

Fundamentals of Programming:

Programming for Data Science with Python

1. Identifiers

Overview

- Identifiers are the names that identify the elements of a program such as classes, methods, variables, constants, etc.
 - Identifiers can be variables or constants in a program.
 - Identifiers or names are case sensitive.
 - They can contain letters, digits (or numbers), and underscore.
 - However, they cannot start with digits or numbers.

****IMPORTANT NOTES:****

- Reserved words cannot be used as identifiers.
- Here is the list of some reserved words in Python:

```
and, assert, break, class, continue, def, del, elid, else, except, exec, finally, for,  
from, global, if, import, in, is, lambda, not, or, pass, print, raise, return, try,  
while
```

****Run the following code:****

```
In [38]: # x is the name of a variable. x is an identifier.  
x = 3  
  
print ("x is a variable. It is an identifier. Its value is: ",x, "\n")  
print("Data type of x: ", type (x), '\n')  
  
#stdName is the name of a variable that represents the name of a student.  
#stdName is an identifier  
  
stdName = "DeAundrie Howard"  
  
print ("stdName is a variable. It is an identifier. Its value is: ", stdName, "\n")  
print("Data type of stdName: ", type (stdName), '\n')
```

x is a variable. It is an identifier. Its value is: 3

Data type of x: <class 'int'>

stdName is a variable. It is an identifier. Its value is: DeAundrie Howard

Data type of stdName: <class 'str'>

```
In [39]: # fName, lName, age is the name of a variable. These are also known as an identifier.
fName = 'DeAundrie'
lName = 'Howard'
age = 33

print ("fName, lName, and Age are variablee. It is an identifier. My first name is: ",
print("Data type of my first name is: ", type (fName), '\n')
print ("My last name is: ",lName, "\n")
print("Data type of my first name is: ", type (lName), '\n')

#fName and lName is the name of a variable that represents the first and last name of
#stdName is an identifier

print ("age is a variable. It is an identifier. Age value is: ", age, "\n")
print("Data type of age: ", type (age), '\n')
```

fName, lName, and Age are variablee. It is an identifier. My first name is: DeAundrie

Data type of my first name is: <class 'str'>

My last name is: Howard

Data type of my first name is: <class 'str'>

age is a variable. It is an identifier. Age value is: 33

Data type of age: <class 'int'>

```
In [40]: # x is the name of a variable. x is an identifier.
x = 3

print ("x is a variable. It is an identifier. Its value is: ",x, "\n")
print("Data type of x: ", type (x), '\n')

#stdName is the name of a variable that represents the name of a student.
#stdName is an identifier

stdName = "John Smith"

print ("stdName is a variable. It is an identifier. Its value is: ", stdName, "\n")
print("Data type of stdName: ", type (stdName), '\n')
```

x is a variable. It is an identifier. Its value is: 3

Data type of x: <class 'int'>

stdName is a variable. It is an identifier. Its value is: John Smith

Data type of stdName: <class 'str'>

```
In [41]: # x is the name of a variable. x is an identifier.
color = "red"

print ("x is a variable. It is an identifier. Its value is: ",color, "\n")
print("Data type of x: ", type (color), '\n')

#stdName is the name of a variable that represents the name of a student.
#stdName is an identifier

fav_Artist = "John Givens"

print ("fav_Artist is a variable. It is an identifier. Its value is: ", fav_Artist, "\n")
print("Data type of stdName: ", type (fav_Artist), '\n')
```

x is a variable. It is an identifier. Its value is: red

Data type of x: <class 'str'>

fav_Artist is a variable. It is an identifier. Its value is: John Givens

Data type of stdName: <class 'str'>

1.1 Variables: Data - Data Containers - Data Types

Variables are data containers whose values are likely to be changed along the course of executing a program.

Variable/Identifier x:

- x is an identifier. It is a **variable**.
- its value is 3: 3 is **data**
- The variable (identifier) x is the **data container** that contains the piece of data "3".
- **Data type** of x is "int"

****Run the following code:****

```
In [42]: # x is a name of a variable. x is an identifier.
x = 3
print ("x is a variable. It is an identifier. Its value is: ", x, "\n")
print("Data type of x: ", type(x), '\n')
```

x is a variable. It is an identifier. Its value is: 3

Data type of x: <class 'int'>

```
In [43]: # x is a name of a variable. x is an identifier.
pi_abbrev = 3.14
```

```
print ("pi is a variable. It is an identifier. Its value is: ", pi_abbrev, "\n")
print("Data type of x: ", type(pi_abbrev), '\n')
```

pi is a variable. It is an identifier. Its value is: 3.14

Data type of x: <class 'float'>

Variable/Identifier stdName:

- **stdName** is an **identifier**. It is a **variable**.
- Its **value** is "John Smith": "JohN Smith" is **data**.
- The **variable (identifier)** stdName is the **data container** that contains the piece of data "John Smith".
- **Data** type of stdName is "str" (or String)

IMPORTANT NOTES:

- In Python, identifiers are unlimited in length. Case is significant.
- However, the user is strongly discouraged from using a very long name to label variables. Besides, the names should be meaningful.

****Run the following code:****

```
In [44]: #stdName is the name of a variable that represents the name of a student.
#stdName is an identifier
std = "John Smith"
print ("stdName is a variable. It is an identifier. Its value is: ", stdName, "\n")
print("Data type of stdName: ", type (stdName), '\n')
```

stdName is a variable. It is an identifier. Its value is: John Smith

Data type of stdName: <class 'str'>

```
In [45]: #stdName is the name of a variable that represents the name of a student.
#stdName is an identifier
dts = "htimS nhoJ"
print ("dts is a variable. It is an identifier. Its value is: ", dts, "\n")
print("Data type of stdName: ", type (dts), '\n')
```

dts is a variable. It is an identifier. Its value is: htimS nhoJ

Data type of stdName: <class 'str'>

1.2. Constants

Constants are data containers that contain permanent values, i.e., these values cannot be changed, along the course of executing a program.

2. Assignments

Overview

Binding a variable in Python means setting a name to hold a **reference** to some object.

- Assignments creates references, not copies.

Names in Python do not have an intrinsic type. **Objects have types.**

- Python determines the **type of the reference** automatically based on the **data object** assigned to it.

A name is created when it appears the first time on the left side of an assignment expression:

```
x = 3
```

A **reference is deleted** via garbage collection when NO variables or identifiers are referring to it.

2.1. Reference Sematic in Python

Overview: The process of assigning a value

What happens when we type: `x = 3`?

- First, an integer 3 is created and stored in memory
- Then a name is created
- Next, a reference to the memory location storing the 3 is assigned to the name x

2.2 Assignment manipulates references

- `y = x`: An assignment of x to y
- `y = x`: The assignemnt does not make a copy of the value of x.
- `y = x`: The assignment makes y refer to the same object as x does.

```
Let's say you assign x = 3
```

```
And then you assign y = x
```

When you assign `y = x`, you now have `y = 3` since `x = 3` (Run the code below.)

****Run the following code:****

```
In [46]: x=3  
y=x
```

```
y
```

```
Out[46]: 3
```

```
In [47]: x=3
          y=x
          z=y
          z
```

```
Out[47]: 3
```

3. Operators and Operations

3.1 Numeric Operations

Operator	Operation	Example	Result
+	Addition	33 + 3	36
-	Subtraction	33 - 3	30
*	Multiplication	33 * 3	99
/	Division	11 / 3	3.666
%	Remainder	33 % 3	0
**	Exponent	33 ** 3	35937
//	Floor	11 // 3	3

3.2 Augmented Operators

Operator	Operation	Example	Result
+=	Addition assignment	i += 8	i = i + 8
-=	Subtraction assignment	i -= 8	i = i - 8
*=	Multiplication assignment	i *= 8	i = i * 8
/=	Division assignment	i /= 8	i = i / 8
%=	Remainder assignment	i %= 8	i = i % 8

Operator	Operation	Example	Result
<code>**=</code>	Exponent assignment	<code>i **= 8</code>	<code>i = i ** 8</code>
<code>//=</code>	Floor assignment	<code>i //= 8</code>	<code>i = i // 8</code>

3.3 Relational Operators (a.k.a. Comparison Operators)

- Relational operators, a.k.a. comparison operators, are used for comparisons.
- A comparison, e.g. `a > b`, is called a conditional expression or boolean expression.
- The result of a boolean expression is either `True` or `False`.

Operator	Meaning	Example	Ex. Result
<code>(<)</code>	Less than	<code>radius (<) 0</code>	<code>False</code>
<code>(<=)</code>	Less than or equal to	<code>radius (<=) 0</code>	<code>True</code>
<code>(>)</code>	Greater than	<code>radius (>) 0</code>	<code>True</code>
<code>(>=)</code>	Greater than or equal to	<code>radius (>=) 0</code>	<code>False</code>
<code>(==)</code>	Equal to	<code>radius (==) 0</code>	<code>True</code>
<code>(!=)</code>	Not equal to	<code>radius (!=) 0</code>	<code>False</code>

3.4 Logical Operators

Operator	Meaning	Example
<code>not</code>	Opposite	<code>not (radius < 0)</code>
<code>and</code>	And	<code>(a > b) and (c < d)</code>
<code>or</code>	Or	<code>(a > b) or (c < d)</code>

3.4.1 `not` Operators

X	<code>not X</code>
<code>True</code>	<code>False</code>
<code>False</code>	<code>True</code>

3.4.2 `and` Operators

`a and b` **is true** only when ***both a and b are true***.

a	b	a and b
True	True	True
True	False	False
False	True	False
False	False	False

3.4.2 or Operators

a or b *is false* only when *both a and b are false*.

a	b	a or b
True	True	True
True	False	True
False	True	True
False	False	False

3.5 Identity Operators

In Python, *identity operators* are used to check if the *operands are identical*, i.e., they refer to the same object.

Operator	Result	Example
is	True if the operands are identical, i.e., they refer to the same object;	x is y
is not	True if the operands are not identical, i.e., they do not refer to the same object	x is not y

****Run the following 3 code blocks:****

```
In [48]: x = 5
y = 5

isTrue = "x is y"
isFalse = "x is not y"

if (x is y):
    print (isTrue)
else:
    print (isFalse)
```


x is y

```
In [49]: fav_TvShow = "Demon Slayer"

isTrue = "Go Tanjiro and Nezuko!"
isFalse = "Oh, no! It's Muzan Kibutsuji"

if (fav_TvShow == "Demon Slayer"):
    print (isTrue)
else:
    print (isFalse)
```

Go Tanjiro and Nezuko!

```
In [50]: x = 5
y = 8

isTrue = "x is y"
isFalse = "x is not y"

if (x is y):
    print (isTrue)
else:
    print (isFalse)
```

x is not y

```
In [51]: fName = "DeAundrie"
lName = "Howard"

isTrue = "x is y"
isFalse = "x is not y"

if (fName is lName):
    print (isTrue)
else:
    print (isFalse)
```

x is not y

```
In [52]: x = 5
y = 5

isTrue = "x is y"
isFalse = "x is not y"

if (x is not y):
    print (isTrue)
else:
    print (isFalse)
```

x is not y

```
In [53]: x = 5
y = 5

isTrue = "True. x is " + str(x) + " and y is " + str(y)
isFalse = "False. x is " + str(x) + " and y is " + str(y)
```

```
if (x is not y):
    print (isTrue)
else:
    print (isFalse)
```

False. x is 5 and y is 5

3.6 Membership Operators

In Python, many data structures have their internal structure of a sequence, e.g., String, List, Tuple, etc.

For example:

- aString = "Hello World"
- aList = [1,2,3,4,5]
- aTuple = (1, 2, 3, 4, 5)

Membership operators - in, not in - are used to test **if a value is found** in a sequence or not.

Operator	Result	Example
in	True if the value/variable is found in the sequence;	x in y
not in	True if the value/variable is not found in the sequence	x not in y

****Run the following 2 code blocks:****

```
In [54]: x = 'Hello World'
if ('H' in x):
    print ("H in x")
else:
    print ("H not in x")
```

H in x

```
In [55]: stdName = 'DeAundrie Howard'
if ('H' in x):
    print ("H in x")
else:
    print ("H not in x")
```

H in x

```
In [56]: # You will get an error but think why you have an error.
alist = [1, 2, 3, 4, 5]
if (8 not in alist):
    print ("8 not in alist")
else:
    print ("8 in alist")
```

```
8 not in alist
```

```
In [57]: # You will get an error but think why you have an error.
alist = ["a", "b", "c", "d"]
if ("a" not in alist):
    print ("a not in alist")
else:
    print ("a in alist")
```

```
a in alist
```

5. Pseudo-Code

5.1 Scenario: A Problem

It is assumed that a software developer is asked to write a Python program that can calculate and print the diameter and the circumference of a circle. The user enters the data of the radius and its measurement unit (in, ft, cm, or m) from the console.

5.2 How to Solve the problem

- First, we need to write pseudo-code (or create a flow-chart) of steps that can solve the problem. Formally, we design an algorithm that offers a solution to the problem.
- Then, based on the steps of the pseudo-code/flowchart (or algorithm), we write the code of the program.

These two steps are the foundation of software engineering process.

5.3 Pseudo-Code

With pseudo-code, the developer writes down the steps to solve the problem in plain English.

1. Start
2. Read the input of the radius from the console
3. Read the measurement unit of the radius (in, ft, cm, m)
4. Calculate the diameter of the circle
 - $\text{diameter} = 2 \times \text{radius}$
5. Calculate the circumference of the circle
 - $\text{Circumference} = \text{diameter} \times \text{PI} (3.14159)$
6. Print out the diameter
7. Print out the circumference
8. End

```
In [58]: PI = 3.14
#r = float(input("Enter the radius of a circle: "))
print ("Enter the radius of a circle: ")
r = float(input())
```

```
diameter = 2*r
circumference = 2 * PI * r
area = PI *r * r

print ("The diameter of a circle with a radius of " + str(r) + " is:", diameter)
print ("The circumference of a circle with radius of " + str(r) + " is:", circumference)
print ("The area of the circle with radius of " + str(r) + " is:", area )
```

Enter the radius of a circle:

5

The diameter of a circle with a radius of 5.0 is: 10.0

The circumference of a circle with radius of 5.0 is: 31.400000000000002

The area of the circle with radius of 5.0 is: 78.5

```
In [59]: # BMI Calculator

print ("Please enter weight in pounds (lbs)")
wgt = float(input())
print ("Please enter height in inches (inches):")
hgt = float(input())

#BMI = wgt * 703 / (hgt ** 2)

BMI = wgt * 703 / (hgt ** 2)
print(BMI)
```

Please enter weight in pounds (lbs)

165

Please enter height in inches (inches):

66

26.62878787878788

6. Comments

In Python, comments begin with a hash mark (#), a whitespace character and continue to the end of the line.

****Run the following 4 code blocks:****

```
In [60]: # This is a comment
x = 3
x
```

Out[60]: 3

```
In [61]: # To comment, or not to comment. That is the question.
y = "why"
y
```

Out[61]: 'why'

```
In [62]: # Print "Hello World!" to console
print ("Hello, World")
```

Hello, World

```
In [63]: # Print "Goodbye World" to console
print ("Goodbye, World")
```

Goodbye, World

```
In [64]: # Inline comment
x = 3 # This is an inline comment
x
```

Out[64]: 3

```
In [65]: # Inline comment
comment = "out of line" # This is an inline comment ... that is out of line.
comment
```

Out[65]: 'out of line'

```
In [66]: # This is a comment of multiple lines, also known as a block comment
# Add another line of comment
# Another line
# And another line
x = 3
x
```

Out[66]: 3

```
In [67]: # This is a comment of multiple lines, also known as a block comment
# Typically, this lines of comments...
# Goes at the beginning of project or a section of code...
# Explaining what is to follow. See below.

# This is the end of the Part 1 of the Python Intro.
# Below is a quick recap of everything that was Learne in this part.
# Enjoy!

# Here's a we go!
fName = "DeAundrie"
lName = "Howard"
age = 33
fav_Nums = [3, 7, 8, 23]

if (age <= 18):
    print ("You are not an adult")
elif (19 <= age <=25):
    print ("You are a young adult")
elif (26 <= age <= 40):
    print ("Welcome the pains of adulthood")
elif (40 <= age < 65):
    print ("You are getting up there. Be careful.")
else:
    print("You are old!")

if (2 not in fav_Nums):
    print ("2 is not in your favorite numbers.")
else:
    print ("2 is in your favorite number.")
```

```
if ("A" in fName):  
    print ("A is in your first name")  
else:  
    print ("A is not in your first name")
```

Welcome the pains of adulthood
2 is not in your favorite numbers.
A is in your first name
