**Student Name:** **Student ID:**

# Objectives

* Write debugger code.
* Attach a debugged process to a debugger.
* Get CPU register state.
* Handle debug events.
* Use breakpoints.

# Important Information

* For *every* lab and home assignment, store all your work in your personal repository in a subdirectory named **mXX**, where XX is the module number. Carefully name the program as described in each problem.
* Your programs are extracted from your repository by a Python script. If there are any errors in the program name, then your instructor will never see your program, and you will receive a mark of zero.
* Push your work to the server often, and ensure that you push the final version of a program by the deadline specified, because the script extracting them can be run at any time after the deadline.

# Instructions

1. Read chapter 3 in the *Gray Hat Python* textbook. The following links are also useful:
   * <https://docs.python.org/3/library/pdb.html>
   * <http://www.gnu.org/software/gdb/documentation/>
   * <http://sourceware.org/gdb/current/onlinedocs/gdb.pdf.gz>
   * <https://docs.python.org/2/extending/extending.html>
2. Complete Problems 5, 6 and 7.

**Note:** Problem numbering continues from the module’s lab.

## Problem 5

1. Write a short Python script named **printfloop.py** that prints a line of text every two seconds, using the **printf()** msvcrt.dll library function. Ensure that the script first prints its own PID.
2. Create a file named **m06softbp.py** and an associated test harness named **m06softbp\_test.py**, again using the files from the previous problem.
3. Your program will prompt for PID and then display all debugging events as they arrive.
4. Your program will detach from the process and exit when the letter **Q** (case insensitive) is pressed. Use the **msvcrt** module to get the keystrokes from the keyboard.
5. Run the program and use the PID of the printfloop.py script. The test harness sets a soft breakpoint at the printf() address. The debugger reports the breakpoint, resets the breakpoint and continues.

## Problem 6

Repeat Problem 5, but this time create a file named **m06hardbp.py** and an associated test harness named **m06hardbp\_test.py**. The program’s functionality will be the same, but the breakpoint will be hard (e.g., use processor breakpoint registers).

## Problem 7

1. Create a file named **m06membreak.py** and an associated test harness named **m06membreak\_test.py**, again using the files from the previous problem.
2. Your program will prompt for PID and then display all debugging events as they arrive.
3. Your program will detach from the process and exit when the letter **Q** (case insensitive) is pressed. Use the **msvcrt** module to get the keystrokes from the keyboard.