Project Name	Hopeful - A First Programming Language
Author	Gillian Mullen
Supervisor	David Sinclair
Document Title	Survey Analysis
Summary	This project, Hopeful - A First Programming Language, will involve the development of a programming language. This language will be aimed at beginning programmers in an undergraduate setting. It will have a simple and clean syntax that will allow students to focus on the fundamentals of programming, instead of a complicated syntax.

Overview

From the outset, my intent was to create a language that had a clear and simple syntax that allows the students to focus on the fundamentals of programming. To help myself achieve this end, I surveyed Engineering and Computing students, as well as School of Computing staff. I believe that this survey provided a very good base to create my language upon, as well as backing up my belief that this language needs a clear and simple syntax to succeed in helping to teach students programming.

In the process of this research, four sample groups were taken into account. First is beginning programmers, accounting for first-year Computer Applications students. Secondly is advanced programmers, accounting for fourth-year Computer Applications students. Thirdly is the School of Computing staff. Lastly is Engineering students, second-year and up for all Engineering disciplines. In analysing these results, I weighed the results from beginning programmers and staff more than the other groups. I did this because I believe that these groups will give me a better insight into what beginning programmers need to learn better, while the other groups would be weighed more by personal bias.

Fundamentals

Upon studying the programming curriculum of first-year Computer Applications students, I decided that this language will need multiple types, arrays, functions, and control flow functionality. The types I will be including are integers, strings, booleans, floats, and the void type for return type in functions.

Upon asking staff do students learn better with statically or dynamically typed variables, the majority answer was statically which is what I chose for my language. Upon being asked a similar about question about the benefits of using a text editor versus an Integrated Development Environment (IDE), they answered with text editor which is what I am now developing for. Another question to staff was regarding lists and arrays, asking which one, or both, would be more useful in teaching a student. The results of this question were split between using both or just arrays. I made the decision to choose just arrays, as I believe using only one would make the language simpler. Including both in the language could confuse the students as they are quite similar in concept.

Another decision that needed to be made was passing by value or reference into function. I choose passing by value because I believe it would be simpler to understand for beginning students at first glance.

Formatting

In terms of formatting, there are two styles of programs that are popularly used, especially within languages taught to beginning programmers at DCU. First is the C family and Java styles, which involves using curly braces and brackets for control flow and lines ending in semicolons. Second is the Python style, which involves indentation and colons for control flow and lines ending with a new line. Ideally, a perfect blend between the two would

be the ideal as that means that a new programmer could progress to those languages from this one very easily. In addition to this, Python and Java are popular languages within the Computer Applications curriculum currently which would be an additional bonus.

The results of the survey seem to be split in regards to the formatting. The beginning programmers, who mostly started with Python, prefer its formatting. While Engineers, who mostly started with C, prefer that language's formatting. While the two other groups were split between the two formats. The decision I made was to stick with the C family and Java formatting, with curly braces and semicolons, because I believe it is the more universal formatting and that it allows for easier recognition of meaning within the program. I also made this decision because universal programming standards use indentation within curly braces regardless, so it allows for both formats within the same piece of code.