PROJECT SPECIFICATION - Project (Technical Computing) 2018/19

Student:	Matthew Ball
Date:	18/10/2018
Supervisor:	Chris Bates
Degree Course:	Software Engineering
Title of Project:	Development and implementation of a high-level interpreted language

Elaboration

Human languages are incredibly diverse, there are widespread shared features, as well as nearly unique ones. Programming languages are also diverse; the key difference however is that while human languages have organically grown to be fit for purpose, explaining many differences and quirks, programming languages have been expressly designed for a purpose.

For this project I intend to develop a high-level language, in that aim I will need to make design decisions regarding things like variable types, data structures, syntax and formatting.

The deliverable of this project will be a lexer and parser for the language I design. My goal is to be able to generate executable code in C++.

Project Aims

- Understand ideas and techniques behind programming language design
- Define grammar for language
- Understand how compilers/interpreters work and how to write an interpreter
- Write a lexer for developed language
- Write a parser for developed language
- Evaluate accuracy of interpreter

Project deliverable(s)

A developed high-level language, including a formal grammar definition. The intention of this language is to be as close to readable English as is practical, which allows a language to be intuitive for people with little technical understanding, and allows competent developers to describe complex operations abstractly, this is a core idea of 'literate programming'.

I will also develop a lexer and parser for my language, in C++, with the aim of being able to generate executable C++ code from a script written in my language.

Action Plan

26/10:	Project proposal handed in
02/11:	Setup version control
16/11:	Produced basic lexer prototype, able to lex a simple line of code
30/11:	Understand compiler design
30/11:	Compiled sources ahead of information review
14/12:	Designed language and produced EBNF grammar
11/01:	Complete lexer and parser
22/02:	Code generation
01/03:	Preliminary testing complete
15/03:	Make appropriate changes following testing
22/03:	Completed final testing
05/04:	Draft critical evaluation
26/04:	Online submission
27/04:	Physical hand in to cantor, include clear instructions of how to run project
before 12/05:	Demonstration