Hyperion Development

Task: Defensive programming - Error Handling

www.hyperiondev.com

Introduction

Welcome to the Error Handling Task!

You should now be quite comfortable with basic variable identification, declaration, and implementation. You should also be familiar with the process of writing basic code which adheres to the correct Python formatting to create a running program. This task is aimed at furthering your knowledge of types of variables to create more functional programs. You'll also be exposed to error handling and basic debugging in order to fix issues in your code, as well as the code of others - a skill which is extremely useful in a software development career!

Connect with your mentor

CONNECT

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/support** 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!

Check out the numerical data types in Python, their sizes in bits, minimum and maximum values below.

Category	Types	Size (bits)	Minimum Value	Maximum Value	Example
Integer	`byte`	8	-128	127	`byte b = 65;`
	`char`	16	0	$2^{16}-1$	`char c = 'A';` `char c = 65;`
	`short`	16	-2^{15}	$2^{15}-1$	`short s = 65;`
	`int`	32	-2^{31}	$2^{31}-1$	`int i = 65;`
	`long`	64	-2^{63}	$2^{63}-1$	`long l = 65L;`
Floating-point	`float`	32	2^{-149}	$(2-2^{-23})\cdot 2^{127}$	`float f = 65f;`
	`double`	64	2^{-1074}	$(2-2^{-52})\cdot 2^{1023}$	`double d = 65.55;`
Other	`boolean`	1	--	--	`boolean b = true;`
	`void`	--	--	--	--

-The Hyperion Team

Dealing with Errors

Everyone makes mistakes. Maybe you've made some already. It's important to be able to DEBUG your code.The word DEBUG comes from a time when computers were as big as a room, and bugs literally landed in the machinery blocking electrical signals and causing errors.

Types of Errors

There are 3 different types of errors that can occur:

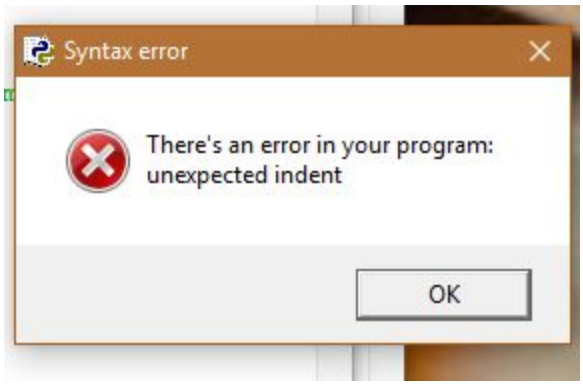
- **COMPILATION ERRORS:** When you try to compile your code the program will immediately crash. This is due to syntax errors, indentations etc.

- **RUNTIME ERRORS:** Your program compiles fine, but when it is actually run, an error occurs and the program stops! An error message will also pop up when trying to run it.

- **LOGICAL ERRORS:** Your program runs and compiles fine, but the output isn't what you're expecting. This means your logic applied to the problem as a solution is wrong.

Compilation Errors

Below is a typical example of a compilation error message. Compilation errors are the most common type of error in Python. They can get a little complicated to fix when dealing with loops and if statements.



When a compilation error occurs, the line in which the error is found will also be highlighted in red, and the cursor will automatically be put there so that error is easily found.

Go to the line indicated by the error message and correct the error, then try to compile your code again. DEBUGGING is a huge part of being a programmer - get used to it!

Runtime Errors

Say we had this line in our program: num = int("18.2")
Your code would compile properly, but when running you'd get the error:

```
ValueError: invalid literal for int() with base 10: '18.2'
```

Look carefully at the TYPE Of error. It must have something to do with the format of the number you gave num when trying to parse a String into an Integer.
You can't cast 18.2 to an Integer because integers don't accept decimal points.
It's important to think in a DEDUCTIVE way to solve runtime errors.

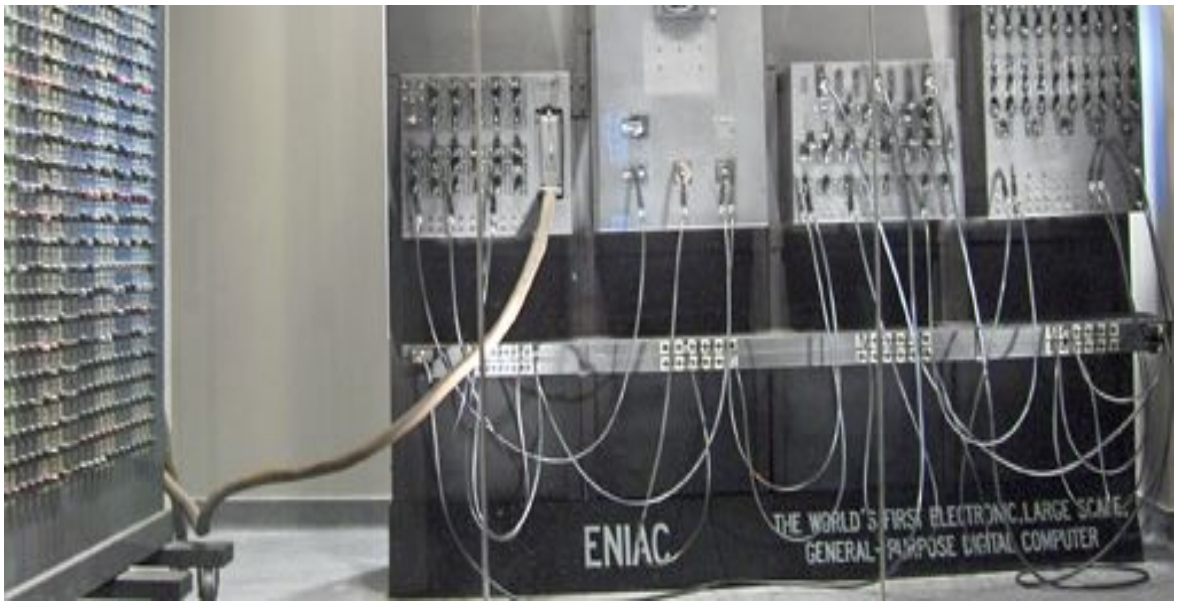
Logical Errors

When dealing with loops and control statements, these will become more frequent. Even after years of programming, it takes a long time to sit down and design an algorithm. Finding out why your program isn't working is more intuition than anything else.

Hey again, I'm sure you have heard the term debugging before? No, it's not an extermination service, well it is kind of, let me explain.

Debuvic - whoops - Debugging

The term 'Debugging' comes from when bugs caused problems in computers - this was only possible when computers were as big as rooms! One of the first computers ever was known as ENIAC. This computer was located at the University of Pennsylvania. Riaz Moola, the Founder of Hyperion Development, recently studied at the University of Pennsylvania where ENIAC is still on display. Below is a picture of an ENIAC.



The principle of "try, try again" is a practice which software developers live by as this leads to the very best code implementation. This is done by the procedure of "testing and debugging", whereby you'll develop code, and run it on numerous occasions to hunt for errors or lack of performance in your program, and thereby editing or adding code to achieve this end goal!

- **Masood Gool**, Online Trainer

Instructions

Before you get started we strongly suggest you start using Notepad++ or IDLE to open all text files (.txt) and python files (.py). Do not use the normal Windows notepad as it will be much harder to read.

First read example.py, open it using Notepad++ (Right click the file and select 'Edit with Notepad++') or IDLE.

- example.py should help you understand some simple Python. Every task will have example code to help you get started. Make sure you read all of example.py and try your best to understand.

- You may run example.py to see the output. Feel free to write and run your own example code before doing the Task to become more comfortable with Python.

- You are not required to read the entirety of Additional Reading.pdf, it is purely for extra reference.

Compulsory Task 1

Follow these steps:

- Open errors.py in your task folder.
- Attempt to run the program. You will encounter various errors.
- Fix the errors and then run the program.
- Save the file such that it runs correctly.
- Now run the program and notice the output - fix the error and indicate which of the three types of errors it was.
- Each time you fix an error, add a comment in the line you fixed it and indicate which of the three types of errors it was.

Compulsory Task 2

Follow these steps:

- Create a new Python file, called 'logic'.
- Write a program that displays a logical error (be as creative as possible!).

Things 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.

Give 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.