Algonquin College Logo

# SCHOOL OF ADVANCED TECHNOLOGY

### ICT - Applications & Programming

### Computer Engineering Technology – Computing Science



A11

Language Specification

Lab Professor / Lab Session:

Paulo Sousa / 312

Team:

Sean Bradbury 041101614

Language Name [Chopin] (DSL)

***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 ‘Sofia language to be R compatible (see* <https://www.r-project.org/>*).*
* *Option 2: Define a* ***DSL*** *– Proper to solve specific problems (ex: science, economy, music, etc.)..*

*This is going to be a 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**

Chopin is the name of the language. The extension will be .cho

It will be similar to Java since I most familiar with it.

**Element 2 – Comments**

“//” will start a single line comment

“///” will start and end a multi line comment

**Element 3 – Keywords**

return

int

dec (Decimal)

pitch

duration

velocity

beat

loop

string

boolean

for

do

while

if

else

**Element 4 – Datatypes**

Integers are 4 bytes from -2,147,483,648 to 2,147,483,647

Floats are 4 bytes from 3.4 x 10^-38 to 3.4 x 10^+38

Pitch, duration, beat and velocity are floats that cannot be negative, also 4 bytes

Pitch represents the frequency of the note in hertz

Duration is the number of beats long a note is

Beat is the beat of the music

Velocity is the volume in dB

Strings are 1 byte per character plus 8 bytes for the pointer

**Element 5 – Variables**

Variables will be declared with the data type and then variable name ex.

Pitch Ab

duration halfNote

velocity forte

dec floatName

string stringName

The # symbol should be allowed for variable names, since it is a common symbol in music. C#maj7 for example should be a valid variable.

**Element 6 – Methods / Functions**

*[Variables: How would a programmer define methods]*

Functions will be declared with the return type, function name, parenthesis with parameters, then curly brackets ie.

pitch functionName ( parameters ) {

code;

return pitchVar;

};

**Element 7 - Commands**

* ***Attribution / assignment****: 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)?*

To assign a variable the “=” symbol will be used ex.

pitch A = 440;

duration dottedQuarterNote = 1.5;

dec floatNumber = 10.1;

Strings will use double quotes.

string stringName = “Hello”;

casts will be allowed and will use parenthesis.

int myNumber = 5;

dec myDecimal = (dec) myNumber;

* ***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?*

Conditionals will work the same as C.

if ( expression ) {

code;

}

Or is || and and is &&. < is less than, > is greater than, >= is greater than or equal, <= is less than or equal.

* ***Interaction****: How will your code handle looping? (You can do one or more of a for-style loop, a while/do loop, etc.)*

For loops will work like in Java

for ( int i = 0, i < 10, 1) {

console(i + “ “);

}

While loops will also be the same as Java:

{

Code;

} while ( expression );

Do {

Code;

} while ( expression );

There will also be an additional way to loop music called “loop”. This will work like a function. It is similar to a for loop but simpler. The first parameter is the note or track being looped, the second parameter is the starting beat, the third parameter is the ending beat, and the fourth parameter is the number of iterations. This will create a track object.

loop (noteName, 1, 4, 24);

Input is interpreted as strings from the keyboard

write() is used to output to the screen. It should take any variable or data type.

The syntax to write a function is as follows:

int foo ( parameters ) {

Code;

return intValue;

}

The return keyword will be used to return a value

**Element 7 – Proper elements**

I plan on including music specific data types such as pitch, beat, and velocity. These will all be non-negative floats. Chords and keys can be represented by arrays of pitches.

Some objects that should be implemented inherently are Notes, Tracks, and Projects. For example, a Note object would have a pitch, duration and velocity (volume). A track would have a map of Notes and the beats they are on.

|  |  |
| --- | --- |
| **Part**  **2** | **Language Comparison** |

**Comparing with C language**

**Differences**

|  |  |  |
| --- | --- | --- |
|  | Chopin is an interpreted language, while C is a compiled language. Chopin will only have strings, not chars. The Boolean data type exists in Chopin. Chopin is object oriented and c is functional. Chopin also has pitch, beat, and velocity data types which are a positive float. Chopin will have multi-line comments using triple slashes, while Ansi C does not have multi-line comments. |  |

**Advantages / Disadvantages (in comparison with C)**

|  |  |  |
| --- | --- | --- |
|  | Chopin will have additional functionality for music data. Pitch, beats, notes, etc. will have support. Chopin will have the value data type which will force a float to be positive to be used for things like pitch, volume, and duration where a negative value makes no sense. Chopin also has multi-line comments which is convenient for documentation.  Chopin will be slower than C due to having to be interpreted. There will also be less features and libraries available. Chopin is very niche, with a specialty in musical data, while C can be used for almost anything. C is also already known by most programmers while nobody knows how to use Chopin. |  |

**Comparing with another language[[1]](#footnote-1)**

**Language Name: JavaScript**

**Differences**

|  |  |  |
| --- | --- | --- |
|  | JavaScript is loosely typed while Chopin is strongly typed. There are ints and floats (decimal) in Chopin while JavaScript has one type of number. JavaScript is able to manipulate html and css and Chopin will manipulate midi type data. |  |

**Advantages / Disadvantages (in comparison with this second language)**

|  |  |  |
| --- | --- | --- |
|  | JavaScript allows you to do strange things like concatenate a number and a string together, Chopin should throw an error when data types are not set properly. Chopin will have a data type for non-negative floats which ensures impossible things such as negative pitch are not allowed.  JavaScript is able to manipulate web pages which Chopin is not able to do. JavaScript is even able to operate databases and play games, while Chopin cannot. In general, JavaScript can be used for almost anything, while Chopin has a very limited scope. |  |

|  |  |
| --- | --- |
| **Part**  **3** | **Architectural Questions** |

**Advantages**

Chopin will be able to read MIDI data from a keyboard or computer. It will also be able to manipulate the data for things like transposing, changing tempo, and ai music creation. The resulting data will be able to be exported into a DAW (digital audio workstation). Having a standard language for this will allow the musical data to be compatible with different systems.

**Strategy: C Implementation**

First, I will clearly define the syntax of Chopin. This includes specifying the grammar rules for various constructs like notes, chords, sequences, loops, and other musical elements. Chopin will have special data types for pitch, duration, velocity, and beat.

Then, I will tokenize the text into a list of tokens. I could use an if/else if/else statement to categorize the tokens.

Then, parsing involves analyzing the token list according to the grammar rules to create an Abstract Syntax Tree (AST). The AST represents the hierarchical structure of the code. Then I can translate the tree into c code.

***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.*

*[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?*

Chopin will use console(“text here”); to print to the console. It should also be able to take in a variable and print it. The compiler will see “console(…)” and recognize it as a function which can use printf() in c to perform the job. If there are no quotation marks inside of the parenthesis then the compiler will know that it is a variable to be printed, if there are quotation marks then the compiler will know to print the literal string.

*[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?*

Chopin will use curly braces to identify scope. The compiler can use a stack to keep track of multiple sets of curly braces.

**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 info in the Assignment Guide (CST8152\_Compilers\_242S--ASSAMG) 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**

1. Data type ranges and bytes: <https://www.tutorialspoint.com/cprogramming/c_data_types.htm>
2. ChatGPT was used in the implementation strategy in Part 3 <https://chatgpt.com/c/1f88492d-fd81-4d28-a756-358a63c1c001>

|  |  |
| --- | --- |
|  | * ***NOTE****: Even if you use any AI tool (ex: ChatGPT), report here, including the references used.* |

Algonquin College

Summer, 2024

1. You can use any language (different from C). Ex: Java, Go, Python, etc. [↑](#footnote-ref-1)