Algonquin College Logo

# SCHOOL OF ADVANCED TECHNOLOGY

### ICT - Applications & Programming

### Computer Engineering Technology – Computing Science



A11

Language Specification

Team:

[Byron Jones] - Id: [040585687] / [Sean Wray] - Id: [041036555]

Language Name [Dublin]

***This template is suggested (not mandatory) to answer A11 Specification.***

|  |  |
| --- | --- |
| **Part**  **1** | **Language User Reference** |

**EXPLANATION**

*The purpose of this assignment is to invent a new computer language.*

* *This language can have the syntax and structure of your choosing.*
* ***Option 1: Adapt the ‘BOA’ language to be Python compatible.***
* *Option 2: Define a DSL – Proper to solve specific problems (ex: science, economy, music, etc.)..*
* *This is going to be a fairly basic language. There's a lot of functionality that we'll be skipping over, while we implement the basics. You will need to tell me those basics, of course. In this document, I'm going to explain the steps of what to do with a bit of detail.*
  1. **User Manual**

**Element 1: Name / Extension**

*[Name your language! Name it after a city that means something to you. We suggest you use one "word" for the name.]*

*[What is the filename extension of your language? For example, for C it is .c, and for Professor Paulo's* ***Boa language*** *it is ".****boa****".]*

*[What is your language patterned after, or what is it similar to? What languages are inspiring your choice? It's okay if you're following Python closely.]*

Name: Dublin

Extension: .dub

Language is patterned after Python due to the increased popularity of Python.

**Element 2 – Comments**

*[Comments: I want to do comments in your language. How do I write them?]*

Commenting will be done with the use of ‘#’ symbol

**Element 3 – Keywords**

*[Keywords: List the sequence of reserved / key words from your language]*

**Keywords:**

**and, for, if, in, is, not or, True, False, while**

**Element 4 – Variables and Datatypes**

*[Datatypes: Define integers, real numbers (float points) and strings]*

* *How many bytes are you needing for your variables? This determines their ranges. (Chambly, for instance, has a special 64-byte integer. This is ridiculously huge for most purposes.)*

*[Remember to define the number of bytes – and, if possible, range]*

**Datatypes:**

**integers: 4 bytes or 32 bits**

**float: 8 bytes or 64 bits**

**Character: 1Ginkgo biloba tea byte**

**String: 1 byte per char**

**Boolean: 2 bytes**

**Element 5 – Variables and Datatypes**

*[Variables: How would a programmer define variables that can hold integer numbers (numbers with no decimal point), floating point numbers (numbers with a decimal point) or text (ie: strings in Java). This is element 1. Consider if you want to flag the variables in a special way, like SOFIA or BASIC, or not, like C or Java.]*

**Integer: int**

**Float: float**

**Character: char**

**String: String**

**Boolean: bool**

**Element 6 - Commands**

* ***Attribution****: How does your language let a programmer assign a value to a variable? (Will you allow casting? If so, how will it work?) How will your language handle math, and will it allow strings to be concatenated (merged)?*

**Assignment in the Dublin language will be done like Python, using equal signs,**

**ie. Num = 5**

**Concatenation will be allowed with strings by using a comma, “+”**

**Ie. Greeting = “hello” , “how are you?”**

**Math functions will follow standard computer symbols used in math operations**

**Ie. Addition uses +, subtraction uses -, multiplication uses \* and division uses /.**

**Casting will be allowed, with the data type to be placed before the variable being casted**

**Ie. (float) totalWeight**

* ***Selection****: How does your language do if-style logic? (Optional: Do you want to do some kind of switch/case as well?). You will need to explain how "conditionals" work in your language. How do you write Boolean operations, such as "or", "and", "not", and other conditions, such as less than, greater than, etc?*
* **Dublin will use a python style approach to if statements, using the same syntax “if condition” and for else “elif someothercondition”. We will continue to uses pythons simple approach to conditional statements, using “and” “or” and “not” for those operators when doing conditionals or comparators. Curly brackets will enclose the blocks of code in each if statement.**
* ***Interaction****: How will your code handle looping? (You can do one or more of a for-style loop, a while/do loop, etc.)*
* **Dublin will use a standard python style for loops and while loops but use parentheses to enclose the body of the loop as the primary method of looping through variables, an example can be seen below**:
* **For someVar in someArray(Perform some task)**
* **While some\_condition (Perform some task)**
* ***Input****: How does your program get input from the keyboard? (Strings are easiest.)*
* ***Dublin will take in user input using Strings****.*
* ***Output****: What would a programmer type to put output on the screen? What sort of variables or data will your code take?*
* **To output to a screen or console, print() will be used and output anything found between the brackets. If they are outputting text, quotation marks should be used like so: print(“your text here”)**
* ***Functions****: [Function definition: parameters and returning types]*
  + *What will be the syntax for making a function or subroutine?*
  + *How will it take parameters?*
  + *How will it return results?*

To define a function initially, Dublin will follow python syntax: def functionName(function parameters), with the body of the function being enclosed in curly braces {} and any value being returned will follow the return keyword.

Ie. Def add\_values(value1, value2) {

Sum = Value 1 + value 2

Return Sum

}

**Element 7 – Proper elements**

*[Include specific features / elements to be included in your language]*

* *What you could include / modify? Think about new datatypes / structures / commands, etc.*
* *Note: Do not share this info (it is supposed to be your proper elements in the language.*
* **Python does not have any genuine constants in its language, as it is a mutable language. This leads to significant slowdown at times. In our Dublin language, which is being compiled in ANSI C, we will try to develop a python compatible version of a constant, or immutable variable. To define a constant we could use the $ symbol, as it is unused in python. For example $constantPi = 3.14.**
* **(https://www.infoworld.com/article/3566382/4-powerful-features-python-is-still-missing.html)**

|  |  |
| --- | --- |
| **Part**  **2** | **Examples** |

**Option 1: Python-like**

**Hello World**

|  |  |  |
| --- | --- | --- |
|  | String hello = “Hello World”  Print(hello) |  |

**Sphere Volume Expression (or any other example)**

|  |  |  |
| --- | --- | --- |
|  | float pi = 3.14  V = 4/3 \* pi \* r^3  $constant float pi = 3.14 |  |

*[TIP: See examples in the Lecture Notes –* ***Appendix 1****]*

**Option 2: DSL**

**[Your example here]**

|  |  |  |
| --- | --- | --- |
|  | [Your Code here] |  |

|  |  |
| --- | --- |
| **Part**  **3** | **Architectural Aspects** |

**Advantages**

*[What's the goal of your language? Are you trying to make something simple, fun, complicated? My personal language, Chambly, is based around being useful to scientists. (You can just make something up here, honestly. Think about it a little bit, have a little fun.)]*

**The goal of Dublin is to create a language very similar to Python and compatible with it but with a few minor changes – notably the addition of brackets to enclose method and loop bodies, rather than relying on indentation to enclose them. Furthermore, the addition of constants will allow programmers working in scientific and math related fields to create constants and not worry about them being overwritten due to the mutability of python constants.**

**Strategy: C Implementation**

*[How your language can be implemented in C – ex: datatypes]*

* *In plain English, or maybe even some high level pseudocode, how are you going to parse your language? You will be writing a compiler for your language, so these are some things you need to think about.*
* **Use functions to scan for keywords in Boa and then translate them into their Dublin equivalents. After studying how Dublin uses various keywords, functions, loops etc. we will know what delimiters to identify when parsing the language out and completing the translations.**

**Ex. While boafile hasnextline (**

**If boaPrintdetected {**

**Print Contents in Dublin**

**}**

**Elif boaIntdetected{**

**Convert to Dublin Int**

**}**

**)**

*[Your ideas about how to identify elements from language]*

* *Consider your "write to the console" command as an example. How will your compiler detect it? How will it sort out what to write to the console? What if there's some literal text (ie: "this is going to get printed") instead of variables?*
* **Once we know what keyword Boa uses to output to console and how it encapsulates the printed messages, we will be able to program our parsing function to identify the keyboard and the encapsulating characters to then convert it into Dublin and perform a print function, ensuring that everything between the print function characters is printed and not treated as a variable.**

*[Your ideas about how to identify scope (ex: blocks between conditionals or functions)]*

* *How do you mark a block of code? If I use your loop logic, how do I control what portion of code gets looped through? In C, you might use { and }. In Python, the indentation is what matters. How does it work in your language?*
* **In the Dublin language we will predominantly use brackets and curly braces to identify scope, as opposed to the indentation format of python.**

**Basic ideas about C implementation**

*[Which structures or datatypes you imagine to use in your language implementation]*

* *What do you think is going to be really hard about this? What would be, in your opinion, the hardest part of parsing your own new language? You don't have to write an essay, a paragraph or two will be fine.*
* **Using C to perform the parsing operations will likely cause a few problems. C is a language where memory allocation is dynamic and manual, and so in our language, which is geared more towards mathematics and scientific calculations using constants issues could arise when using large numbers and complex mathematical calculations. These issues could also make the parsing of large numbers and variables difficult. We will have to take great care that our compiler does not encounter out of memory scenarios during execution.**

**Another interesting challenge will be discovering how to add constants, which are immutable, into a very mutable styled language like python. C will help make this easier as it does have constants.**

***Note 1: C Datatypes***

*Remember that you are implementing your language in ANSI C. For this reason, you cannot create arbitrarily your language (from scratch). You need to use what is already provided by C Compiler. For this reason, think about using and defining the language obeying the datatypes.*

**Problems when using C implementation**

*[Your vision about main problems / difficulties when implementing a new language (ex: memory allocation, range of datatypes]*

**Similar to the section above, Dublin and Python do not use dynamic memory allocation so translating from our compiler language, C, to our output language, Dublin, we may run into issues of memory allocation and compatibility. More research will be needed as we develop our compiler.**

**Some thought will also need to be given towards strings, which are character arrays in C are arrays of Chars, however in Dublin and Python anything between quotations is treated as a String.**

**FINAL SUGGESTIONS**

*Here some ideas to think about your language....*

* *Don't make this assignment harder than it needs to be on yourself. Focus on making the syntax for your language that meets our requirements. Worry about extra features later.*
* *Don’t worry if your new language winds up having really difficult parts. You'll be allowed to change your language as you go along, as long as you make "patch notes" to explain those changes. We'll tell you about this later.*
* *There's a marking key at the end of* ***CST8152\_Compilers\_F22-A11-Specification*** *that should steer you along for grades. Focus your efforts on where you'll get the best results.*
* *Finally, think about creating an “master-piece”: until now, you have used several languages. And if you have conditions to define yours, how it could be?*

**References**

**https://www.infoworld.com/article/3566382/4-powerful-features-python-is-still-missing.html**

Algonquin College

Fall, 2022