PYTHON

ADVANTAGES

- ✓ Python is easy to learn for even a novice developer. Its code is easy to read and you can do a lot of things just by looking at it. Also, you can execute a lot of complex functionalities with ease, thanks to the standard library.
- ✓ Supports multiple systems and platforms.
- ✓ Object Oriented Programming-driven.
- ✓ Python has a plethora of frameworks that make web programming very flexible.
- ✓ Gives rise to quick development by using less code. Even a small team can handle Python effectively.
- ✓ Allows to scale even the most complex applications with ease.
- ✓ A large number of resources are available for Python.

DISADVANTAGES

- ✓ Python is slow
- ✓ Python is not a very good language for mobile development.
- ✓ Python is not a good choice for memory intensive tasks.
- ✓ It's near impossible to build a high-graphic 3D game using Python.
- ✓ Has limitations with database access.
- ✓ Python is not good for multi-processor/multi-core work.

JAVA

ADVANTAGES

- ✓ Does not slow your computer speed
- ✓ The syntax is familiar to the myriad programmers that know any other C based language.
- ✓ Java (the platform) has a very large and standard class library, some parts of which are very well written.
- ✓ Java provides a platform for behavioral transfer from one address space to another. This is particularly evident in the dynamic class loading mechanisms of RMI.
- ✓ Automatic Memory Management implemented by Garbage Collection and No Explicit Pointers.
- ✓ No Explicit Pointers
- ✓ Explicit Interfaces
- ✓ Improving performance
- ✓ Good portability

DISADVANTAGES

- ✓ Much of the Java code as written by experienced coders turns out to be boilerplate. This has led to the charge that Java code is object-oriented Cobol.
- ✓ If you dislike OOP or used mixed paradigms, the only way to write functions is to make them class methods.
- ✓ As with all languages, getting used to the syntax conventions takes a while for those who come from other backgrounds.
- ✓ Lack of garbage collection on resources other than memory
- ✓ Large memory footprint
- ✓ Requires an interpreter. It is still difficult to deliver a self-contained application
- ✓ If you aren't careful, you can write slow programs. Of course, you can do this in any language.
- ✓ Difficult to blur distinction between class and object when desired, which increases the need/want for HOF's.

C#

ADVANTAGES:

- ✓ Common language in most Windows development projects.
- ✓ Don't need any special configurations to get a C# program to run in your Windows environment. Whether it's a web application, a Windows service, or a desktop app, C# programs are easily deployed on the network.
- ✓ C# is easy to find additional developers whether it's for a contract or full-time basis.
- ✓ Microsoft offers Team Foundation Server versions of its software and Enterprise options for large development teams.
- ✓ C# is a compiled language, which means that the code stored on a public-facing server is in binary form. If your server gets hacked, the hacker doesn't automatically have access to your source code.

DISADVANTAGES

- ✓ It's much more difficult to work with since your code must be compiled each time you make even a minor change.
- ✓ Since C# is a part of the .NET framework, the server running the application must be Windows.
- ✓ Microsoft stops supporting older .NET frameworks after a few operating systems upgrades. For instance, older Windows 2000 servers can only support .NET 2.0 applications. While having an old operating system installed seems like a mistake, many enterprise.
- ✓ If your organization uses Windows workstations and servers, .NET is the easiest to integrate. C# can be used for automation such as Windows Services or web applications.

C++

ADVANTAGES

- ✓ **Wide support**: C++ is a very popular language, and as such, many compilers and libraries already exist that are compatible with it.
- ✓ **Powerful**: Because C++ does not require a special runtime to be installed for it to run, any kind of program can be created, all the way down to low-level systems programming up to complicated GUIs.
- ✓ **Speed**: Because it is compiled, C++ gains a lot of speed. Non-compiled languages have to be interpreted at runtime, meaning every action is a 2-step process.
- ✓ **Similarity to other languages**: Many other languages like C, C#, and Java have very similar syntax to C++, making them easy to learn for those that already know C++.
- ✓ **Small standard library**: the standard library of C++ is small compared to other languages like Java, allowing the programmer to do more with less restrictions.

DISADVANTAGES

- ✓ **Unsafe**: the standard allows for many things that can cause unexpected behavior. This allows the programmer to do more, but also forces them to do more.
- ✓ **Little memory management**: C++ does very little memory management, forcing the programmer to do most of it themselves.
- ✓ **Archaic object orientation**: The object orientation system in C++ is unnecessarily basic compared to other languages.
- ✓ **Functions are not first-class types**: First class types are types that can be returned from functions, can be passed to functions, and can be constructed at runtime.
- ✓ **Unstandardized higher-level features**: Common program features such as GUIs, networking, and threading are dependent on operating system, forcing programmers to either make multiple versions of a program or include outside libraries that have already done so.

C

ADVANTAGES

- ✓ C language is a building block for many other currently known languages. C language has variety of data types and powerful operators
- ✓ C is highly portable language.
- ✓ There are only 32 keywords in ANSI C and its strength lies in its built-in functions.
- ✓ Another important advantage of C is its ability to extend itself.
- ✓ C language is a structured programming language. This makes user to think of a problem in terms of function modules or blocks. Collection of these modules makes a complete program. This modular structure makes program debugging, testing and maintenance easier.

DISADVANTAGES

- ✓ C does not have concept of OOPs, that's why C++ is developed.
- ✓ There is no runtime checking in C language.
- ✓ There is no strict type checking. For example, we can pass an integer value.
- ✓ for the floating data type.
- ✓ C doesn't have the concept of namespace.
- ✓ C doesn't have the concept of constructor or destructor.

PERL

ADVANTAGES

- ✓ The running is quick which helps in achieving the right answer in correct timeline.
- ✓ You only need to actually define the class to ensure that everything is clean.
- ✓ Multipurpose.
- ✓ Depending on the need, it may be procedural, object oriented, functional or imperative.
- ✓ The interpreted language is excellent replacement for sed/awk since the syntax is similar but with more functionality.
- ✓ File manipulation is managed by it in an excellent manner.
- ✓ The portable Perl is great for regular expressions, string manipulation as well as pattern matching.

DISADVANTAGES

- ✓ Awful syntax. In fact it is regarded as the sole language that can be uuencoded without even noticing.
- ✓ It has a poor usability factor when compared to many languages.
- ✓ The undocumented object model is unavailable in many places.
- ✓ It is not a great option for beginners; especially if they are used to handling type-free languages.
- ✓ It may be slow for many tasks such as scripting.
- ✓ The same result can be achieved in several ways which makes the code unreadable as well as untidy.
- ✓ It is not portable, with ugly libraries and poor argument handling.

PHP

ADVANTAGES

- ✓ Rapid Development
- ✓ A Framework Makes Your Application More Secure
- ✓ Easier Maintenance
- ✓ Stronger Teamwork
- ✓ The Community Is There for You

DISADVANTAGES

- ✓ Slower Execution
- ✓ General Solutions for Specific Problems
- ✓ A Framework Takes Time to Learn and Master
- ✓ Limited Visibility and Control
- ✓ A Framework Introduces an External Dependency

RUBY

ADVANTAGES

- ✓ Best Industry Standards
- ✓ Speed of Development
- ✓ Vibrant RoR Community

DISADVANTAGES

- ✓ Runtime Speed and Performance
- ✓ Lack of Flexibility
- ✓ High cost of wrong decisions in development

JAVASCRIPT

ADVANTAGES

- ✓ **Speed.** Being client-side, JavaScript is very fast because any code functions can be run immediately instead of having to contact the server and wait for an answer.

- ✓ **Simplicity.** JavaScript is relatively simple to learn and implement.

- ✓ **Versatility.** JavaScript plays nicely with other languages and can be used in a huge variety of applications. Unlike PHP or SSI scripts, JavaScript can be inserted into any web page regardless of the file extension. JavaScript can also be used inside scripts written in other languages such as Perl and PHP.

- ✓ **Server Load.** Being client-side reduces the demand on the website server.

DISADVANTAGES

- ✓ **Security.** Because the code executes on the users' computer, in some cases it can be exploited for malicious purposes. This is one reason some people choose to disable JavaScript.

- ✓ **Reliance on End User.** JavaScript is sometimes interpreted differently by different browsers. Whereas server-side scripts will always produce the same output, client-side scripts can be a little unpredictable. Don't be overly concerned by this though - as long as you test your script in all the major browsers you should be safe.

SWIFT

ADVANTAGES

- ✓ **Open source**

- ✓ **Safe.** Its syntax encourages you to write clean and consistent code which may even feel strict at times. Swift provides safeguards to prevent errors and improve readability.

- ✓ **Fast.** Swift was built with performance in mind. Not only does its simple syntax and hand-holding help you develop faster, it also lives up to its name: as stated on apple.com, Swift is 2.6x faster than Objective-C and 8.4x faster than Python.

- ✓ **In demand**. As of March 2018, it's 12th most popular language, surpassing Objective-C, Go, Scala, and R. With more than 40K stars on GitHub and 187K Stack Overflow questions, this young language is rightfully becoming one of the dominant technologies in the industry.

DISADVANTAGES

- ✓ The language is still quite young
- ✓ Swift is considered a "moving target"
- ✓ Limited talent pool
- ✓ Poor interoperability with third-party tools and IDEs
- ✓ Lack of support for earlier iOS versions