Codio Activity - Exploring Python tools and features

Part I

In this example, you will compile and run a program in C using the **Codio workspace** provided (Buffer Overflow in C). The program is already provided as bufoverflow.c - a simple program that creates a buffer and then asks you for a name, and prints it back out to the screen.

This is the code in bufoverflow.c (also available in the Codio workspace):

```
#include <stdio.h>
int main(int argc, char **argv)
{
char buf[8]; // buffer for eight characters
printf("enter name:");
gets(buf); // read from stdio (sensitive function!)
printf("%s\n", buf); // print out data stored in buf
return 0; // 0 as return value
{
```

Now compile and run the code. To test it, enter your first name (or at least the first 8 characters of it) you should get the output which is just your name repeated back to you.

Run the code a second time (from the command window this can be achieved by entering ./bufoverflow on the command line). This time, enter a string of 10 or more characters.

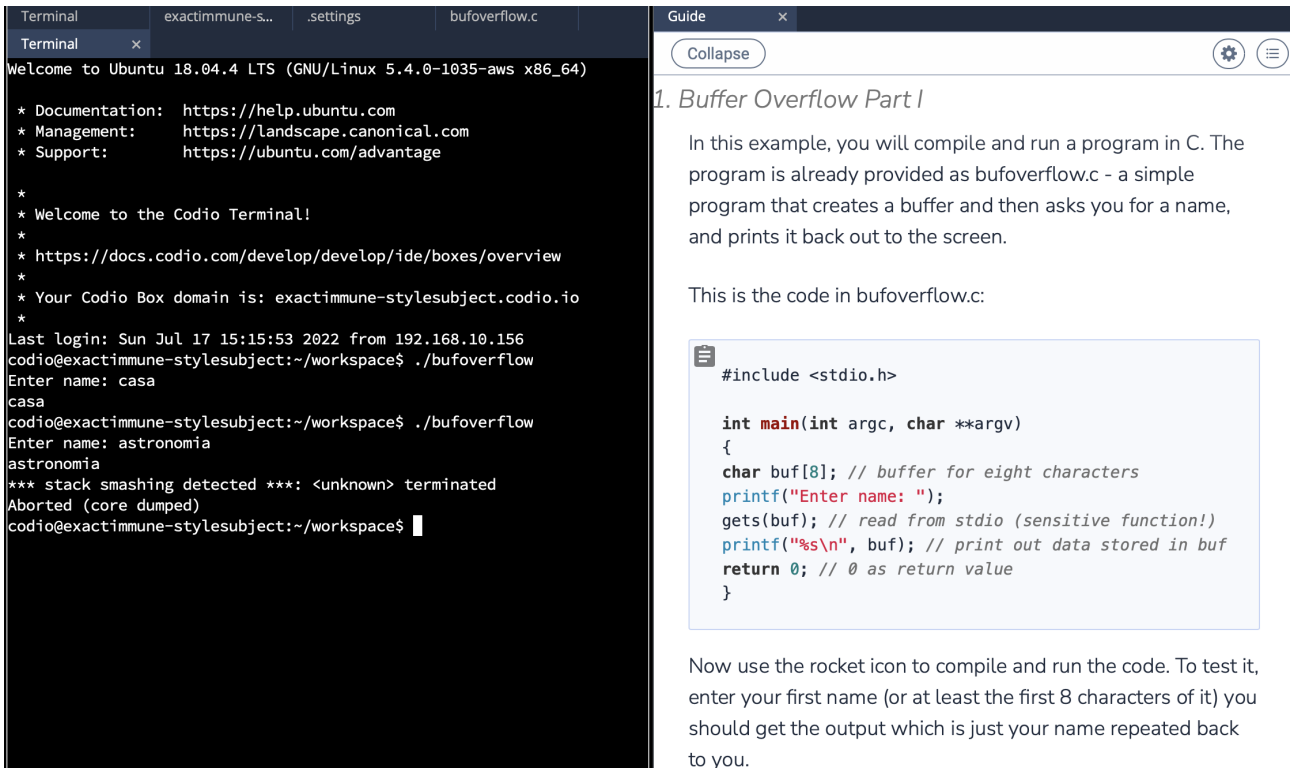
- What happens?

  If insert a word with 10 or more characters will appear ***** **stack smashing detected ***: <unknown> terminated**

  Aborted (core dumped)

- What does the output message mean?

  The output means that the user input a word exceeding the buffer capacity



Part II

Now carry out a comparison of this code with one in Python (Buffer Overflow in Python), following these instructions:

In the Codio workspace, you will be using the file called Overflow.py:

```
buffer=[None]*10
for i in range (0,11):
    buffer[i]=7
print(buffer)
```

- Run your code using: Python overflow.py (or use the codio rocket icon)
- What is the result?

Traceback (most recent call last):

File "Overflow.py", line 5, in <module>

buffer[i]=7

IndexError: list assignment index out of range

•Read about Pylint at http://pylint.pycqa.org/en/latest/tutorial.html

•Install pylint using the following commands:

pip install pylint (in the command shell/ interpreter)

•Run pylint on one of your files and evaluate the output:

pylint your_file

•(Make sure you are in the directory where your file is located before running Pylint)

•What is the result? Does this tell you how to fix the error above?

Python is a memory safe language and doesn't variables to reach regions that are not allocated