Discovery Piscine

Module5 - Python

Summary: *In this Module, we see how to use arrays and their associated functions.*

Version: 1.00

Contents

Chapter I

A word about this Discovery Piscine

Welcome !
You will begin a Module of this Discovery Piscine of computer programming. Our goal is to introduce you to the code behind the software you use daily and immerse you in peer learning, the educational model of 42.

Programming is about logic, not mathematics. It gives you basic building blocks that you can assemble in countless ways. There is no single "correct" solution to a problem—your solution will be unique, just as each of your peers' solutions will be.

Fast or slow, elegant or messy, as long as it works, that's what matters! These building blocks will form a sequence of instructions (for calculations, displays, etc.) that the computer will execute in the order you design.

Instead of providing you with a course where each problem has only one solution, we place you in a peer-learning environment. You'll search for elements that could help you tackle your challenge, refine them through testing and experimentation, and ultimately create your own program. Discuss with others, share your perspectives, come up with new ideas together, and test everything yourself to ensure it works.

Peer evaluation is a key opportunity to discover alternative approaches and spot potential issues in your program that you may have missed (consider how frustrating a program crash can be). Each reviewer will approach your work differently—like clients with varying expectations—giving you fresh perspectives. You may even form connections for future collaborations.

By the end of this Piscine, your journey will be unique. You will have tackled different challenges, validated different projects, and chosen different paths than others—and that's perfectly fine! This is both a collective and individual experience, and everyone will gain something from it.

Good luck to all; we hope you enjoy this journey of discovery.

Chapter II

Introduction

What this Module will show you:

- You will learn how to handle arrays and their associated functions.

Chapter III

General instructions

Unless otherwise specified, the following rules apply every day of this Piscine.

- This document is the only trusted source. Do not rely on rumors.

- This document may be updated up to one hour before the submission deadline.

- Assignments must be completed in the specified order. Later assignments will not be evaluated unless all previous ones are completed correctly.

- Pay close attention to the access rights of your files and folders.

- Your assignments will be evaluated by your fellow Piscine peers.

- All shell assignments must run using `/bin/bash`.

- You <u>must not</u> leave any file in your submission workspace other than those explicitly requested by the assignments.

- Have a question? Ask your neighbor on your left. If not, try your neighbor on your right.

- Every technical answer you need can be found in the `man` pages or online.

- Remember to use the Piscine forum of your intranet and Slack!

- Read the examples thoroughly, as they may reveal requirements that aren't immediately obvious in the assignment description.

- By Thor, by Odin! Use your brain!!!

Chapter IV

Exercise 00: create_array

	Exercise 00
	Create an array
Turn-in directory : *ex00/*	
Files to turn in : `create_array.py`	
Allowed functions : `All`	

- Create a program called `create_array.py`.

- Ensure this program is executable.

- Define an array of numbers.

- Display the array on the screen:

```
?> ./create_array.py | cat -e
[2, 8, 9, 48, 8, 22, -12, 2]$
?>
```

Print.

Chapter V

Exercise 01: play__with__arrays

	Exercise 01
	Manipulating arrays
Turn-in directory : *ex01/*	
Files to turn in : `play_with_arrays.py`	
Allowed functions : `All`	

- Create a program called `play_with_arrays.py`.

- Ensure this program is executable.

- Define an array of numbers.

- Iterate over this array, creating a new array by adding 2 to each value in the original array.

- Your program should contain two arrays: the original array and the modified array.

- Display both arrays on the screen.

- For example, if the original array is `[2, 8, 9, 48, 8, 22, -12, 2]`, the output should be:

```
?> ./play_with_arrays.py
Original array: [2, 8, 9, 48, 8, 22, -12, 2]
New array: [4, 10, 11, 50, 10, 24, -10, 4]
?>
```

Chapter VI

Exercise 02: play_with_arrays++

	Exercise 02
	Manipulating arrays, once again
Turn-in directory : *ex02/*	
Files to turn in : `play_with_arrays.py`	
Allowed functions : `All`	

- Modify your previous program to process only the values greater than 5 in the original array.

- For example, if your original array is `[2, 8, 9, 48, 8, 22, -12, 2]`, the output should be:

```
?> ./play_with_arrays.py | cat -e
[2, 8, 9, 48, 8, 22, -12, 2]$
[10, 11, 50, 10, 24]$
?>
```

Chapter VII

Exercise 03: play_with_arrays+=2

	Exercise 03
	Manipulating Arrays, Still!
Turn-in directory : *ex03/*	
Files to turn in : `play_with_arrays.py`	
Allowed functions : `All`	

- Modify your previous program to remove duplicates in the output. You should not explicitly remove values from your arrays.

- For example, if your original array is `[2, 8, 9, 48, 8, 22, -12, 2]`, the output should look like this:

```
?> ./play_with_arrays.py | cat -e
[2, 8, 9, 48, 8, 22, -12, 2]$
{10, 11, 50, 24}$
?>
```

Search about set.

Chapter VIII

Exercise 04: Parameters

	Exercise 04
	Yes, but how many?
Turn-in directory : *ex04/*	
Files to turn in : `parameters.py`	
Allowed functions : `All`	

- Create a program called `parameters.py`.

- Ensure this program is executable.

- The program should display the number of parameters passed to it, followed by a newline.

```
?> ./parameters.py
Number of parameters: 0.
?> ./parameters.py "initiation"
Number of parameters: 1.
?> ./parameters.py "this" "is" "crazy" "there's" "everywhere!"
Number of parameters: 5.
?>
```

Refer to the sys.argv list and the len() function.

Chapter IX

Submission and peer-evaluation

- You must have `discovery_piscine` folder at the root of your home directory.

- Inside the `discovery_piscine` folder, you must have a folder named `module5`.

- Inside the `module5` folder, you must have a folder for each exercise.

- Exercise 00 must be in the `ex00` folder, Exercise 01 in the `ex01` folder, etc.

- Each exercise folder must contain the files requested in the assignment.

> ⚠ Please note, during your defense anything that is not present in the folder for the module will not be checked.