10. Popular programming languages:

1. Java

Java uses a compiler, and is an object-oriented language released in 1995 by Sun Microsystems. Java is the number one programming language today for many reasons. First, it is a well-organized language with a strong library of reusable software components. Second, programs written in Java can run on many different computer architectures and operating systems because of the use of the JVM (Java virtual machine). Sometimes this is referred to as code portability or even WORA (write once, run anywhere). Third, Java is the language most likely to be taught in university computer science classes. A lot of computer science theory books written in the past decade use Java in the code examples. So learning Java syntax is a good idea even if you never actually code in it.

Java Strengths: WORA, popularity

Java Weaknesses: Slower than natively compiled languages

2. C

C is a compiled, procedural language developed in 1972 by Dennis Ritchie for use in the UNIX operating system. Although designed to be portable in nature, C programs must be specifically compiled for computers with different architectures and operating systems. This helps make them lightning fast. Although C is a relatively old language, it is still widely used for system programming, writing other programming languages, and in embedded systems.

Strengths: Speed

Weaknesses: Memory management can be difficult to master

3. C++

C++ is a compiled, multi-paradigm language written as an update to C in 1979 by Bjarne Stroustrup. It attempts to be backwards-compatible with C and brings object-orientation, which helps in larger projects. Despite it's age, C++ is used to create a wide array of applications from games to office suites.

Strengths: Speed

Weaknesses: C++ is older and considered more clumsy than newer object-oriented languages such as Java or C#.

4. PHP

PHP uses a run-time interpreter, and is a multi-paradigm language originally developed in 1996 by Rasmus Lerdorf to create dynamic web pages. At first it was not even a real programming language, but over time it eventually grew into a fully featured object-oriented programming language. Although PHP has been much criticized in the past for being a bit sloppy and insecure, it's been pretty good since version 5 came out in 2004. It's hard to argue with success. Today, PHP is the most popular language used to write web applications. Even English 4 IT, the program you are currently using, is written in PHP ;)

Strengths: Web programming, good documentation

Weaknesses: Inconsistent syntax, too many ways to do the same thing, a history of bizarre security decisions

5. VB (or Visual Basic) Visual Basic is an interpreted, multi-paradigm language developed by Microsoft Corporation for the Windows platform. It has been evolving over the years and is seen as a direct descendant of Microsoft's old BASIC from the 1970's. Visual Basic is a good language for scripting Windows applications that do not need the power and speed of C#.

Strengths: None.

Weaknesses: Only runs in Windows

6. Python

Python is an interpreted, multi-paradigm programming language written by Guido van Rossum in the late 1980's and intended for general programming purposes. Python was not named after the snake but actually after the Monty Python comedy group. Python is characterized by its use of indentation for readability, and its encouragement for elegant code by making developers do similar things in similar ways. Python is used as the main programming choice of both Google and Ubuntu.

Strengths: Excellent readability and overall philosophy

Weaknesses: None

7 C#

C# is a compiled, object-oriented language written by Microsoft. It is an open specification, but rarely seen on any non-Windows platform. C# was conceived as Microsoft's premium language in its .NET Framework. It is very similar to Java in both syntax and nature.

Strengths: Powerful and pretty fast

Weaknesses: Only really suitable for Windows

8. JavaScript

JavaScript is an interpreted, multi-paradigm language. A very strange one too. Despite it's name, it has nothing whatsoever to do with Java. You will rarely, if ever, see this language outside of a web browser. It is basically a language meant to script behaviors in web browsers and used for things such as web form validation and AJAX style web applications. The trend in the future seems to be building more and more complex applications in JavaScript, even simple online games and office suites. The success of this trend will depend upon advancements in the speed of a browser's JavaScript interpreter. If you want to be correct, the real name of this programming language is ECMAscript, although almost nobody actually calls it this.

Strengths: it's the only reliable way to do client-side web programming

Weaknesses: it's only really useful in a web browser

9. Perl

Perl is an interpreted, multi-paradigm language written by Larry Wall in 1986. It is characterized by a somewhat disorganized and scary-looking syntax which only makes sense to other PERL programmers ;) However, a lot of veteran programmers love it and use if every day as their primary language. 10 years ago, Perl was more popular than it is today. What happened? A lot of newer programmers and even old Perl programmers (such as myself) have switched to other languages such as PHP, Python, and Ruby. Perl is perhaps still the best language for text processing and system administration scripting. I personally do not recommend it however as a primary programming language.

Strengths: text processing and system administration

Weaknesses: strange syntax, and perhaps too many ways to do the same thing

10. Ruby

Ruby is an interpreted, object-oriented language written by Yukihiro Matsumoto around 1995. It is one of the most object-oriented languages in the world. Everything is an object in Ruby, even letters and numbers can have method calls. It's a great language to learn if you love objects. The only negative is that it's love of object-orientation makes it a bit slow, even for an interpreted language.

Strengths: Perhaps the world's most object-oriented language

Weaknesses: its superior object model comes at a price... namely speed

11. What is the difference between C# and .NET?

- .NET isn't just a **library**, but also a **runtime** for executing applications.
- The knowledge of C# implies some knowledge of .NET (because the C# object model corresponds to the .NET object model and you can do anything interesting in C# just by using .NET libraries). The opposite isn't necessarily true as you can use other languages to write .NET applications.

The distinction between a *langauge* and *runtime* and *library* is more strict in .NET/C# than for example in C++, where the language specification also includes some basic library functions. The C# specification says only a very little about the environment