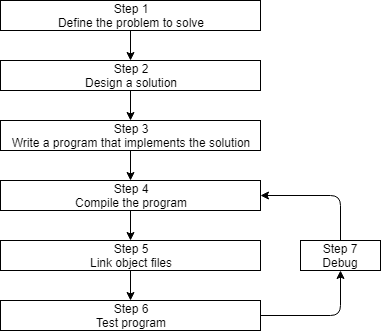
C++ Programming

**Process of development:**



**Object file**: Compiler translating C++ code into machine language instructions.  
Named “file.o”, ‘file’ is the name of the .cpp file and .o is the extension.\

**Linker:** Links all the .o ‘object’ files and produces the output file (usually an executable). Process is called **linking.**

A diagram of a linker

Description automatically generated

**Build:** Process of converting source code files into executables.   
(*build automation:* ***make / build2****)*

**Clean:** Removes all cached objects and executables so the next time the project is built, all files will be recompiled, and a new executable produced.

**Rebuild:** does a “clean”, followed by a “build”.

**Compile:** Recompiles a single code file (regardless of where it has been cached previous). This option does not invoke the *linker* or produce an *executable*.

**Run/Start:** executes the executable from a prior build. Some IDEs (e.g. Visual Studio) will invoke a ‘***build’*** before doing a “***run***” to ensure you are running the latest version of your code.

**Note:** to compile our programs we will typically choose the “build” (or “run”) option in our IDE to do so.

**Build Configuration (**also ***build target:*** collection of project settings that determines how your IDE will build your project. Includes: exe. Name, library/code file directories, keep/strip debugging information, how much the compiler optimizes the program.

**Debug configuration:** Helps debug the program, used when writing programs. *Turns off all optimisations, includes debugging information (*makes programs larger and slower, but easier to debug). **Active by default.**

**Release configuration:** Used when releasing public build. Optimised for *size* and *performance,* doesn’t contain extra debugging information. Useful for testing ***code performance.***