UNIVERSITY OF CALIFORNIA, SANTA CRUZ
BOARD OF STUDIES IN COMPUTER ENGINEERING

CMPE012/L: COMPUTER SYSTEMS AND ASSEMBLY PROGRAMMING

LAB 1: A SIMPLE PROGRAMMING ASSIGNMENT

MINIMUM SUBMISSION REQUIREMENTS:

- DeadBeef.XXX in the lab1 folder (where XXX is the extension for one of the approved programming languages listed below)
- Output.txt in the lab1 folder
- README.txt in the lab1 folder
- Commit and Push your repo
- A Google form submitted with the Commit ID taken from your GITLAB web interface verifying that the above files are correctly named in their correct folders
- All of the above must be completed by the lab due date

LAB OBJECTIVE:

This lab will verify that you have the prerequisite programming background by having you program a simple task in the programming language of your choice (see list below). This class has a prerequisite of programming experience; this should not be a hard task. You **WILL NOT BE** able to complete the lab if you start it the day it is due (this will be true of all subsequent labs as well).

If you spend more than an hour or two to program this lab, or need to spend time looking up solutions on the web, then you do not have the prerequisite knowledge to succeed in this class. If this is difficult for you, you will be better off taking an introductory programming class and coming back to CE12 when you are ready.

Be very careful about using code you find on the web. The code you turn in **MUST BE YOUR OWN**. It does not matter if you understand the code; if you did not create it, it isn't yours. This is an academic integrity violation pure and simple. Failure to meet the minimum submission requirements listed will result in failing the class (see syllabus). This will be true for all subsequent labs as well.

This program is to iterate through a set of integers (1-1000) and output one of four outputs depending on the number: if the number is divisible (no remainder) by 4, then the output is "DEAD." If the number is evenly divisible by 9, then the output is "BEEF." If the output is divisible by both 4 and 9, then the output should be "DEADBEEF." Lastly, if the output is not divisible by either 4 or 9, then the number itself should be printed. Each entry is on its own line, and you will be submitting both the code and the output.

This program is also something of a first non-trivial program beyond the "hello world" and similar programming assignments are used in hiring interviews all the time.

ACCEPTABLE PROGRAMMING LANGUAGES AND STANDARD FILE EXTENSIONS:

In this lab we are allowing you to use any programming language of your choice given that the prerequisite is programming experience (without specification of any specific programming language). In order to be reasonable we have limited this to the most popular languages summarized in the table below. The extension you put on the file in the repo will inform the grading script which language to invoke. Make sure that you match the file extension to the language that you are using.

Progamming Language	File extension
C	.c
C++	.cpp
Python	.py
Java	.java
Ruby	.rb
Go	.go
BASIC / FreeBasic	.bas
MATLAB	.m

If there is another language you need to use that is not on this list, contact the instructional staff via piazza as soon as possible, and we will see about extending the list. Note that this list includes almost all of the popularly used languages. Also if you know one of the more exotic ones, you probably can code this up in one of the more conventional programming languages.

LAB SPECIFICATIONS:

This program is to iterate through a set of integers (1-1000) and output one of four outputs depending on the number: if the number is divisible (no remainder) by 4, then the output is "DEAD." If the number is evenly divisible by 9, then the output is "BEEF." If the output is divisible by both 4 and 9, then the output should be "DEADBEEF." Lastly, if the output is not divisible by either 4 or 9, then the number itself should be printed. Each entry is on its own line, and you will be submitting both the code and the output.

- Program iterates through the integers 1 to 1000
- If the number is divisible by 4 output: DEAD
- If the number is divisible by 9 output: BEEF
- If the number is divisible by both 4 and 9 output: DEADBEEF
- If the number is NOT divisible by either 4 or 9 output the number itself

You will turn in your code in a file named DeadBeef.XXX where XXX is the correct file extension corresponding to the language you used (see table above). For example, if C were used, the file would be DeadBeef.c

You will turn in the output generated by your code in a plain text file named Output.txt

You will edit your README.txt file to include your name, CruzID, which programming language you used to code up the assignment, and any notes about your code that you wish to include. Each of these should be on a separate line.

NOTES:

This lab is quite short but if you are rusty in your programming it can take some time. Start early and there should be no issues with finishing it well before the due date. This lab is really a check on the prerequisite programming knowledge required to succeed in this class; if you cannot complete this easily, we strongly recommend you take an introductory programming class and come back another term when you are ready.

Ensure that you have met the minimum submission requirements. Failure to meet this requirement will result in failing the class regardless of any other scores. You can submit a new commit ID as many times as you like, we will always grade the last one you submit on the google form. Also note that until you push, the files don't exist in a place where we can get to them. This is how the program knows which files to pull from your repo. This is why we need the commit ID to match what you want to turn in.

One reminder about committing and pushing: You should get in the habit of committing early and often. This is very easy to do, and guarantees that you have something turned in. Every time you make a change, add a file, modify a file, go ahead and commit and push. You will always be able to return to that place and never lose your work.

SAMPLE OUTPUT:

This program will generate a file with 1000 lines in the output. You will need to capture that to a file for the Output.txt file that you will turn in. Each line in the file should be either a number, the text DEAD, BEEF, or DEADBEEF. The number does not appear on a line in which DEAD, BEEF, or DEADBEEF appear. The words should be in all capital letters. The output file should be plain text (no formatting).

Below is the output of the code run from the number 1 – 80 demonstrating what the output should look like (to prevent this lab from being several pages long, we've split it up into 3 columns):

1	DEAD	55
2	29	DEAD
3	30	57
DEAD	31	58
5	DEAD	59
6	33	DEAD
7	34	61
DEAD	35	62
BEEF	DEADBEEF	BEEF
10	37	DEAD
11	38	65
DEAD	39	66
13	DEAD	67
14	41	DEAD
15	42	69
DEAD	43	70
17	DEAD	71
BEEF	BEEF	DEADBEEF
19	46	73
DEAD	47	74
21	DEAD	75
22	49	DEAD
23	50	77
DEAD	51	78
25	DEAD	79
26	53	DEAD
BEEF	BEEF	

Lastly, just to show you the pattern you should see at the end (you should actually get to the number 1000) the last few entries are:

991	BEEF
DEAD	DEAD
993	
994	
995	
DEAD	
997	
998	