Week 5

Exercise 40

../40/40.txt

```
1. Pointer variables and arrays
They are very similar, in that declaring a size_t array of size 10, i.e. size_t array
    [10] is actually just a pointer to the first element of that array (i.e. array[0]
     = *array). The difference lies in the fact that the location that an array
    points to is immutable, whereas a pointer variable can be changed.

2. pointer variables and reference variables
how element [3][2] is reached (make a clear drawing)

        a: for the variable 'tt(int array[20][30])'
        b: for the variable 'tt(int *pointer[20])'.

Your drawing should clearly show how the memory accessed by array and pointer is (
    differently) organized;

3. what is meant by 'pointer arithmetic';
4. explain why accessing an element in an array using only a pointer variable is
    preferred over using an index expression. By implication: why are repetitions
    iterating over a series of elements using a pointer-type loop control variable
    preferred over repetitions in which the loop control variable is, e.g., a size_t
    type variable?
```

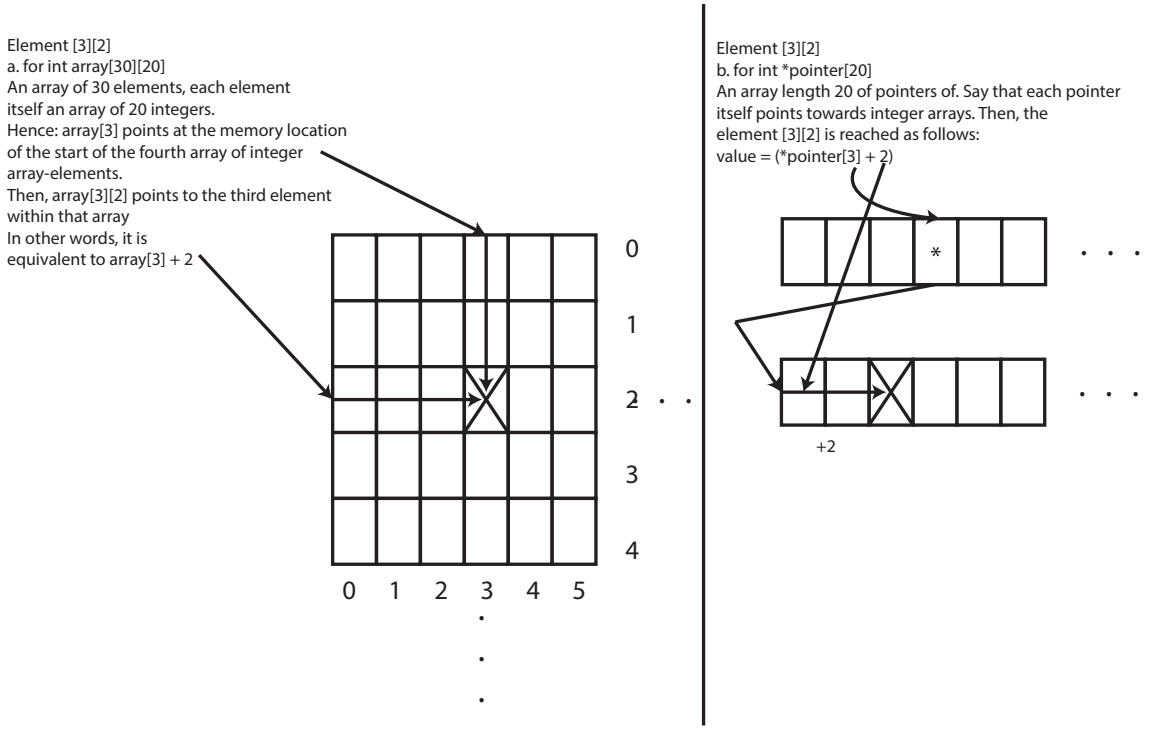

Figure 1: Illustration

Exercise 41

../41/main.cc

```
1 // Programming in C/C++
2 // Week 5: Assignment 41
3 // Tjalling Otter & Emiel Krol
4 // Main file
```

```
5
6   #include "main.ih"
7
8   int main(int argc, char const *argv[])
9   {
10    extern char **environ;
11
12    for (size_t index = 0; index != argc; ++index)  // For elements of argv (0-argc)
13      cout << environ[index] << '\n';               // print associated env var
14
15    for (size_t index = 0; environ[index] != nullptr; ++index)  // For all elements
16      cout << argv[index] << '\n';                              // of environ[] print
17  }                                                             // associated argv
```