Department of Computer Science
University of Pretoria

# Programming Languages
# COS 333

Practical Lab Experience 1: Research Assignment

July 25, 2017

## 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. Marks will be awarded for adherence to these guidelines (see Section 6):

### 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 [3]. 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. Because 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. You may reference online sources. You will receive marks for managing your references with BibTeX. Documentation for BibTeX is also provided on the course site [4].

### 3.2 Upload

Upload your practical-related files as tar-gzip archives to the appropriate assignment upload slots on the course website. There is an upload slot for the research questions document, where you must upload a tar-gzip file (named `s99999999.questions.tgz`, where `99999999` is your student number), which must include your complete research report (both a compiled PDF file, and the complete LaTeX and BibTeX source). Each implementation task has a separate upload slot, where you must upload a tar-gzip file (named `s99999999.implX.tgz`, where `X` is the number of the implementation question, and `99999999` is your student number), which must include the complete source code implementation of the task. Multiple uploads are allowed, but only the last one will be marked. The deadline is **Monday, 7 August 2017, at 23:00**.

## 4 Research Questions [Total: 45]

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. [2]
2. The general consensus is that esoteric programming languages are little more than amusing diversions for computer science researchers. **Argue** for and against this viewpoint. [5]
3. Choose any two (2) esoteric programming languages. **Describe** each language in terms of its designer(s), year of initial design, general syntactic and semantic characteristics, and whether the language is Turing complete. For each language, provide a short example code snippet, to illustrate its general characteristics (you do not have to write the code yourself). [20]
4. Investigate stack-based programming languages. **Briefly explain** what the basic premise is that underlies stack-based programming languages. **Provide and example** of a stack-based programming language. [4]
5. Turing is an example of an educational language. Consider the syntax of Turing, especially in relation to very simple programs. **Suggest** why Turing is a good choice as an introductory programming language as opposed to a fully-featured higher-level programming language like C++ or Java. **Suggest** a reason that a programming language like Turing is not a good choice as a general instructional programming language. [4]
6. Consider the concept of "Design by Contract" (DbC). **Explain** what DbC broadly entails. **List** two (2) languages that natively support DbC. [5]
7. **Consider** the AWK programming language. **Briefly explain** what the intended application area of AWK is. **Briefly describe** the overall syntactic structure of an AWK program, and **explain** how this structure facilitates the use of AWK in its intended application area. [5]

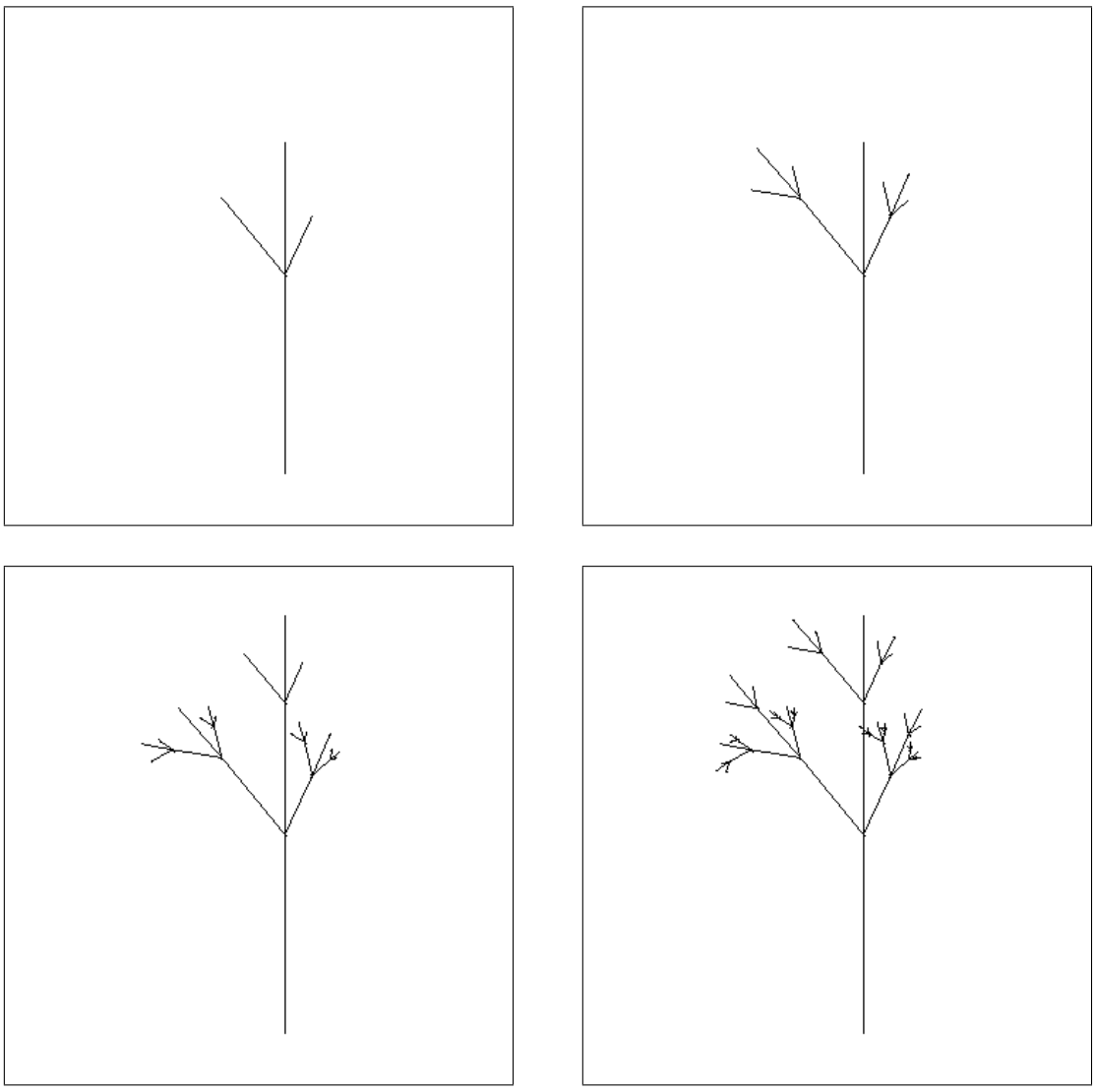
# 5 Implementation Questions [Total: 10]

Implement the following programs. All the required software is available under the Informatorium Linux installations. For each question, include at least one comment at the top of the program, that details the complete instructions for executing the program:

## 5.1 Snap! [5]

Snap! is an educational programming language developed at the University of California, Berkeley [2]. 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 draw a recursively defined tree, where branches are sub-trees. Provide a mechanism for specifying to which level the tree should be generated. Below is an example of what the basic tree structure should be:



### 5.2 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 the office and the lab. The player starts in the office.
2. The office contains a CD.
3. The lab contains a computer. The computer can be either activated or deactivated. If the computer is activated (define this as a new action), it turns on. If the computer is deactivated (also a new action), it turns off. The computer can't be activated once it is already on, and can't be deactivated when it is already off. The CD can be inserted into the computer, but only if the computer is activated.
4. To complete the game, a player must be able to: pick up and carry the CD from the office into the lab, after which they must activate the computer and insert the CD into the computer.
5. If these actions are performed correctly, the computer should display a message, ending the game. To achieve all of this, use verbs, objects, containers and descriptions.

## 6 Marking

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

| **Category** | **Mark Allocation** |
|---|---|
| Research questions | 45 marks |
| Implementation questions | 10 marks |
| Use of LaTeX and BibTeX | 10 marks |
| Report structure | 5 marks |
| References | 10 marks |
| Language style, grammar and language | 10 marks |
| **TOTAL** | **90 marks** |

## References


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

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

[3] Tobias Oetiker, Hubert Partl, Irene Hyna, and Elisabeth Schlegl. The not so short introduction to LaTeX $2_\varepsilon$, version 4.14, 4 April 2004.

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