PP8 Tasks

Phil Reinartz

June 6, 2025

1 Task 2

How do you pass a file name to a program using the -i and -o options?

That is done, by simply passing the name of the file to the program. The selection of what which file is, is done with the options: "-i and -o". In the script itself is defined, that a string that is written behind the -i is saved at the "*infile pointer" and behind the -o is saved at the "*outfile pointer". The addresses of the filenames are now saved on these pointers. Now you could use the filenames in the program. For example if you want to open these files.

What are typical use cases for parameters versus flags? How do the differ from one another?

I think as we have seen in booth programs, parameters differ from flags. Flags are predefined values, which could be set true or false (1 or 0). For example, if you want to check if someone used the right characters, you could give the possible characters a flag , which than is 0 if an false character has been put. This is pretty useful, because the flags are rather easy to use for the following logic. A parameter is another story, it's nothing else than giving the program a string ("Parameter") under a predefined option. This is very useful, especially if you want to use several parameters, the "options" help you to remember which parameter belongs to which place and also helps you to keep your program clean. I don't know what I should tell about the difference, because for me they've not much to do with each other.

2 Task 3

Why is a run-to-completion (batch) approach often preferable to interactive input

Interessting question, I think that run-to-completion and interactive input are two different stories. However, the run-to-complemention is used more often, because the function "to read something from a file" is very nescessary for a programmer. Honestly I think that the interactive input is rarely used, but if you have to get some "interactive" response it's the way to go. So I think I answered the question, by not directly answering it. A function that is used more often is also preferable I think.

3 Task 4

What is the difference between redirecting to stdin and explicitly opening a file with fopen

The difference between those methodes is large. By reading from stdin, the program does nothing other than reading a string from stdin, which is our terminal buffer. Because we redirected the string from input.txt to our script, the string gets into stdin and than into our program. The fopen works differently, the program opens with fopen a file(in read mode), for Example "data.txt" and reads than the string, that is written inside saves it on a Identifier. Again both a different functions with both different advantages and disadvantages.

4 Task 5

Explain in your own words what the encryption and decryption processes are doing in both ciphers.

Ceasar:
The Ceasar encryption works on a simple principal. It is taking the characters piece by piece, for Example 'A' and switches them with another character, that is a defined distanceaway (For example 3). It does this simply by adding a Number to the ASCI value.

$$\texttt{A} = 65 \rightarrow 65 + 3 = 68 \rightarrow 68 = \texttt{D}$$

The Encryption works the same, but backwards. Instead of adding a number to the ASCII value, it gets subtracted. In the 'demo' code is also a mechanism included, that ensures that only Alphabetic chars are emitted.

Xor:
The Xor encryption works with an encryption key, that is given to the function (with the character). They than get both worked out to a new character, that usually can't be converted backwards without the decryption key. But in our case we use a prototype function, that has the same encryption and decryption key.