Topic: 1.5.4 Language translators

The final type of system software that you need to know is translator software. This is software that allows new programs to be written and run on computers, by converting source code (human readable) into machine code. There are three types that we'll cover in a lot more detail shortly:

- Assembler - converts assembly code into machine code
- Interpreter - converts 3rd generation languages such as JavaScript into machine code one line at a time
- Compiler - converts 3rd generation languages such as C++ into machine code all at once

Assembler

An **assembler** translates assembly language into machine code. Assembly language consists of mnemonics for machine opcodes so assemblers perform a **1:1 ratio** translation from mnemonic to a direct instruction. For example:

```
LDA #4 converts to 0001001000100100
```

Conversely, one instruction in a high level language will translate to one or more instructions at machine level.

Advantages of using an Assembler	Disadvantages of using Assembler
• Very fast in translating assembly language to machine code as 1 to 1 relationship • Assembly code is often very efficient (and therefore fast) because it is a low level language • Assembly code is fairly easy to understand due to the use of English-like mnemonics	• Assembly language is written for a certain instruction set and/or processor • Assembly tends to be optimized for the hardware it's designed for, meaning it is incompatible with different hardware • Lots of assembly code is needed to do relatively simple tasks, and complex programs require lots of programming time

Topic: 1.5.4 Language translators

Compiler

A **Compiler** is a computer program that **translates code** written in a high level language to a lower level language, object/machine code. The most common reason for translating source code is to create an executable program (converting from a high level language into machine language).

Advantages of using a compiler	Disadvantages of using a compiler
• Source code is not included, therefore compiled code is more secure than interpreted code • Tends to produce faster code than interpreting source code • Produces an executable file, and therefore the program can be run without need of the source code	• Object code needs to be produced before a final executable file, this can be a slow process • The source code must be 100% correct for the executable file to be produced

Interpreter

An interpreter program executes other programs directly, running through program code and **executing it line-by-line**. As it analyses every line, an interpreter is slower than running compiled code but it can take less time to interpret program code than to compile and then run it — this is very useful when prototyping and testing code. Interpreters are written for multiple platforms, this means code written once can be run immediately on different systems without having to recompile for each. Examples of this include flash based web programs that will run on your PC, MAC, games console and Mobile phone.

Advantages of using an Interpreter	Disadvantages of using an Interpreter
• Easier to debug(check errors) than a compiler • Easier to create multi-platform code, as each different platform would have an interpreter to run the same code • Useful for prototyping software and testing basic program logic	• Source code is required for the program to be executed, and this source code can be read making it insecure • Interpreters are generally slower than compiled programs due to the per-line translation method.

Topic: 1.5.4 Language translators

Topic: 1.5.4 Language translators

Few of the high-level language programs may be partially compiled and partially interpreted, such as Java. Typical Java environment consists of two programs: Java compiler and Java Virtual Machine. Java compiler takes the source code written in Java programming language, together with precompiled libraries, and compiles programs written in Java programming languages into class files containing Java byte-code.

The Java Virtual Machine takes the byte code prepared by the Java compiler and executes it. The byte-code itself is platform-independent; it is the responsibility of the Java Virtual Machine implementation to execute the program in the byte-code form on the real computer.

So, the java code is partially compiled for optimization by the programmer. The user runs the code on a Java virtual machine on their computers that interprets the java code for the users' computer specific architecture.