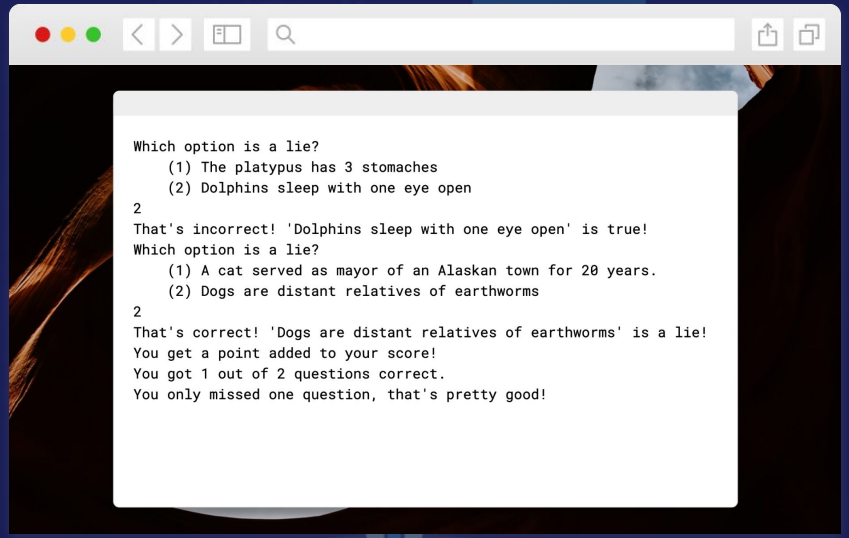
SOLUTION

index.py

```
...
if user_choice == 1:
    ...
    print("You get a point added to your score!")
    correct_answers += 1
...
else:
    ...
    print("You lose 1 point for that transgression!")
    correct_answers -= 1
```

HOMEWORK: 1.1

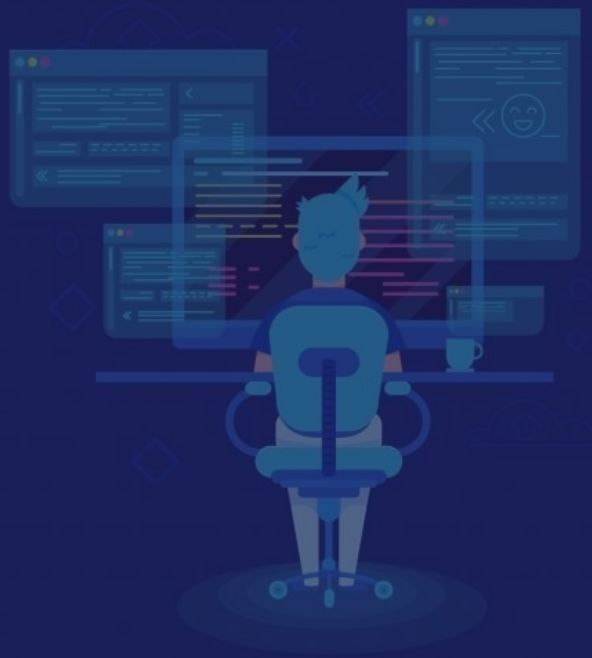
Add another question to your quiz



In your **`quiz-app/index.py`** file,

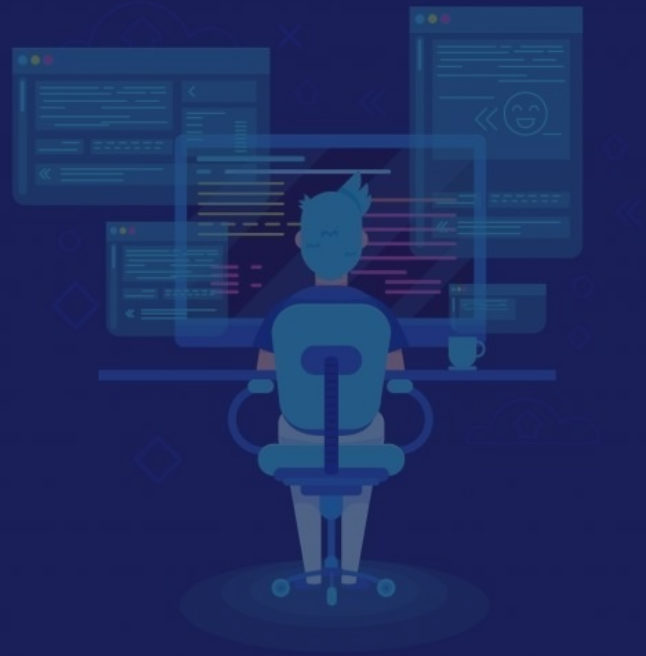
Follow the same steps we did in class to add another question to your quiz.

- ❏ Create two options; 1 should be **`true`** and the other should be a **`lie`**.
- ❏ Store these in a **`multi-line`** string.
- ❏ Ask the user which option is a lie and store their response.
- ❏ Use **`if...elif...else`** conditional statements to determine if the user guessed which option was a lie.
    - ❏ Increment and decrement the **`correct_answers`** variable in the appropriate place



HOMEWORK: 1.2

Print different responses based on the quiz results.

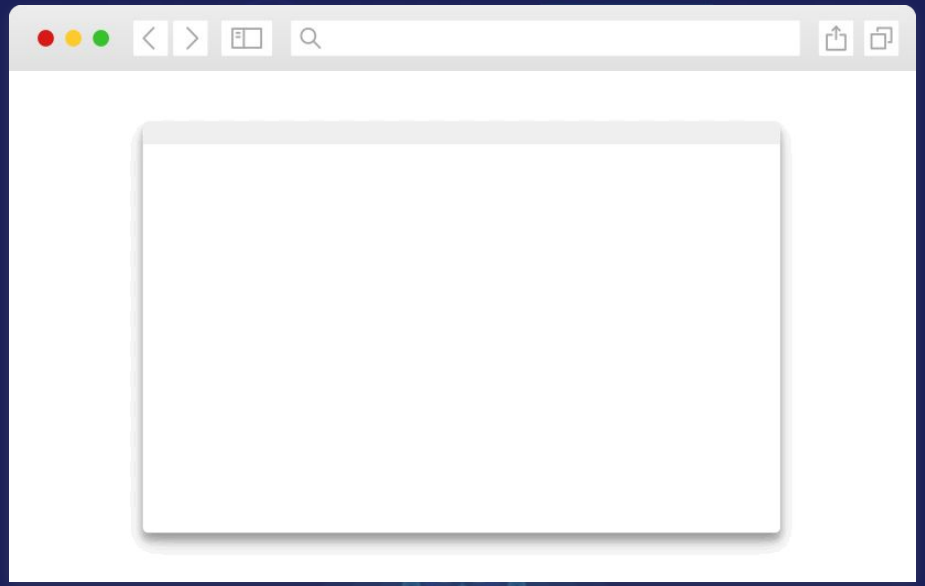


In your `quiz-app/index.py` file,

Print three different responses:

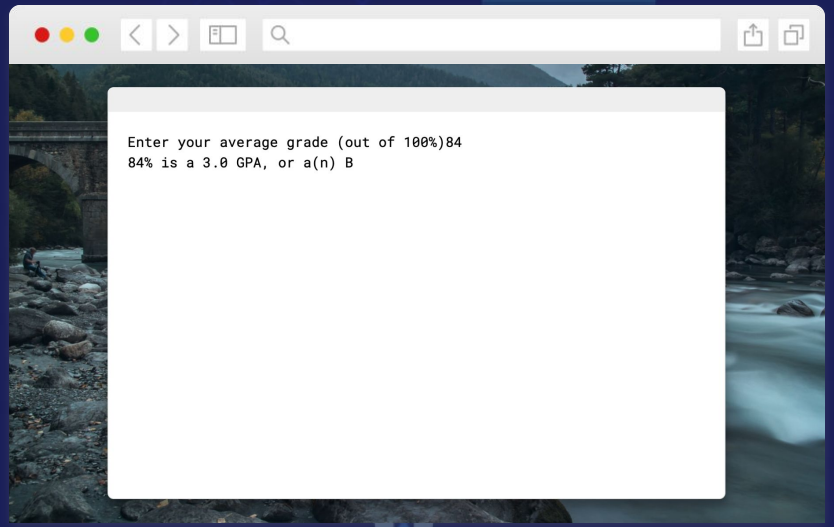
1. If they got a perfect score
2. If they missed only one question
3. If they missed more than one question

- ❏ You should keep track of the total number of questions in a variable and use that variable in a conditional statement to determine which response to print.



HOMEWORK: 2

Create a GPA calculator



In your `gpa-calculator/index.py` file,

- Prompt the user for their average grade and print their GPA and letter grade based on the given scale.
- You should have a long chain of if...elif statements.
- Order matters! While writing your conditional statements make sure to check them ascending or descending order (i.e. check if it's a 4.0 first, then 3.7, then 3.3... or 0.0, then 1.0, then 1.3). Otherwise, a result might trigger too early/late!

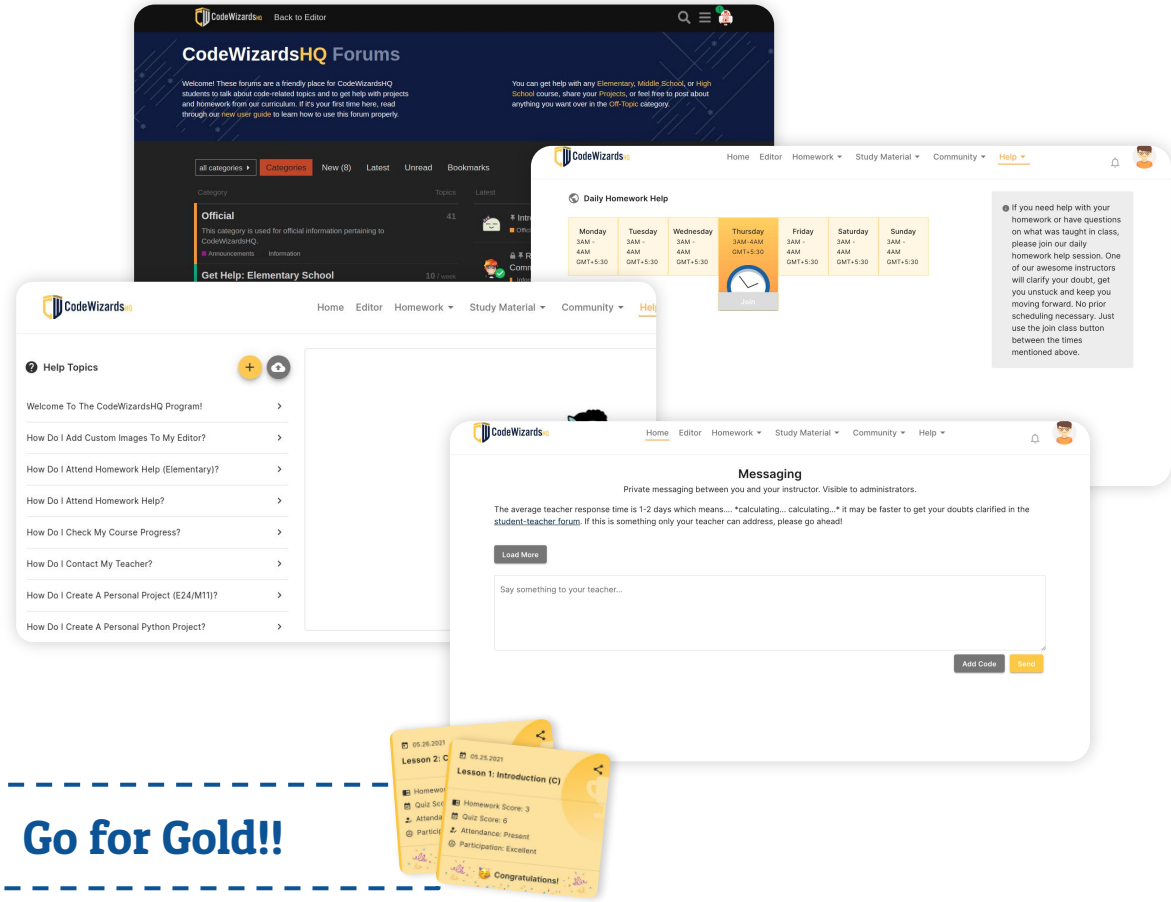
Score	GPA	Grade
94 and up	4.0	A
90 to 94	3.7	A-
87 to 90	3.3	B+
83 to 87	3.0	B
80 to 83	2.7	B-
77 to 80	2.3	C+
73 to 77	2.0	C
70 to 73	1.7	C-
67 to 70	1.3	D+
60 to 67	1.0	D
Less than 60	0.0	F

WHERE CAN I GET HELP?

- Slides and Recordings
- Forum
- Homework Help
- Messaging

Reminder:

You need at least 80% to pass.



Go for Gold!!

WHAT'S NEXT

LESSON : 5

Making Decisions: Part II