Name:

|  |  |  |
| --- | --- | --- |
| Problem | Points | Score |
| 1 | 40 |  |
| 2 | 40 |  |
| 3 | 20 |  |
| Total | 100 |  |

Notes:

1. The first step in this exam is to create a workspace in the following directory:

/data/courses/ece\_1111/current/exams/ex\_01

Your directory should be your last name all lowercase, followed by an underscore, following by your first name (e.g, “picone\_joseph”). Set the permissions using “chmod u+rwx,g-rwx,o-rwx <lastname>” so only you have read and write permission to this directory. Create subdirectories within this directory: p01, p02, … You will use these for problems 1, 2, … respectively. Put ALL your code in these directories. Do not touch your files after the exam is over.

Failure to follow these instructions will result in a grade of 0. This preamble is part of the process of demonstrating you have basic Linux literacy.

1. Your code must be nicely formatted and well commented, or I will deduct at least 10 points per problem.
2. For this exam you are allowed to open a terminal window on your computer, you are allowed to web surf with Google, but you cannot use online chat or other interactive services.

**Problem No. 1**:

Write a shell script p01.sh (in the directory /p01) that does the following:

* Identifies all the processes in the ece-000 that are owned by the username specified as the first argument (e.g., “p01.sh <username>”).
* Displays the following information in a list that is sorted alphabetically in reverse order by the command name:

**PID %CPU Command**

Your output should look like this:

**ece-000\_[1]: p01.sh picone**

**PID: %CPU: Command:**

 **27 1.0 zebra.exe**

 **16 3.0 whois.exe**

 **35 4.5 abet.exe**

**Problem No. 2:**

Write a C program that generates integers in the range [0, 255] and outputs the bits that correspond to the binary coded decimal representation of the number. Your output should look like this:

**Integer: BCD Representation:**

**000 000000**

**001 000001**

**002 000010**

**003 000011**

**... ...**

**127 111111**

You must implement this using a for loop that iterates over the range [0, 255] – you can’t hardcode the output. You can assume that the integers are limited to an 8-bit range.

Place your code in /p02. You need to use a make file and your driver program name should be example.cc.

**Problem No. 3**:

This problem tests your ability to creatively solve problems using Google search and some rudimentary Linux commands. Write a single command line that factors a whole number and only produces output when the number is not divisible by 2. Put your command line in a file p03.txt in the directory /p03.