Department of Computer Science
University of Pretoria

Programming Languages COS 333

Practical 1: Research Assignment

August 2, 2022

1 Objectives

This practical lab experience aims to achieve the following general learning objectives:

- Provide experience in independent research, focusing on topics related to programming language theory;
- Give superficial exposure to some of the more esoteric topics related to programming languages, which are not the primary focus of the course or the prescribed material;
- Provide some introductory experience in the use of the LaTeX typesetting system;
- Provide some introductory experience in the use of some special-purpose programming languages.

2 Plagiarism Policy

The Department of Computer Science considers plagiarism as a serious offence. Disciplinary action will be taken against students who commit plagiarism. Plagiarism includes copying someone else's work without consent, copying a friend's work (even with consent) and copying material (such as text or program code) from the Internet. Copying will not be tolerated in this course. For a formal definition of plagiarism, the student is referred to `http://www.ais.up.ac.za/plagiarism/index.htm` (from the main page of the University of Pretoria site, follow the *Library* quick link, and then click the *Plagiarism* link). If you have any form of question regarding this, please ask one of the lecturers, to avoid any misunderstanding. Also note that the OOP principle of code re-use does not mean that you should copy and adapt code to suit your solution.

3 Submission Instructions

The following submission requirements must be adhered to for this practical lab experience. Failure to do so will result in your practical submission receiving a zero mark:

3.1 Document

The final research report must be compiled using the LaTeX typesetting system. Documentation related to the LaTeX system is available from this practical's folder on the course site [5]. The Informatorium Linux installations include the commonly used teTeX implementation of LaTeX. The MikTeX system is available for free download, if you prefer to use Windows (see `http://www.miktex.org/`).

You must include a list of references for the sources you consult. All references must be cited at the appropriate location within each question. Since your references will be marked (see Section 6), ensure that there are sufficient (do not make unsubstantiated statements, unless they are clearly your own opinion) and that each is complete and correct. Note that this does not mean you must include a reference for every question (questions that ask for your opinion, for example, do not require a reference). You will receive marks for managing your references with BibTeX. Documentation for BibTeX is also provided on the course site [6].

3.2 Upload and Assessment

There is a separate upload slot for the research questions report and each of the practical tasks. Upload your practical-related files to the appropriate assignment upload slots on the course website. Multiple uploads are allowed. The deadline is **Monday, 15 August 2021, at 23:00**. The upload for the research questions report must be an zip archive, and must include your complete research report (both a compiled PDF file, and the complete LaTeX and BibTeX source). You are also required to submit only your research questions PDF to a Turnitin assignment upload slot. Submissions will not be permanently uploaded to the Turnitin archive. Submission to Turnitin is required in order to receive a mark for your answers to the research questions. The uploads for the practical implementation tasks must each be only the program source file for the corresponding task. It must be possible to run your submissions with only the files you upload.

The reports will be assessed offline by the teaching assistants. Details related to the assessment of the implementation questions will be announced on the course ClikcUP page.

---

4 Research Questions [Total: 15]

Answer the following questions. Your answers should be as complete and clear as possible. Provide only information relevant to the question, and be as concise as possible. Overly verbose and lengthy answers will probably disadvantage you during the marking process:

1. **Explain** what an esoteric programming language (or *esolang*) is. [1]
2. Esoteric programming languages can be categorised in a variety of different ways. For each of the following categories **identify** one advantage and one disadvantage associated with languages falling within the category:
   (a) Funges. [2]
   (b) Stateful encoding languages. [2]
3. Choose any two esoteric programming languages. **Describe** the languages in terms of their designer(s), year of initial design, and their general syntactic and semantic characteristics. Provide a short example code snippet, to illustrate the language's general characteristics (you do not have to write the code yourself). [5]
4. Consider the concept of "Design by Contract" (DbC). **Explain** what DbC broadly entails. **List** two (2) languages that natively support DbC. [5]

---

5 Implementation Questions [Total: 20]

Implement the following programs. All the required software is either already available in the Informatorium, or can be easily installed on the lab machines. Where the languages allow, include descriptive comments in each program, to explain your code, and describe how to run your program and modify its sample operation. Make sure that your programs run easily, as you will not receive marks if their operation is obscure.

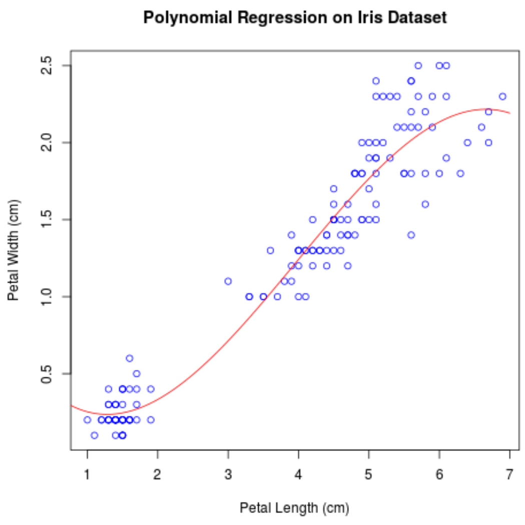
5.1 Gnuplot [5]

Gnuplot is a program designed for two- and three-dimensional graph generation. It is not a programming language in the strict sense of the word, but does provide scripting functionality. Documentation and tutorials for gnuplot are available on its official website (`http://gnuplot.info/`).

For this task you will use the well-known Iris plants data set, which is provided as the `iris.data` file on the course website. You may not modify either the name or the content of this file. Background information on the data set is available from the UCI Machine Learning Repository (`https://archive.ics.uci.edu/ml/datasets/iris`).

Write a gnuplot script that draws a scatter plot comparing the petal length and petal width attributes. Use the least squares method to perform a third degree polynomial regression on the scatter plot data points, and superimpose the polynomial on the scatter plot. Do research online regarding scatter plots and polynomial regression if you are not familiar with these concepts. Save your program in a file named `task1.plt` and submit this file to the appropriate upload slot on the course website.

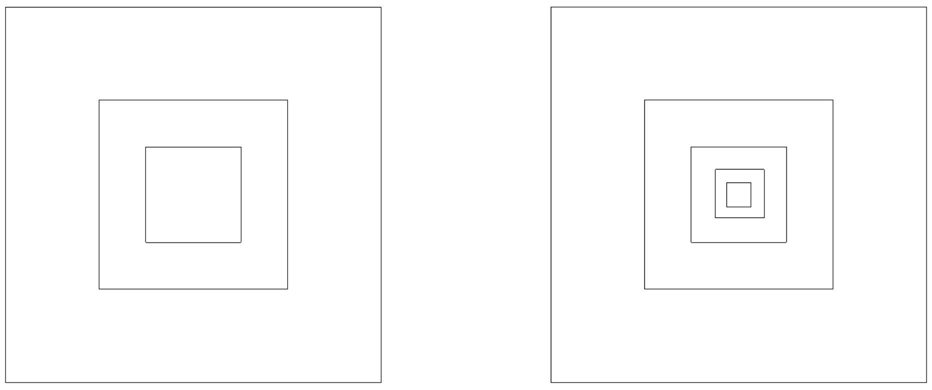
Below is an example of the scatter plot (represented by the blue circles) and the polynomial (represented by the red line):



5.2 Snap! [5]

Snap! is an educational programming language developed at the University of California, Berkeley [3]. Snap! uses a drag-and-drop web interface allowing a user to specify a program, and provides a cloud service to save programs. Snap! and its documentation are hosted at `http://snap.berkeley.edu/`.

Implement a Snap! program to recursively draw a set of squares nested inside one another, similar to the following examples:



(a) (b)

In the above examples, Figure (a) shows three squares nested inside each other, while Figure (b) shows five squares nested inside each other.

Snap! uses the concept of a block to define a function. The block should accept two inputs (arguments). The first should specify the length of the outer square's sides, and the second should indcate how many squares should be drawn. The block should draw each inner square with a side length that is half the length of its containing square. Include an example call to the block.

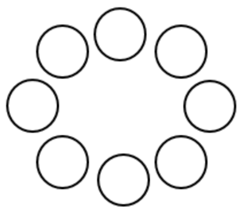
Note that you **must** use an iteration command to draw each square, and the procedure that draws the nested squares **must** be recursive. Save your program in a file named `task2.snap` and submit this file to the appropriate upload slot on the course website (Snap! allows programs to be stored online, but for submission purposes, save your program locally).

5.3 LOGO [5]

LOGO is a functional programming language often used for turtle graphics [2]. UCBLogo is an open source implementation of the LOGO language. A Windows installer, along with documentation [4] on how to use it, is available on the COS 333 course page. The source code and a Linux implementation can be obtained from `http://www.eecs.berkeley.edu/~bh/logo.html`.

For this task, you must **implement** a recursive procedure in UCBLogo, which will draw a set of circles arranged in a circle. It must be possible to specify the number of circles that have to be drawn in a simple fashion (i.e. with minor modification to the program source code). Save your LOGO program in a file named `task3.lgo` and submit this file to the appropriate upload slot on the course website.

Below is an example of the output the program should provide, assuming 8 is the number of circles to be drawn:



5.4 Inform 7 [5]

Inform 7 is a simple programming language for writing interactive fiction which is based on natural language [1]. You may download it here `http://inform7.com/`. This comes with examples and ample documentation. Using Inform 7, **implement** a simple game with the following functionality:

1. There are two rooms, namely room A and room B. The player starts in room A.
2. Room A contains two boxes. Provide descriptions of each of these, detailing them with the flags of any two countries of your choosing.
3. Room B contains an aquarium which, in turn, contains Paul the Octopus.
4. To complete the game, a player must be able to: pick up and carry each box from Room A into Room B, upon which they may approach the aquarium and place each box within the tank. Ensure that the boxes can be dropped into the the aquarium in either order.
5. Only once both boxes have been placed, Paul the Octopus must open a box of your choosing. To achieve all of this, use simple verbs, objects, containers and descriptions.

Save your Inform 7 program in a file named `task4.inform` and submit this file to the appropriate upload slot on the course website.

6 Marking

The marks for this practical lab experience will be allocated as follows:

Category	**Mark Allocation**
Research questions	15 marks
Implementation questions	20 marks
References	5 marks
Use of LaTeX and BibTeX	5 marks
TOTAL	**45 marks**

References

[1] Wikipedia, The Free Encyclopedia. Inform. Online: `http://en.wikipedia.org/wiki/Inform`, accessed 24 July 2010.

[2] Wikipedia, The Free Encyclopedia. Logo (programming language). Online: `http://en.wikipedia.org/wiki/Logo_(programming_language)`, accessed 24 July 2010.

[3] Wikipedia, The Free Encyclopedia. Snap! (programming language). Online: `https://en.wikipedia.org/wiki/Snap!_(programming_language)`.

[4] Brian Harvey. *Berkeley Logo 6.0 — Berkeley Logo User Manual*, 1993.

[5] Tobias Oetiker, Hubert Partl, Irene Hyna, and Elisabeth Schlegl. The not so short introduction to LaTeX 2_ε, version 6.2, 28 February 2018.

[6] Oren Patashnik. BibTeXing, 8 February 1988.