# Computer Architecture

## Using if-else and switch

Which conditional expression (if-else, switch) is best suited to solve the following problems and why? To illustrate your choice, write a small C function that implements the problem using if or switch. You can assume that helper functions are available for the functionality that is not related to the conditional expression (e.g. write\_log(), print\_error(), ...)!

### Use different algorithm for positive and negative numbers

Case 1:

Case 2:

The if statement will be satisfied if its conditions returns True and then skips its else block.

The switch code first needs to make an indirect run, then check its cases.

A break will then have to be mentioned, as it would otherwise fall through, running through cases until it reaches a break.

=======================================================================

The if-else doesn’t require any indirect jumps, and doesn’t require any breaks for something ***not to*** occour, but rather if ***something else needs to occour.***

The if-else statement is the way to go for this example.

=======================================================================

### Print different messages to the log based on a status code

This is a case were the switch is useful.

With possibly hundreds of status codes, where ***each having the same possibility to occour***, the switch statement will then be the least confusing one to read as a programmer, with its performance being neglectable larger than the if-else statement.

…

…

====================================

The switch condition is the way to go for this case.

====================================

### Return from a function immediately when a parameter is not in the expected range:

This is how I interpret this case:

Contra:

With this interpretation, the switch would have to make an indirect jump to its tables, finding that after its jump to the first address it finds a case were the condition is False. It then returns.

The if else statement would only have to check whether, its condition was met and then jump to the block that returns.

=======================================================================

No indirect jumps would occour with an if statement. The switch statement here becomes clumsy for the programmer.   
If-else is the way to go for this case.

=======================================================================

# Mangler at skrive C implementationerne af disse tilfælde

## Applying Amdahl’s law

You have measured the runtime of your program and it looks like it’s not fast enough. After profiling it, you found that there is a bottleneck in the code which can be improved by replacing a sequential code segment with a concurrent implementation. Your goal is to speed up the whole program by at least 50%. Assume that it’s possible to parallelize the bottleneck code to any degree and you get linear speedup with each new CPU core.

How much speedup can be achieved with an 8 core CPU? At least how many cores are needed to realize the advertised 50% speedup? The bottleneck code is 65% of the total sequential runtime.

Let’s note the facts that’s given to us.

The CPU has 8 cores.

The goal is to achieve an at least 50% speed boost for our program.

The code that is desired to be increased by a 50% speedup makes up 65% of the total sequential runtime.

Our task is to.

Speed up the program by *at least* 50 %.

Check how much the code can be speed up with an 8 core CPU.

Check how many cores are needed to fulfill the goal of a 50% speedup.

Amdahl’s Law is as followed.

We assume that the speedup factor comes from the CPU delegating tasks to different cores.

The desired time taken is a 50% speedup, for our current 100% time taken:

Ligningen løses for k vha. WordMat.

With the requirement being at least 50%, the least amount of cores needed is:

===============

===============

We then want to check how much of a speedup, the code can achieve.

The formular showing how fast the code can run, then means that our final performance is 43,125% of our initial time, increasing our code by (100% - 43,125%)

============================================

============================================