DEPARTMENT OF COMPUTER SCIENCE
College of Engineering
University of the Philippines – Diliman

CS 150 Machine Problem Specs

MP Grade Breakdown:

Language Functionalities	**125 pts**
Imperative	
1. Identifiers /Constants/Data Type/s	5 pts
2. Expressions	10 pts
3. Assignment Statements	20 pts
4. Conditional Statements	30 pts
5. Iterative Statements	50 pts
6. Standard I/O	10 pts
Functional	
1. Identifiers /Constants/Data Type/s	5 pts
2. Expressions	20 pts
3. Conditional Statements	30 pts
4. Functions	60 pts
5. Standard I/O	10pts
Error/Exception handlers	**30 pts**
1. Syntax Checker (differentiate a lexical error vs a syntax error)	5 pts
2. 5 Runtime Errors (Type, Logic,Binding, etc.)	25 pts (5pts each)
User Manual	**5 pts**
Sample Programs	**10 pts**
Language Design & Functionalities Report *	**0 pts**
Source Code Documentation *	**0 pts**
TOTAL	**170 pts**
Student's Grade	**AP * TOTAL**

where AP is the average percentage score from the peer evaluation (0-100%)

**** No proper Language Design & Functionalities Report AND Source Code Documentation (PDF) - No MP grade***

Guidelines:

1. This requirement is to be accomplished by pair. Committing plagiarism or assisting another group committing plagiarism shall be dealt with accordingly (minimum 0 for this requirement for all parties involved).

2. Format for papers (softcopy):
    a. A4 size
    b. Font: Arial
    c. Font size: 10
    d. File type: PDF
    e. Special comments: **Include page numbers.**
3. The coursework cannot be completed at the last minute. Spread the work over the time provided.
4. Do not assume anything, If there are questions about the specifications, ask your teacher. This is very important. Please NEVER hesitate to approach or ask .
5. **EVAL** or existing functions evaluating expressions are not allowed to be used for this machine problem.
6. Testing environment is Linux.

DEADLINE: Submit the requirements below:

1. Language Design & Functionalities (PDF)
2. Source Code
   * *Source Code Documentation* (comments on important parts of your code esp. function definitions)
3. Makefile for compiling your sources together
4. User Manual (PDF)
5. Sample Programs
6. Peer Evaluation

on or before the December 23, 2017, 11:59PM

- **Soft copy** of req 1-6 at rapineda1@up.edu.ph
    - Email Subject: **CS150 MP - <PROGRAMMING LANGUAGE>** (Ex: CS150 MP - JOSE RIZAL LANGUAGE)
    - **Do not** compress all your files. Attach each file individually.
    - Submit only **ONCE**. Your email should contain all the files for submission and the names of your groupmates.

MINIMUM SPECIFICATIONS

1. Design an imperative or a functional programming language. Include standard I/O functions/capabilities.
2. Implement an interpreter using Flex and YACC/Bison in **C/C++** or **Python (**http://www.dabeaz.com/ply/**)** for the language you designed. You may import C/C++ libraries for trees/graphs/hashtables.

3. Display debugging information such as line number and details on syntax or runtime errors.

DELIVERABLE:

1. **Language Design & Functionalities Report**
    a. **Introduction**
        i. Language Name
        ii. Paradigm
        iii. Inspiration
    b. **Grammar Definition** using BNF/EBNF of the ff:
        i. Imperative
            1. Identifiers/Constants
            2. Data Type/s
            3. Expressions and Assignment Statements
            4. Conditional Statements
            5. Iterative Statements
        ii. Functional
            1. Identifiers/Constants
            2. Data Type/s
            3. Expressions
            4. Conditional Statements
            5. Functions
2. **Source Codes**
   This includes your flex, and yacc files. Put comments on important parts of your code esp. function definitions.

3. **Makefile**
   Create a makefile that compiles all the source codes into one executable.

4. **User Manual**
   Step-by-step instructions on how to run a source code using your implementation method. Include FAQs, and details on the errors that may be encountered (similar to: https://docs.python.org/3/library/exceptions.html#bltin-exceptions).

5. **Sample Programs**
   Provide 2 sample programs that tests the capability of the implemented compiler/interpreter. I will be using your interpreter/compiler to run other solutions to problems (aside from the ones below).

    a. The first sample program should solve the towers of hanoi problem. Inputs/parameters are number of disks, name of start peg, name of middle peg, and name of goal peg.

Ex: towersofhanoi(3,”A”,”B”,”C”) [functional]/ looped version for imperative
Output:
move disk A to B
move disk A to C
move disk B to C
move disk A to B
move disk C to A
move disk C to B
move disk A to B

b. The second sample program should be an implementation of heapsort using the language you designed. (Heapsort function for functional; iterative version for imperative)

6. **Peer Evaluation**
   Each member of the team will be evaluated based on contributions. Given a total of 6 points, allocate the adequate amount of points per team member (including yourself) based from the contributions rendered. Justify your score. Use the table below as a template. Submit individually to rapineda1@up.edu.ph.

Example:

Name	Score	Justification
1. Riza	**3**	Responsible for implementing functions Designed the language
2. Rae	**3**	Responsible for implementing conditional statements, constants Responsible for the documentation work

DEDUCTIONS

1. Late submission of MP will be penalized with a deduction of **30 points** per day.
2. A maximum of **1 week** will be given for late MPs. Afterwards, the late MP will merit a grade of **0**.
3. Follow all instructions. There will be **10 points deduction** for every unfollowed instruction. *(Even as simple as submitting a .doc instead of .pdf)*
4. If your source code does not work due to errors, the maximum grade that you will get is 40 points. There will be minimal partial points given to programs that do not work.