Arrays:

1. By capturing the length of the word, the length of the sentences, the position of punctuation etc, we can implement a program that takes a paragraph and determines what level of reading it is.
2. Cryptography – scrambling or encoding a message in a way that it can be sent safely and securely even though it may get intercepted.

Compiling:

1. When we use the make command to make an executable file, we are actually using the compiler called clang. When we run it, we get an output file named a.out, which means assembler output. If we run ./a.out, then the program itself will run.
2. Essentially, the make command runs the clang program, and then renames a.out to whatever your file name is.
3. Command line arguments are just words or prompts that you enter on the command line that modify the behavior of your program.
4. To do the assembling and renaming part of the program which make does, without using make, we need to write **clang –o hello hello.c**
5. An issue with the clang command is that if you are using third party libraries in your program, then clang will not recognize it. You will need to mention it explicitly.
6. To mention it explicitly, on the command line, we type **clang –o filename filename.c –lcs50**
7. The **–l** means library.
8. The # symbol means preprocessor directive.
9. To turn source code to binary, there are 4 steps:
   1. Preprocessing – When we write #include<headerfile>, the header file actually contains the prototypes for whatever functions we need. So when we run the clang command, what happens is those functions are found and directly copy pasted into your code. This is the preprocessing stage. Preprocessing converts those #include lines into the prototypes of the functions we need.
   2. Compiling – After the required functions are copy pasted into your code from the previous step, in this step, your code is converted into assembly language (As in, MV Left, ADD string, SUB string etc.).
   3. Assembling – In this step, the assembly code from above is converted into 0’s and 1’s, i.e, binary.
   4. Linking – When we include our header files like stdio.h, there is actually a file called stdio.c that exists somewhere. That is also converted into binary. The final step includes “linking” this file and the binary file of the program we wrote.
10. If you try to reverse binary code to original source code, which is technically possible, you will still lose a lot of the variable names and function names, to the point that just writing the program yourself is easier than performing the entire reverse engineering process.

Debugging: