S1.3 What is the difference between a compiler and an interpreter?

Compilers and Interpreters are two strategies for obtaining runnable code (traditionally called as *machine code*) from a program written in some programming language, say, high-level source language.

Compilers translates a program written in *source language* to *target language.*

- Compilation can be slow because it is difficult to translate from a high-level source language to low-level languages
- *Target language* of compilers is *machine code*, which the computer processor knows how to execute

Interpreter reads the program and does whatever computation code describes.

- Given a program, interpreter can start running it without the time spent to compile it
- Code is more portable to different hardware architectures (any hardware architecture to which interpreter has been ported)
- It is slower than hardware execution of same computation because interpreter has to do many operations to figure out what it is supposed to be doing

No	Compiler	Interpreter
1	Complete program is given as input to the compiler (in human readable format)	Interpreter takes single step as input
2	Intermediate object code is generated	No intermediate object code is generated
3	Conditional control statements execute faster	Conditional control statements execute slower
4	Comparatively, more memory is required	Memory requirement is less
5	Errors are displayed after entire program is checked	Errors are displayed for every instruction interpreted
6	Example: C Compiler	Example: Python