David Pick
CM 2403

Python is a high level-language designed to emphasize code readability. This is accomplished by having a space sensitive syntax. While many people find this design to be irritating because it can be difficult to fix a spacing issue once they have occur, I enjoyed this feature. Being dependent on spacing forces python developers to use the same standard when they are coding. This greatly improves readability, as well as creates a standard among python programs.

Python is a multi-paradigm language, meaning that it doesn't force the developer to chose a particular style of programming, like Java does. Python allows the programmer to chose between object-oriented programming and structured programming as well as supporting parts of functional programming and aspect-oriented programming. After spending a significant amount of time working in Java and C, one of my favorite aspects of Python was the fact that I didn't have to define a main method / function when I wanted to test code in the same file. This also meant that when writing Python it took a lot less time to develop simple applications compared to Java and C.

One of my other favorite features of Python, was it's lists. Python lists are dynamically grow able and can contain multiple kinds of data types. This meant that for storing information about things a single list could be used to store all kinds of information, where as in Java a whole new class would have to be created in order to achieve the same functionality. Python's lists also include some very useful methods to make them act like other data types. They include an insert and pop method which enable them to act like a queue.

Python also has an extremely useful feature called splicing. Splicing allows you grab a subset of a list very easily, in several different ways. The programmer can specify an exact range of values to get, a value in the middle of the list to the end of the list, or the beginning of the list to somewhere in the middle. This feature becomes very useful when working with strings because they are treated as a list of characters in Python.

While I do enjoy using object-oriented programming languages, I think real power is gained by having the ability to work in multiple paradigms. In Python having the ability to have anonymous functions inside of objects was very useful. This feature made writing Tetris much simpler.

Python would be a good language of choice for any project looking to get on its feet quickly. Python is a very simple language to pick up and it allows for other developers to easily understand your code. This is very good if the project being worked on has a lot of turnover, it also makes doing code reviews much easier. It is also a good choice if the project needs to be extensible. The reflective capabilities in Python make it a good choice for someone looking to possibly change some of the underlying code to the language.

I enjoyed working in the object-oriented paradigm, objects create a really useful way to store data about one thing, such as personal information. This made the wacky quotes assignment much easier to do. Objects also allow for a huge amount of code reuse. This is really nice because it means a lot less typing for the programmer. Objects also allow for a good way of organizing code. The programmer can create several files with classes in them, import them into the main file for the program and use all of that code, without having to see it while developing. This goes right along into one of Python's greatest features, the sheer number of modules you can import and use. As shown by the XKCD shown in class you can type "import ..." and it will most

likely work. This means that if your working with any number of current technologies Python probably has a module to support it.