# Control Statements

## Exercises

### Week 3

Prior to attempting these exercises ensure you have read the lecture notes and/or viewed the video, and followed the practical. You may wish to use the Python interpreter in interactive mode to help work out the solutions to some of the questions.

Download and store this document within your own filespace, so the contents can be edited. You will be able to refer to it during the test in Week 6.

Enter your answers directly into the highlighted boxes.

For more information about the module delivery, assessment and feedback please refer to the module within the MyBeckett portal.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

## ©2021 Mark Dixon / Tony Jenkins

What is the *data-type* of the result when evaluating comparison (relational) expressions such as < and >?

*Answer:*

When evaluating comparison (relational) expressions such as < and >, it always results in a

value with a Boolean data-type (that is, either True or False).

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

For each of the following expressions write the result of their evaluation.

100 < 101

*Answer:*

The result of this expression is ‘true’ because 100 is indeed less than 101.

100 > 99

*Answer:*

The result of this expression is ‘true’ because 100 is indeed greater than 99.

100 >= 100

*Answer:*

The result of this expression is ‘true’ because 100 is equal to 100, and the >= operator includes equality.

100 != 100

*Answer:*

The result of this expression is ‘false’ because 100 is equal to 100, and the ‘!=’operator evaluates to ‘false’ when the values on both sides are equal.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

For each of the following expressions write the result of their evaluation.

"abc" < "xyz"

*Answer:*

The result of the expression is ‘true’ because, in lexicographic order, "abc" comes before "xyz."

"abc" < "XYZ"

*Answer:*

The result of this specific expression would be ‘false’ because in lexicographic order, uppercase letters generally come before lowercase letters. Therefore, "abc" is considered greater than "XYZ" in this context.

"100" == 100

*Answer:*

False

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

For each of the following expressions write the result of their evaluation.

10 > 20 and 10 >= 10

*Answer:*

The result of the expression 10 > 20 and 10 >=10 is ‘false’.

10 > 30 > 20

*Answer:*

The result of the expression 10 > 30 > 20 is ‘false’.

40 < 20 or 20 < 30

*Answer:*

The result of the expression 40 > 20 or 20< 30 is ‘false’.

not True

*Answer:*

The result of the expression not true is false

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

What would be the output shown following the execution of the following Python statements?

colours = [ "Blue", "Black", "Orange" ]

print("The colour black is in the list : ", "Black" in colours)

*Answer:*

The colour black is in the list : True.

print("The colour orange is in the list : ", "orange" in colours)

*Answer:*

The colour orange is in the list : False

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Which of the following concepts does the Python ‘if’ statement support?

**Sequence**, **Selection** or **Iteration**?

*Answer:*

Python if statement support ‘selection’ concepts.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

What would be the output shown following the execution of the following Python statements?

num1 = 100

num2 = 10

if num1 % num2 == 0:

print("num1 is divisible by num2")

else:

print("num1 is not divisible by num2")

*Answer:*

The output shown following the execution of the following python statement is ‘num1 is divisible by num2’.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

What would be the output shown following the execution of the following Python statements?

num1 = 99

num2 = 70

if num1 < num2:

print("num1 is less than num2")

elif num1 > num2:

print("num1 is greater than num2")

else:

print("num1 is equal to num2")

*Answer:*

The output shown following the execution of the following Python statements ‘num1 is greater than num2’.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

What is the name given to the following type of Python operator shown below?

lowest = x if x < y else y

*Answer:*

The name given to the following type of Python operator shown below is ‘The Ternary Operator’

And, what value would be assigned to the variable ‘lowest’ when ‘x’ was equal to 10 and ‘y’ was equal to 5?

*Answer:*

5 would be assigned to the variable ‘lowest’

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Within the answer box below write a small Python program, that asks the user to enter a value between 1 and 10.

Once the value has been input display a message saying whether the value was in the requested range.

Remember: values returned from the **input()** function are *strings*, and need converting before being used within expressions, i.e. you will need code such as this -

num = input("please enter a number between 1 and 10 : ")

num = int(num)

*Answer:*

num = input("please enter a number between 1 and 10 : ")

num = int ( num)

if 1 <= num <= 10:

print(f"The entered number {num} is in the requested range.")

else:

print(f"The entered number {num} is not in the requested range.")

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Within the answer box below write a small Python program that asks the user to enter two values. Store these in variables called x and y respectively.

If the 'x' value is larger than 'y' then print

The value 'x' is larger than the value 'y'

otherwise print

The value 'y' is larger than the value 'x'

*Answer:*

x = (input("Enter the value for x: "))

y = (input("Enter the value for y: "))

if x > y:

print(f "The value {x} is larger than the value {y}.")

else:

print(f "The value {y} is larger than the value {x}.")

Examine the output generated by the above program. Is the displayed text entirely accurate in all cases? If not Why?

*Answer:*

No, the displayed text is not entirely accurate in all cases. The input values for ‘x’ and ‘y’ are stored as strings, and the program performs a string comparison using the ‘>’ operator. This can lead to unexpected results when comparing numerical values as strings. (i.e., when comparing ‘5’ and ‘10’ it shows “The value 5 is greater than the value 10”).

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Within the answer box below write a small Python program that asks the user to enter two values.

Store these values in two variables then output a message displaying the result of dividing the first value by the second value.

Include code that prevents a run-time error being reported when the user inputs a value of '0' for the second input. *Hint:* use an ‘if’ statement

If a '0' value is input, print a message saying "division by 0 is not possible".

*Answer*

The result displaying the result of dividing the first value by the second value is

x = int (input (" Enter the value for x: "))

y = int (input (" Enter the value for y: "))

if y ! = 0:

result = x / y

print (f "The result of {x} divided by {y} is: {result}")

else:

print ("division by 0 is not possible.")

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Which of the following concepts does the Python while statement support?

**Sequence**, **Selection** or **Iteration**?

*Answer:*

The Python while statement support ‘Iteration’

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

What would be the output shown following the execution of the following Python statements?

num = 5

while num > 0:

print(num)

num -= 1

*Answer:*

The output shown following the execution of the following Python statements is

5

4

3

2

1

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Write a small Python program that prints your name to the screen 100 times, then enter the program into the answer box below. Hint: use a ‘while’ loop.

*Answer:*

A small Python program that prints your name to the screen 100 times is

x = 0

while x < 100:

print("Mridu")

x += 1

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

What would be the output shown following the execution of the following Python statements?

vals = ["A", "B", "C", "D"]

for letter in vals:

print(letter)

*Answer:*

The output shown following the execution of the following Python statements is

A

B

C

D

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

What would be the output shown following the execution of each of the following Python statements?

for num in range(5):

print(num)

*Answer:*

The output shown following the execution of each of the following Python statements is

0

1

2

3

4

for num in range(10,16):

print(num)

*Answer:*

The output shown following the execution of each of the following Python statements is

10

11

12

13

14

15

for num in range(0,10,-1):

print(num)

*Answer:*

This Python statement does not produce any output because the range cannot generate a sequence with a negative step.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Enter and execute the python code shown below, then show the exact output into the answer box.

for x in range(1,10):

for y in range (1,x):

print("\*")

print()

*Answer:*

\*

\*

\*

\*

\*

\*

\*

\*

\*

\*

\*

\*

\*

\*

\*

\*

\*

\*

\*

\*

\*

\*

\*

\*

\*

\*

\*

\*

\*

\*

\*

\*

\*

\*

\*

\*

What is the term used to refer to code blocks that appear inside other code blocks as in the above program?

*Answer:*

The term used to refer to code blocks that appear inside other code blocks as in the above program ‘Nested code blocks in a nested loop’.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

## **Exercises are complete**

Save this logbook with your answers. Then ask your tutor to check your responses to each question.