Programming Paradigms

JOIE ANN M. MAC
Instructor

[CS315 PROGRAMMING LANGUAGES]

Final Project Specifications and Guidelines

PROGRAMMING PARADIGMS

Contents

Rationale and Overview

This project is a major requirement for the course CS315 Programming Languages, one of the core courses of the BS Computer Science program. This project is a mixture of research work and programming.

This project is composed of three major parts: (a) machine problem solution, (b) language definition, and (c) language evaluation.

General Instructions

1. This final project must be done in **5-6 members.**
2. **Each group will propose a machine problem with mid-level complexity**. This is a first come first serve idea. You may write the problem in the given LINK.
3. **This machine problem must be solved in four different languages, each of which is under in a different paradigm (imperative, object-oriented, logic, functional).**
4. No alterations of groupings and machine problem shall be allowed, unless there are compelling reasons to do so.

Grading

This project is 30% of the student's **Final Grade**. It will be graded according to the following distribution of points and weights as shown in the table below.

Machine Problem Solutions			
Code Professionalism		8 points	**50 pts**
Input/output		1 points	
Inline Documentation		<u>1 points</u>	
	TOTAL	10 points	
		X 4 Codes	
Documentation			**50 pts**
Content		15 points	
Professional Furnish		5 points	
		TOTAL	**100 pts**

Table 1 Project Grading Scheme

Machine Problem Solutions

- All programs are assumed to run perfectly and must produce correct output.
- Every program must demonstrate the characteristics of its paradigm.
- Students may use the following languages suggested, together with the development environment, but may opt for another option if they wish to.

Paradigm	Language	Environment
Imperative	C	Dev-C++
Object-Oriented	Java	JCreator or Notepad++ and cmd
Logic	Prolog	SWI-Prolog
Functional	Common Lisp	Steel Bank Common Lisp

Table 2 Suggested Languages and Environments

Documentation Contents

Cover Page
Table of Contents
I.   The Machine Problem
    A.   The Problem Statement – *Define the problem here. Make sure that I/O is clear*
    B.   Solutions – *Copy and paste all source code for every program here*
        a.   Imperative Paradigm Solution
        b.   Object-Oriented Paradigm Solution
        c.   Logic Paradigm Solution
        d.   Functional Paradigm Solution
II.   Language Definition
    A.   Syntax
        a.   BNF from basic syntax until control structures
        b.   Syntax Graphs of the Grammars
        c.   At least 3 derivations
        d.   Parse trees in lieu with the derivations
    B.   Semantics
        a.   Informal Semantics *(in lieu with each BNF)*
            1.   Basic Syntax
            2.   Control Structures
            3.   Data Types
            4.   Subprograms
        b.   Formal Semantics *(in lieu with each BNF, whichever is applicable)*
            1.   Operational Semantics
            2.   Denotational Semantics
            3.   Axiomatic Semantics
III.   Evaluation of the programming languages *(in your own ideas, evaluate these programming languages according to the criteria for evaluating them. You may emphasize their strengths and drawbacks)*
IV.   Concluding Statements *(you may write a generalization of your research)*
V.   Bibliographic Sources *(list of all reliable sources of your research)*
VI.   Appendices
    A.   Screenshots *(show sample run of your programs)*
    B.   Photo documentation *(photos of members working on the project)*

Formatting

- Observe 1 inch margin at all sides.
- Use readable fonts, e.g. Cambria 11, Times New Roman 12, or Arial 11. For source codes in the document, use Courier New 12.
- Document must be professionally furnished.