High level code and machine code | Part C

We now need to look at things from the computers point of view. Unfortunately, computers only understand binary, but that's trickier for humans. High level languages, such as Scratch, and Python, are more convenient for humans, but computers can't understand them. So, we need to have a translator to convert the high level language into something that the computer can make sense of.

There are two ways we can do this. We can either check the whole program is correct and can be understood, and convert it into low level code ready to be run. This is called compiling. The problem is, if I find an error, I need to go back through the program to find it. Then I will need to compile it again. The other way is to check the program line by line. This is called interpreting. When we reach a line that cannot be interpreted, because it doesn't make sense, we have to stop the program executing, and give an error. Compiling is preferable, because we find errors before the program is executed.

I have here a crazy mouse catching contraption. I've built the whole thing. If I test it, I can check that the whole system works. This is rather like compiling a program. The other way I could test it is I could build a small section and then test that separately, and then move onto the next section and test that. That's like interpreting. An assembler is a program that specifically translates assembly language into binary machine code. Some programming languages use a combination of translators together.