TASK

Your First Computer Program

Visit our website

Introduction

Welcome to Your First Computer Program Task!

In this task, you are introduced to the Python programming language. Python is a widely used programming language. It is consistently ranked in the top 10 most popular programming languages as measured by the TIOBE Programming Community Index. Many familiar organizations make use of Python, such as *Wikipedia*, *Google*, *Yahoo!*, *NASA* and *Reddit* (which is written entirely in Python).

Python is a high-level programming language, along with other popular languages such as Java, C# and Ruby. High-level programming languages are closer to human languages than machine code. They're called "high-level" as they are several steps removed from the actual code that runs on a computer's processor.

This task is a gentle introduction to Python where you will be asked to create a simple program. In doing so, you will become familiar with the structure of a Python program.

Get in touch

Connect for support

Remember that with our courses, you're not alone! You can contact your mentor to get support on any aspect of your course.

The best way to get help is to login to **www.hyperiondev.com/portal** to start a chat with your mentor. You can also schedule a call or get support via email.

Your mentor is happy to offer you support that is tailored to your individual career or education needs. Do not hesitate to ask a question or for additional support!

A note from the Hyperion Team

Hope you're excited to start learning such a popular and fun programming language!

A brief history of Python:
Python is named after the Monty Python comedy group. It was created by 'Benevolent Dictator For Life' Guido van Rossum in 1991, who now works for Dropbox. At the time when Guido van Rossum began implementing the Python language, he was also reading the published scripts from *Monty Python's Flying Circus* (a BBC comedy series from the seventies). His inspiration for Python stemmed from the desire to create a simple scripting language and his experience with the ABC programming language.

Guido van Rossum

Why Python?

Python is a powerful, widely used programming language. Unlike Java, Python is a more recent, efficient and arguably faster programming language. The syntax (the way the code is written) is very similar to Java.

Here are a few more reasons to use Python:

- **Simple, yet powerful:** Looking at languages like C++ and Java can flummox and scare the beginner. But Python is intuitive with a natural way of presenting code. Python's succinctness and economy of language allows for speedy development and less hassle over useful tasks. This makes Python easy on the eyes and mind.

- **From child's play to big business:** While Python is simple enough to be learned quickly (even by kids), it is also powerful enough to drive many big businesses. Python is used by some of the biggest tech firms such as *Google*, *Yahoo!*, *Instagram*, *Spotify* and *Dropbox*, which should speak volumes about the job opportunities out there for Python developers**.**

- **Python is on the web:** Python is a very appealing language of choice for web development. Sites such as *Pinterest* and *Instagram* make use of the versatility, rapidity and simplicity of Django (a web development framework written in Python).

- **Even *Dropbox* was built using Python:** *Dropbox* must save massive amounts of files while supporting massive amounts of user growth. 99.9% of *Dropbox* code is written in Python! Using Python has helped *Dropbox* gain more than a hundred million users. Using only a few hundred lines of Python code, they were able to scale up massively in user numbers. Learn from *Dropbox* and use Python!

ZEN OF PYTHON

The Zen of Python, written in 1999 by Tim Peters, mentions all the software principles that influence the design of the Python language.

Beautiful is better than ugly.
Explicit is better than implicit.
Simple is better than complex.
Complex is better than complicated.
Flat is better than nested.
Sparse is better than dense.
Readability counts.
Special cases aren't special enough to break the rules.
Although practicality beats purity.
Errors should never pass silently.
Unless explicitly silenced.
In the face of ambiguity, refuse the temptation to guess.
There should be one — and preferably only one — obvious way to do it.
Although that way may not be obvious at first unless you're Dutch.
Now is better than never.
*Although never is often better than *right* now.*
If the implementation is hard to explain, it's a bad idea.
If the implementation is easy to explain, it may be a good idea.
Namespaces are one honking great idea — let's do more of those!

Ever need to recall these principles? Try entering this into your Python interpreter!
>> import this

A note from Masood...

Know this, Python is hot:

As of 2018, the demand for Python programmers is only growing. Python boasts the highest year-on-year increase in terms of demand by employers (as reflected in job descriptions put up online) as well as popularity among developers. Python developers are one of the highest-paid programmers! Also, the demand for Python is only set to grow further with its extensive use in analytics, data science and machine learning.

Installing Python IDLE

Before you get started, we strongly suggest you start using IDLE or Notepad++ to open all text files (.txt) and Python files (.py). Do not use the normal Windows notepad for reading code files.

IDLE stands for Integrated DeveLopment Environment. An Integrated Development Environment is the software that programmers use to write, debug and execute their code.

Your content folder contains a sub-folder named “Installers” which will help you download and set up Python on your machine. Once that is done, you can access the IDLE through the Start Menu on a Windows computer or by typing "idle" at the command line on a Mac or Linux computer.

What is Programming?

Programmers write statements of code to create **programs**. Programs are executable files that perform the instructions given by the programmer.

Code can be written in different programming **languages**, such as Python, Java and C++. In this course, you will start by learning Python.

After writing Python commands or code, save them in a Python file. A Python file has the following file naming format:

filename.py

Where filename can be any valid filename and .py is the file extension.

You can then 'run' the Python file. The Python program you have written is executed and displays the outcomes that may result based on what the code statements say. Information about how to 'run' Python files are given in the example file (**example.py**) that accompanies this task. We will now show you how to write some basic code in Python, and perform some basic operations.

The Print Function

You may want your program to display or output information to the user. The most common way to view program output is to use the *print* function. To use *print*, we enter the *print* command followed by one or more arguments. In programming, a **command** is an instruction given by a user telling a computer to do something. **Arguments** can be thought of as the values on which you want the command to act. Together a command and an argument are known as a **statement**. Consider the Python statement below:

```
print ("Hello, World!")
```

When you run this program, the computer will output the argument “Hello, World!” Note that the argument is enclosed in double quotes ("..."). This is because "Hello, World!" is a string or a list of characters, but this will be discussed in more detail in later tasks.

The Python Shell (the window that is displayed when you run a Python program) only shows the output of the program. Other statements in your code will be executed but not displayed in the Python Shell.

Syntax rules

All programming languages have **syntax** rules. Syntax is the "spelling and grammar rules" of a programming language and determines how you write correct, well-formed statements.

A common syntax error you could make above is by forgetting to add a closing quotation mark ("). Remember that all opening quotation marks (") require a closing one! Another common syntax error that you could make above is by forgetting to add a closing bracket ‘)’. Remember that all opening brackets ‘(’, require a matching closing one!

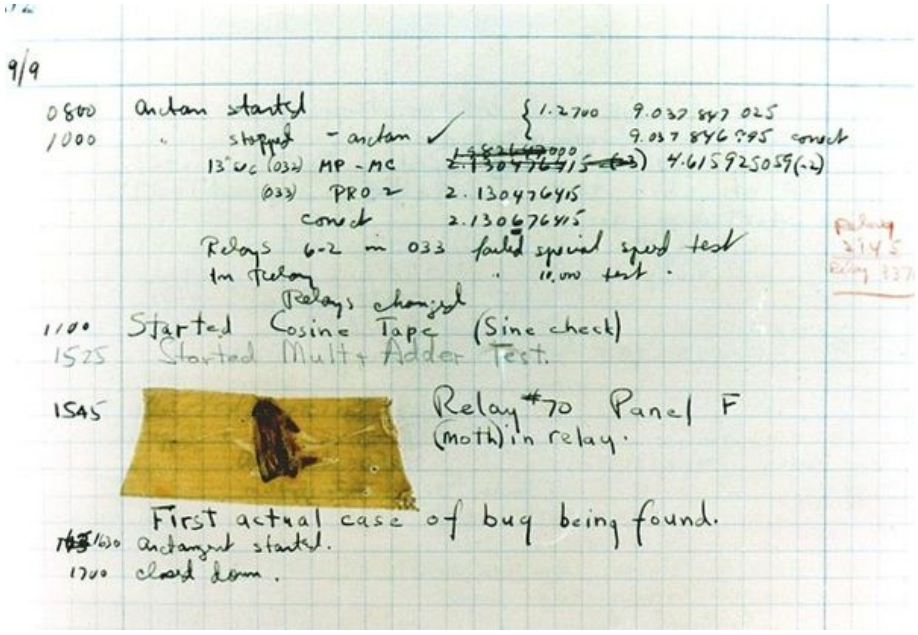
Any program you write must be **exactly** correct. All code is case sensitive. This means that *‘Print’* is not the same as *‘print’*. If you enter an invalid Python command, misspell a command or misplace a punctuation mark, you will get a syntax error when trying to run your Python program.

Errors appear in the Python shell when you try to run a program and it fails. Be sure to read all errors carefully to discover what the problem is. Error reports in the Python shell will even tell you what line of your program had an error. The process of resolving errors in code is known as **debugging**.

A note from Riaz...

Sorry to interrupt, but did you know that the first computer “bug” was named after a real bug? Yes, you read that right! While the term “bug” in the meaning of a technical error was first coined by Thomas Edison in 1878, it was only 60 years later that someone else popularised the term.

In 1947, Grace Hopper, a US Navy admiral, recorded the first computer ‘bug’ in her logbook as she was working on a Mark II computer. A moth was discovered stuck in a relay and thus hindering the operation. She proceeded to remove the moth, thereby ‘debugging’ the system, and taped it in her logbook. In her notes, she wrote, “First actual case of bug being found.”



- **Riaz Moola,** *Founder and CEO*

How To Get Input

Sometimes you want a user to enter data that will be used by your program through the keyboard. To do this, use the *input* command.

The *input* command, in the example below, will show the text "Enter your name: " in the output box of the program. The program will then halt until the user enters something with their keyboard and presses enter.

```
name = input("Enter your name: ")
```

The variable *name* stores what the user entered into the box as a string. Storing and declaring variables doesn't produce any output.

Instructions

This lesson is continued in the **example.py** file provided in this task folder. Open this file using IDLE or Notepad++ (right-click on the file and select 'Edit with IDLE').

The context and examples provided in **example.py** should help you understand some simple basics of Python.

You may run **example.py** to see the output. The instructions on how to do this are inside the file. Feel free to write and run your own example code before attempting the task, to become more comfortable with Python.

Compulsory Task 1

Follow these steps:

- Create a new Python file in this folder called **hello_world.py**
- Inside this file, write Python code to takes in a user's name using *input()* and then print out the name.
- Also, take in a user's age using the same method and print out their age.
- Finally, print out a new line and the string "Hello World!"

Compulsory Task 2

Follow these steps:

- Create a new Python file in this folder called **menu.py**
- Write a program that asks the user to order their 3 favourite food items on a menu using *input()*. Store them as variables *item1*, *item2* and *item3*.
- Print each item separately.
- E.g. your program could print:

  Order confirmation! You have ordered:

  Chicken nuggets

  Fish and chips

  Spaghetti bolognaise

Thing(s) to look out for:

1. Make sure that you have installed and setup all programs correctly. You have setup **Dropbox** correctly if you are reading this, but **Python or Notepad++** may not be installed correctly.
2. If you are not using Windows, please ask your mentor for alternative instructions.

Rate us

Share your thoughts

Hyperion strives to provide internationally-excellent course content that helps you achieve your learning outcomes.

Think the content of this task, or this course as a whole, can be improved or think we've done a good job?

Click here to share your thoughts anonymously.