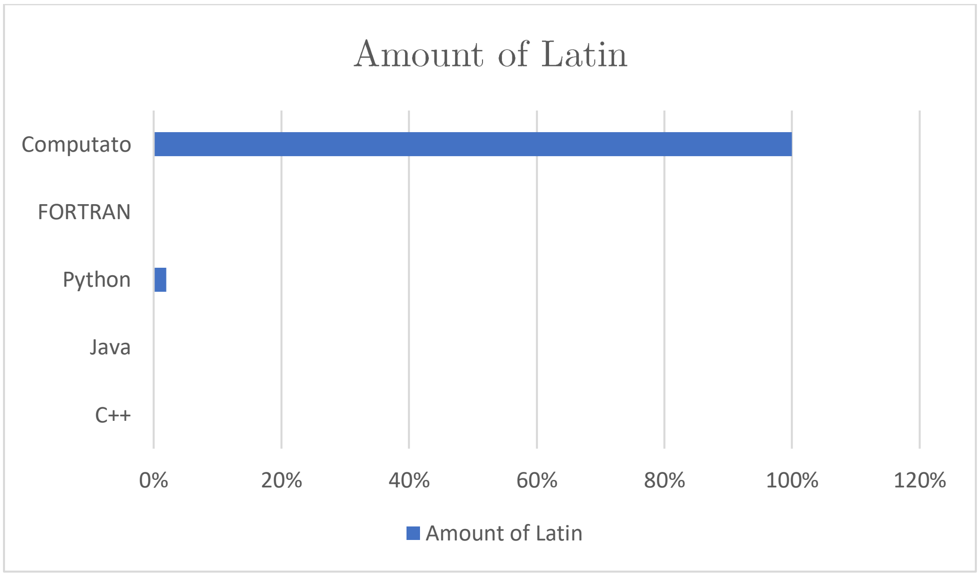
Writing Your First Program

So you, young reader, have taken an interest into the wonderful world of *Computato,* the programming language for true Romans. As someone who always enjoys plagiarism, I find no better way to introduce the language to you than through learning by example. So let's just jump right into it. What is *Computato*? Why was it made? What benefits does it serve above other programming languages? To begin, *Computato* is an object-oriented programming language developed in C++. It has all of your favorite parts of Java and Python mixed with an unreadable and terribly developed syntax. Why was it made? Because I strongly believe that there is an absence of Latin based programming languages in our society, and it occurred to me that if an ancient Roman were to accidently time travel to our future, he might feel a little out of place. It is for that reason that it's imperative we develop a language entirely in Latin. Otherwise we're just turning away that poor Roman, who'll probably resort to suicide. Do you want death on your hands? I sure don't. That's why we have *Computato*. For the third and final point, this language provides no benefits in speed, usability, availability, or documentation. But I feel as though those qualities are meaningless when compared to how better the language is with using Latin. For reference, I have put together a graph to illustrate my point:



Now, I don't want to say that it's definitely better, but looking at that chart, it's hard to say that it's not the greatest language ever written. Basically, if you start programming in *Computato*, you will learn some Latin words, girls will start becoming more attracted to you, your boss will give you a raise, and some programmers have even reported lottery winnings[1]. You'd be crazy to want to not learn this language, but just maybe you have to be crazy to use it.

[1] This is a lie. Please don't sue me I'm poor.

Enough about the language's extensive benefits and historical ingenuity, let's get into coding. Every class is a separate file with the extension '.svs', short for *slavus* meaning slave, because objects are really just your slaves. Some people have taken up with 'object civil rights' movements, 'equality for all entities regardless of computerization,' and other such mental retardations but when it comes down to it you're dictating to the class what you want it to do, and it has no say. So the first line of every class comes down to:

```
	slavus className:
```

The token 'className' is replaced with whatever you want the name of your class to be. I recommend using the filename as the class name, while not necessary, can create some grievances (slave revolts/bad import syntax).

All programs must have a class named OPERATOR. The compiler takes the OPERATOR class and runs an instance of it, and if you've programmed well the OPERATOR class should link any other necessary classes. How might you link classes? Well that's also easy! Above your class declaration you add:

```
	utendum filename
	slavus OPERATOR:
```

Originally the word was 'cum' meaning 'with' in Latin but I couldn't write a program without laughing so I dropped it in favor of 'utendum' meaning 'that which is to be used.' All it does is add the filename to the source so that it can be references by the rest of the program (not just that single class). For our 'Hello World' program that we are writing, we will be using the LIBER library, which allows output and input to a console. Importing LIBER goes like this:

```
	utendum LIBER
	slavus OPERATOR:
```

Now we can use the functions inside the LIBER class and any functions from the classes the LIBER might reference. The OPERATOR class requires a constructor defined likewise as OPERATOR. The constructor is where the compilation begins, so your program will not run if you don't have the function exactly like this:

```
	utendum LIBER
	slavus OPERATOR:
		creandus OPERATOR<>:
		^
	^
```

'Creandus' tells the compiler that this function is a constructor and is to be called when the object is instantiated. Therefore, when the class is instantiated, the OPERATOR<> function is called. The '^' denotes the end of a function/class. The contents extend from the ':' to the '^.' I made a few stylistic choices based on Ancient Latin /Ancient Greek written syntax, and one of their characteristics was to end paragraphs with a C in the margin. Because it's irrational to try and use a letter as an end delimiter in a programming language I used '^' because it kind of looks like a little downfacing C. If you don't feel that that it's a good token for the job please send me an angry email so I can be sure to consider correcting it. Also parenthesis are replaced with the carets. I just thought they looked more ancient because they were more blocky. So anytime you used parenthesis before, now you're going to use < … >. If you wrote an angry letter for the first stylistic choice I implore you to write one for this. I don't get angry letters that much and I just want to read one.

We're almost done with our first program! All we need now is to have it output a message. To do that, the LIBER library has a function called clamaLinea<> which outputs a string (verba in *Computato*) and goes to a new line. As opposed to just clama<> which does not go to a new line. Or as opposed to novaLinea<> which goes to a new line without printing anything. Angry email writer, please keep your AOL open because you're about get kicked in the balls:

```
utendum LIBER
slavus OPERATOR:
      creandus OPERATOR<>:
            LIBER~clamaLinea<|Salve Munde!|> *
      ^
^
```

Dots to determine the children functions of classes? Who needs 'em, the tilde has a much more extensive history. Quotation marks? Never heard of them, we just scratch lines in tablets to dictate what is being said, so now you will too. And semicolons to end statements? Bah! The interpunct is the separator of choice. Too bad the closest we have on our keyboards is the asterisk. You don't have to have the asterisk a space away from the line I just like it because it looks nicer. So now we are done! The program can run as it is. Fire up that compiler and take a look! Yep, it's awful. It just prints a single god damn line. While you were learning how to write five lines of code some big shot just wrote a program in COBALT that predicts the stock market two years in the future. Learn something real, you disappoint your parents with your unnecessary efforts.