Variables and Types

Exercises

Week 2

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.

Which is the purpose of a *variable* within Python?

Answer:

> The purpose of a variable within python is to store value and give data to computer for processing.

Write a simple Python statement that creates and assigns a value of `3.142` to a variable called `'pi'`

Answer:

> pi= 3.142

Which of the following is **NOT** a valid name for a variable within Python?

```
total
```

```
result
```

```
question?
```

```
name_1
```

Answer:

> question?

Following the execution of the code below, what will be stored in the variable 'age'?

```
age = 10 + 20
age = age + 5
```

Answer:

> 35

In the answer box below write the *exact* output that would be displayed if the following statement was executed (assuming `age` has been created as in the previous question):

```
print("The age value is",age)
```

Answer:

> The age value is 35

Which of the following is an example of an **Augmented Assignment** in Python?

```
total = 20

total = total + 5

total *= 100

total  = max
```

Answer:

> total*=100

Which of the following is an example of an **integer** type variable?

```
result = "xyz"

result = 20

result = 20.5

result = False
```

Answer:

> result = 20

What are the only two legal values of a **boolean** type variable?

Answer:

> true and false

Following the execution of the code below, what will be the *data-type* of the variable 'average'?

```
average = total / count
```

Answer:

> float

Following the execution of the code below, what will be the *data-type* of the variable 'message'?

```
message = "hello there!"
```

Answer:

> string

What determines the current data-type of a variable?

Answer:

> type() determines the current data-type of a variable

What is the purpose of the built-in `type()` function?

Answer:

> to return the type of data stored in the objects or variables in the program.

What would be the output following execution of the following code?

```
type(10.2)
```

Answer:

> float

Does the Python language support *Dynamic Typing*, or *Static Typing*?

Answer:

> Dynamic Typing

Which of the following is an example of a *function call*?

```
answer = 10

print(answer)

total *= 10

10 + 20
```

Answer:

> print(answer)

What is the name given to the values that are passed to a function within the parentheses?

Answer:

> arguments

What is the purpose of the built-in `input()` function?

Answer:

> The purpose of built-in input() function is to prompt the user for data and return it as a string.

What is the data-type of the value returned by the `input()` function?

Answer:

> string

Use the Python interpreter to input a small Python program that prints your name and address on the screen. Once this works type the program in the answer box below.

Answer:

```
a= input("Enter your name:")
b= input("Enter your address:")
 print(a)
 print(b)
```

Within the answer box below write a small Python program, that when run, would print the following message including the double quotes -

```
Hello, is your name "Bwian"?
```

Answer:

> ```
> print('"Hello, is your name "Bwian"?"')
> ```

Now write a second small Python program, that when run, would print the following message including the single quotes -

```
Or is your name 'Woger'?
```

Answer:

> ```
> print("Or is your name 'Woger'?")
> ```

Within the answer box below write a small Python program, that when run, uses *escape sequences* to print the following text exactly.

```
This is a string containing a backslash (\),
    a single quote ('), a double quote (")
    and is split across multiple lines
```

Answer:

```
print("This is a string containing a backslash (\),
a single quote ('), a double quote (\")
and is split across multiple lines")
```

_____

Within the answer box below write a small Python program, that when run, uses *triple quotes* to print the following text exactly.

```
This is a string containing a backslash (\),
    a single quote ('), a double quote (")
    and is split across multiple lines
```

Answer:

```
print("'This is a string containing a backslash (\),
    a single quote ('), a double quote (")
    and is split across multiple line"')
```

_____

Use the Python interpreter to input a small Python program that asks the user to input a temperature in fahrenheit. Once the value has been input, display a message that shows the same temperature in celsius. You may have to do some research in order to find out the conversion method. Once this works, type the program in the answer box below.

Answer:

```
temp= float(input("Enter a temperature in fahrenheit:"))
celsius = (temp - 32) * 5/9
print("The temperature in celcius is",celsius)
```

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

Answer:

```
a= input("Enter first value:")
b= input("Enter second value:")
print(a)
print(b)
```

Once the values have been input use three calls to the `print()` function to show output such as the following (in this example the user entered *10.2* and *18.3*) -

```
The value 'a' was 10.2 and the value 'b' was 18.3

The sum of 'a' and 'b' is 28.5

The product of 'a' and 'b' is 186.66
```

Answer:

```
a,b = float(input("Enter first value: ")), float(input("Enter Second Value: "))
print("The value 'a' was" , a , "and the value 'b' was" ,b)
print("The sum of 'a' and 'b' is, a+b)
print("The product of 'a' and 'b' is", a*b)
```

Python includes a built-in function called **max()**. When this is called with multiple argument values it returns the largest of the given arguments. e.g.

```
max(20, 50, 30)  # this would return 50
```

Within the answer box below write a small program that asks the user to input three values. Store these in variables (the names are up to you) then use the **max()** function to display the largest of the input values.

Answer:

```
first= input("Enter first value:")
second= input("Enter second value:")
third= input("Enter third value:")
x= max(first, second, third)
print(x)
```

Using the Python interpreter execute your code, then examine the output generated when the input the values are 'hello', 'welcome', and 'bye'

Does the program still show the maximum value? If not, what does it show?

Answer:

Yes, it shows the maximum value as 'welcome'.

Given the following definition:

```
name = "Black Knight"
```

What would each of the following Python statements display?

```
print( name[0] )
```

Answer:

B

```
print( name[4] )
```

Answer:

k

```
print( name[-1] )
```

Answer:

t

```
print( name[-2] )
```

Answer:

h

```
print( name[2:5] )
```

Answer:

ack

```
print( name[6:] )
```

Answer:

knight

```
print( name[:5] )
```

Answer:

Black

```
print( name[:] )
```

Which of the following creates a variable containing a **List**?

```
names = "Terry"

names = 10

names = [ "Mark", "Jon", "Amanda", "Edward", "Sally" ]

names = "Mark", "Jon", "Amanda"
```

Answer:

> names = ["Mark", "Jon", "Amanda", "Edward", "Sally"]

Is the following a valid **List**, even though it contains values based on different data-types?

```
values = [10.2, "Jon", False, "Edward", True ]
```

Answer:

> Yes

If a value is **mutable**, can it be modified after it has been created?

Answer:

> Yes

What term is used to describe a value that cannot be changed once it has been created?

Answer:

immutable

Is a List **mutable** or **immutable**?

Answer:

mutable

Is a String **mutable** or **immutable**?

Answer:

immutable

Given the following definition -

```
names = ["Terry", "John", "Michael", "Eric", "Terry", "Graham"]
```

What would each of the following Python statements display?

```
print( names[2] )
```

Answer:

Michael

```
print( names[-2] )
```

Answer:

Terry

```
print( names[0:3] )
```

Answer:

'Terry', 'John', 'Michael'

```
names = names + "Brian"
print( names )
```

Answer:

Cannot concatenate string to list

```
names[0:1] = ["Mark", "Jon"]          Text
print( names )
```

Answer:

['Mark', 'Jon', 'John', 'Michael', 'Eric', 'Terry', 'Graham']

What built-in function within Python can be used to find out how many elements are contained within a string or list?

Answer:

function len()

Exercises are complete

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