Assignment 2: Parallel Implementation

Overview

This assignment will test your skills in programming applications to specification in a number of different programming languages and is worth 20% of your non-invigilated (type A) marks for this course. Completion of this assignment requires the:

- Understanding of various programming languages' features
- Understanding of Implementing a program in multiple languages

Timelines and Expectations

Percentage Value of Task: 20%

Due: 04:00 pm Friday the 31st May 2019 (week 11)

Minimum time expectation: 20 hours

Assignment Overview

You are asked to create a program for a pizza shop – however, as this is a comparative languages course, you will be creating the same application in the following programming languages:

- C
- Python,
- Java and
- Lisp

As you implement the application in each language, you should keep notes on the features of the languages used, which you found useful, as well as any issues or complications which arose due to the complexity or lack of any language features. A brief discussion based on these programming features for each individual language accompanying each implementation is required. Finally, a comparative overview of the languages highlighting applicability based on your experience in the design, implementation and debugging of your code is also required.

If you foresee or encounter any complications, you may opt to implement or incorporate additional language features which may be lacking, i.e. data structures. This can be done via:

- Your own implementation,
- Through libraries, or
- Via the incorporation of existing source code. You can use code found on the Internet, but use of any existing code **must be referenced**.

Assessment Details

Sab, the owner of a new Pizza and Pasta shop, is opening a takeaway service for selling delicious Pizza and Pasta to the customers. She wants to offer some packages to interested customers to promote her business. To do so she is offering the following packages:

1. 1 large Pizza = 12 AUD
2. 2 large Pizzas = 22 AUD
3. N large pizzas = N*10 AUD, where N>=3, and the customer will receive 1 garlic bread for every three pizzas [For example, if a customer is interested to buy 10 large pizzas, Sab will provide 3 complementary garlic bread for 100 AUD]
4. 1 large pasta = 8 AUD
5. 2 large pastas = 15 AUD
6. M large pastas = M*7 AUD, where M>=3, and the customer will receive 1.25 Liter soft drinks for every 3 pastas [For example, if a customer is interested to buy 6 large pastas, Sab will provide 2 complementary 1.25 liter soft drinks for 42 AUD]
7. For every 3 pizzas AND 3 pastas, Sab will give a small box of Baklava (a famous dessert item) in addition to garlic-bread and 1.5-liter soft drinks.

You have agreed to design and develop a small console program for Sab, enabling her to select the appropriate item and the package, and calculate the corresponding cost. Once an order is processed, the program will return to the menu ready to commence another order. This payment information should display:

- total payment amounts received for pizza order
- total payment amounts received for pasta order
- total amount of pizzas and pastas sold in that session*

*A session indicates the duration Sab is using the program after opening the program. There is no need for this data to persist once the program has stopped running.

The owner wants the system to be flexible so that she can include additional items and packages at a later date without having to rewrite the entire program. This means you will need to use an interface for processing payments, and polymorphism for the various food items classes, so that new and different packages may be added at a later date with minimal updates to the code. She asked that you provide her with some documentation before you commence coding, so that she is able to verify that the program you intend to code will address her requirements. She would like to see the use cases to summarize the requirements in written format, as well as use case diagrams, class diagrams and sequence diagrams.

Suggested Development Environments

Codeblocks for C '99

Code::Blocks can be downloaded from http://www.codeblocks.org/downloads/binaries To create a new C project is: When you create a project, choose File | New and then Console Application, and then choose C as the programming language.

IDLE for Python:

Python, including the IDLE development environment, can be downloaded from https://www.python.org/downloads/

Eclipse for Java 7 or Java 8

Eclipse may be freely downloaded from http://www.eclipse.org/downloads/

Eclipse does not come with the Java JDK, which must the downloaded separately from
http://www.oracle.com/technetwork/java/javase/downloads/index.html

Ensure that your Eclipse type and Java type match – i.e. 32-bit Java for 32-bit Eclipse, or 64-bit Java for 64-bit Eclipse. If you mix and match it won't work.

GNU CLisp for Common Lisp

CLISP 2.49 can be sourced from http://sourceforge.net/projects/clisp/files/latest/download

Any good text editor would be suitable for writing the source code.

Additional Documentation – Language Suitability Report

The design of each programming language incorporates a number of decisions about the language which make it more or less suitable for given tasks. During your implementation of the pizza program in each of the languages, you should make notes about the language features which exist or do not exist, and which have therefore made program development easier or more difficult.

Where a language has not provided a feature which would have been useful to the implementation of the program, or where the complexity of using a language feature has been high you should remark upon it and briefly discuss a mechanism or feature of another language which would have made development easier.

After completing the application in all languages (or as many as you can), discuss the comparative ease of implementation in terms of the design, implementation and debugging for each programming language, including how robustness issues were addressed.

Submission and Marking Process

You must supply your program source code files and language suitability report documentation in as single compressed archive called:

ITECH5403_Assignment_2_<YOUR-NAME>_<YOUR-STUDENT-ID>.zip

You may supply your programming language suitability report in either Word or LibreOffice/OpenOffice format in which the document can be edited – no proprietary Mac-specific formats, please.

Assignments will be marked on the basis of fulfillment of the requirements and the quality of the work. In addition to the marking criteria, marks may be deducted for failure to comply with the assignment requirements, including (but not limited to):

- Incomplete implementation(s), and
- Incomplete submissions (e.g. missing files), and
- Poor spelling and grammar.

Submit your assignment (all program source files plus your discussion document) to the Assignment 2 Upload location on Moodle before the deadline of Friday of Week 11 at 4 pm.

The mark distribution for this assignment is explained on the next page.

Marking Criteria/Rubric

Assignment 2 – Parallel Implementations

Student name: **Student ID:**

Requirement	Available Marks	Student Mark
Implementation of the pizza shop program in the C programming language. Areas of note include: - ○ Use of data structures ○ Robust input handling which does not cause program termination if provided with bad data (i.e. program expects a number, gets given alphanumerical data). Discussion on implementation: ○ Language features, issues and suitability.	15 5	
Implementation of the pizza shop program in the Python programming language. Areas of note include: ○ Python Standard library ○ List mechanisms Discussion on implementation: ○ Language features, issues and suitability	15 5	
Implementation of the pizza shop program in the Java programming language. Areas of note include: ○ Object orientation mechanism/method calls ○ Error handling ○ Standard Java libraries Discussion on implementation ○ Language features, issues and suitability	15 5	
Implementation of the pizza shop program in the Lisp programming language. Areas of note include the Lisp: ○ Use of recursion ○ Lists ○ Inbuilt data structures Discussion on implementation ○ Language features, issues and suitability	15 5	
Documentation and discussion of the comparative ease of implementation (design/implement/ debug) in each programming language, including how robustness issues were addressed.	15	
Spelling and grammar	5	
Total		/100
Contribution to unit mark (out of 20%)		%

Feedback

Marks will be uploaded in fdlGrades and a completed marking guide provided in Moodle within 2 weeks of assignment submission.

Plagiarism:

Plagiarism is the presentation of the expressed thought or work of another person as though it is one's own without properly acknowledging that person. You must not allow other students to copy your work and must take care to safeguard against this happening. More information about the plagiarism policy and procedure for the university can be found at http://federation.edu.au/students/learning-and-study/online-help-with/plagiarism.