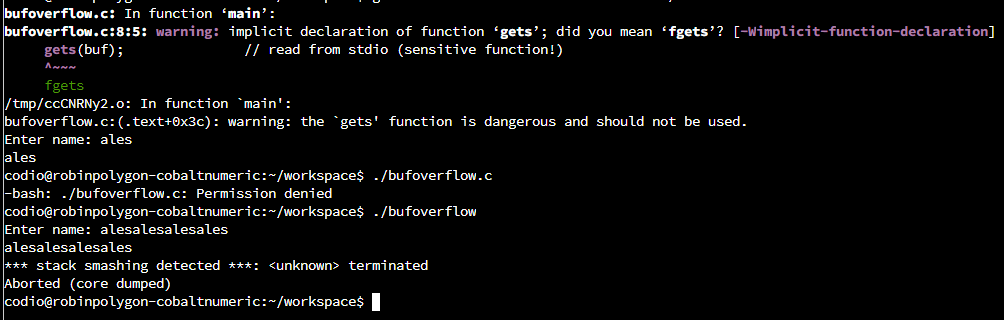
C program:  
  
#include <stdio.h>  
  
int main(int argc, char \*\*argv)  
{  
char buf[8]; // buffer for eight characters  
printf("Enter name: ");  
gets(buf); // read from stdio (sensitive function!)  
printf("%s\n", buf); // print out data stored in buf  
return 0; // 0 as return value  
}  
  


The output returns following message:  
\*\*\* stack smashing detected \*\*\*: <unknown> terminated  
Aborted (core dumped)

It means, that the storage capacity of the memory has been exceeded. In this particular example, we allocated a buffer to store up to 8 characters in the memory, however when I entered the string longer than that, the program returned above mentioned error. In this example, the extra characters that I typed in caused buffer overflow error, consequently that extra data was overflown into adjecent memory location. As a consequence, the data that had already been in memory on that location got overwritten and corrupted (Fortinet, n.d.).

In contrast to the above error in C program, Python handles these errors in different way. For example, this code:  
  
buffer=[None]\*10  
for i in range (0,11):  
buffer[i]=7  
print(buffer)  
  
Returns the following error:  
  
Traceback (most recent call last):  
  File "Overflow.py", line 3, in <module>  
    buffer[i]=7  
IndexError: list assignment index out of range  
  
after installing pylint and running the code again with pylint, the error is as follows:  
  
  
\*\*\*\*\*\*\*\*\*\*\*\*\* Module Overflow  
Overflow.py:4:0: C0303: Trailing whitespace (trailing-whitespace)  
Overflow.py:5:0: C0304: Final newline missing (missing-final-newline)  
Overflow.py:1:0: C0103: Module name "Overflow" doesn't conform to snake\_case naming style (invalid-name)  
Overflow.py:1:0: C0114: Missing module docstring (missing-module-docstring)  
  
-----------------------------------  
Your code has been rated at 0.00/10  
  
  
The results are the instructions of how to correct the code properly. It encourages a developer to follow best design patterns.

References:

Fortinet. (n.d.). Buffer Overflow. Fortiguard Labs Threat Intelligence. Available from: <https://www.fortinet.com/resources/cyberglossary/buffer-overflow> [Accessed 5 June 2023]