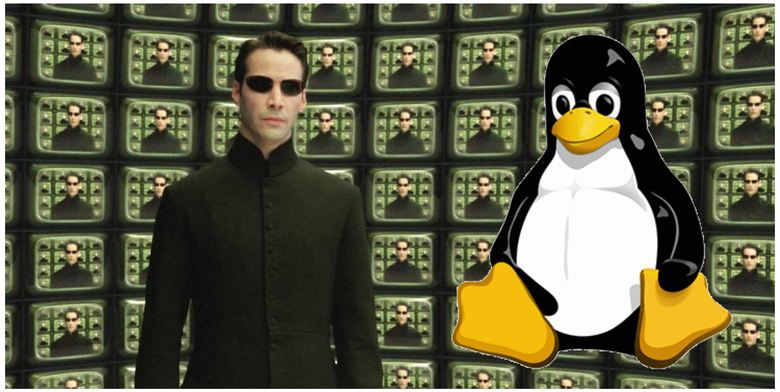
14.09.21 20:18

CSCI 120
Introduction to Computer Science and Programming I
Lecture 1: Your First Python Programme II



Jetic Gū

Overview

- Focus: Basic C/C++ Syntax
- Architecture: Linux/Unix OS
- Core Ideas:
  1. Variables
  2. `input()` function
  3. `if` condition

P0
Recap

`print()` function

- `print("...")`
  prints whatever is inside the quotes to `stdout`, and add a newline (`"\n"`)

- `print("...", end="\n")`
  equivalent to above

- `print("...", end="")`
  prints whatever is inside the quotes, and don't automatically add a newline (`"\n"`)

- `print("...\n", end="")`
  prints whatever is inside the quotes, and add a newline (`"\n"`)

Review

Variables

Variables

- Computers deal with data
- Data needs to be stored in Memory before we can manipulate them
- We use variables to denote these data, just like in math

Concept

Variables

```
a = 10
b = 10
c = a + b
```

- Variable declaration
  - start with the variable name, then followed by =, then the value
  - the value here can be an expression, such as mathematical calculation

Concept

Variable Names

```
this_is_a_valid_name = 1
thisIsAlsoValid2 = 2
variable3 = 3
```

- A variable name must start with a lowercase letter or the underscore character
  - Names starting with uppercase letter is by convention for `Class` names, not recommended for variables
- A variable name cannot start with a number
- A variable name can only contain alpha-numeric characters and underscores (`A-z`, `0-9`, and `_`)
- Variable names are case-sensitive (`agE`, `age` and `aGE` are three different variables)

Concept

P1
Variables

Spot Illegal Variable Names

```
# Legal variable names:
myvar = "Jetic"
my_var = "Jetic"
_my_var = "Jetic"
myVar = "Jetic"
MYVAR = "Jetic"
myvar2 = "Jetic"

# Illegal variable names:
2myvar = "Jetic"
my-var = "Jetic"
my var = "Jetic"
```

Concept

1. https://www.w3schools.com/python/gloss_python_variable_names.asp

Variables

```
a = 10          CPU exe.
b = 10
c = a + b
```

CPU: This equals to 20

Memory Table

Address	Variable	Type	Value
Slot 0	a	int	10
Slot 1	b	int	10
Slot 2	c	int	20
....			

- Memory is like a table
  - Variables are references for you to access memory locations
  - When you declare a new variable, a new memory slot is `allocated` for you
  - You must declare it first, before you can use a variable!

Example

Python Variable Types

- Python types are called `classes`
- `type` function
  For checking types of classes
- `int` class
  Integers, converts automatically to float when seeing fractions
- `float` class
  Fractions
- `str` class
  Strings

```
>>> type(10)
<class 'int'>
>>> type(10.1)
<class 'float'>
>>> type("Hello World!")
<class 'str'>
```

Example

1. You will see this screenshot in `interactive` mode, also known as the Python console

Printing Variables

```
a = "This is a string"
b = 10
print(a)
# prints variable a, then "\n"
print(b)
# prints variable b, then "\n"
print(a, b)
# This will print variable a and b, separated by " ", then "\n"
```

- You can print the variable's values to stdout using `print`

Example

Printing Variables

```
print(a, b)
# Separated by " "
print(a, b, sep=" ")
# Same as above, this is the default value
print(a, b, sep="")
# No separation
print(a, b, sep="-")
# Separated by "-"
```

- You can change the separation string in `print` separation using the `sep` option. By default it is `sep=" "`

Example

Printing Variables

```
print(a, b, sep="-", end="\n")
# This works
print(a, b, end="\n", sep="-")
# This also works
```

- `sep` and `end` are optional arguments of the `print()` function
- Python does not force the order of optional arguments for a function, as long as they are at the end (certain languages do force the order!)

Example

P1
Variables

Getting Values from `stdin`

```
var = input()
var2 = input()
var3 = input()
var4 = input()
var4 = int(var4)
```

```
>>> var = input()
12345
>>> print(var)
12345
>>> var2 = input()
Hello world!
>>> print(var, type(var))
12345 <class 'str'>
>>> var3 = input()
10.234
>>> print(var, type(var))
12345 <class 'str'>
>>>
```

- `stdin` by default, is what you enter in the console
- When you press `enter`, what you typed gets sent over to `stdin`, which you can assign to variables through the `input()` function
- `input()` function always return in `str` type
  to convert to `int`/`float`, use the `int(...)` or `float(...)` function

Example

Getting Values from `stdin`

```
var = int(input("Enter an integer"))
# This is equivalent to the following two instructions

print("Enter an integer", end="")
var = int(input())
```

- `input()` can also take an argument, a `str` to be printed to `stdout` before prompting for input from `stdin`

Example

P2
Exercise

Lab 0:
A plus B Simple