[Jahangirnagar University](https://www.coursera.org/programs/jahangirnagar-university-na-qyc9d)

[Skip to Main Content](https://www.coursera.org/learn/python-crash-course/quiz/QrKqf/practice-quiz-conditionals/attempt?redirectToCover=true#main)

* 

MD. TAZEL HOSSAN

[Crash Course on Python](https://www.coursera.org/learn/python-crash-course/home/welcome)

[Week 2](https://www.coursera.org/learn/python-crash-course/home/week/2)

Practice Quiz: Conditionals

[Prev](https://www.coursera.org/learn/python-crash-course/supplement/R9diu/conditionals-cheat-sheet)

[Next](https://www.coursera.org/learn/python-crash-course/lecture/nxhLW/basic-syntax-wrap-up)

**PRACTICE QUIZ • 25 MIN**

**Practice Quiz: Conditionals**

**Submit your assignment**

Try again

**Receive grade**

**TO PASS**80% or higher

**Grade**

80%

View Feedback

We keep your highest score

Practice Quiz: Conditionals

Practice Quiz • 25 min

**Congratulations! You passed!**

**TO PASS**80% or higher

Keep Learning

**GRADE**

80%

**Practice Quiz: Conditionals**

**TOTAL POINTS 5**

1.Question 1

What's the value of this Python expression: (2\*\*2) == 4?

**1 / 1 point**



4



2\*\*2



True



False

**Correct**

You nailed it! The conditional operator == checks if two values are equal. The result of that operation is a boolean: either True or False.

2.Question 2

Complete the script by filling in the missing parts. The function receives a name, then returns a greeting based on whether or not that name is "Taylor".

**1 / 1 point**

1

2

3

4

5

6

7

8

def greeting(name):

  if name == "Taylor":

    return "Welcome back Taylor!"

  else:

    return "Hello there, " + name

print(greeting("Taylor"))

print(greeting("John"))





RunReset

**Correct**

Great work! You're getting the hang of conditionals in Python.

3.Question 3

What’s the output of this code if number equals 10?

1

2

3

4

5

6

7

8

if number > 11:

  print(0)

elif number != 10:

  print(1)

elif number >= 20 or number < 12:

  print(2)

else:

  print(3)





**1 / 1 point**

2

**Correct**

Right on! Our number is 10, which is smaller than 12, so it matches that condition.

4.Question 4

Is "A dog" smaller or larger than "A mouse"? Is 9999+8888 smaller or larger than 100\*100? Replace the plus sign in the following code to let Python check it for you and then answer.

1

2

print("A dog" < "A mouse")

print(9999+8888 > 100\*100)





RunReset

**1 / 1 point**



"A dog" is larger than "A mouse" and 9999+8888 is larger than 100\*100



"A dog" is smaller than "A mouse" and 9999+8888 is larger than 100\*100



"A dog" is larger than "A mouse" and 9999+8888 is smaller than 100\*100



"A dog" is smaller than "A mouse" and 9999+8888 is smaller than 100\*100

**Correct**

You got it! Keep getting Python to do the work for you.

5.Question 5

If a filesystem has a block size of 4096 bytes, this means that a file comprised of only one byte will still use 4096 bytes of storage. A file made up of 4097 bytes will use 4096\*2=8192 bytes of storage. Knowing this, can you fill in the gaps in the calculate\_storage function below, which calculates the total number of bytes needed to store a file of a given size?

**0 / 1 point**

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

def calculate\_storage(filesize):

    block\_size = 4096

    # Use floor division to calculate how many blocks are fully occupied

    full\_blocks = 8192//block\_size

    # Use the modulo operator to check whether there's any remainder

    partial\_block\_remainder = filesize%block\_size

    # Depending on whether there's a remainder or not, return

    # the total number of bytes required to allocate enough blocks

    # to store your data.

    if partial\_block\_remainder >0:

        return 8192

    return block\_size

print(calculate\_storage(1))    # Should be 4096

print(calculate\_storage(4096)) # Should be 4096

print(calculate\_storage(4097)) # Should be 8192

print(calculate\_storage(6000)) # Should be 8192





RunReset

**Incorrect**

Not quite. One concept to keep in mind is the difference

between float division (/) and floor division (//). Floor

division rounds down to the nearest whole number, which is

useful for this question. Use filesize // block\_size to get

the amount of full blocks. Something else to remember here

is the format of modulo arithmetic. The divisor is always to

the right of the modulo operator. E.g.- 2 = 6 % 4 <- This is

the divisor.