Programmering og problemløsning 5100-B1-2E16: Ugeseddel #5

Due on Wednesday, October 12, 2016

Mehrdad Khodaverdi ctm546@alumni.ku.dk

October 12, 2016

i5-3

`arraySort l` starts by matching if `l` is an empty or an array that has only one element. If so return the empty or 1 element array.
For all other cases go through the array `l` and return a new array where all the elements are smaller then the first element of `l`. The result gets feeded to `arraysort` till the smallest element in `l` is found.

```
l |> Array.filter ((>)(Array.head l)) |> arraySort
```

This smallest element we place in front of the first element in `l` and assign it to `temp`.

```
let temp = [|Array.head l|] |> Array.append
```

Next we start filtering l for all elements bigger or equal the first element in `l`

```
l |> Array.filter ((<=)(Array.head l))
```

we find the tail and feed it to `arraysort` so it can get sorted as well.

```
|> Array.tail |>arraySort
```

and one by one we attach the next smallest element behind the last smallest element.

```
|> Array.append temp
```

```
let rec arraySort l =
    match l with
    |[||]  |[|_|] -> l
    | _ ->
        let temp = [|Array.head l|] |> Array.append
         (l |> Array.filter ((>)(Array.head l)) |> arraySort)
        l |> Array.filter ((<=)(Array.head l)) |> Array.tail
         |>arraySort |> Array.append temp
```

i5-4

ArraySortD is a bubbleSort algorithm. Here it uses two loops to sort a list, here in this case an array and i use one for-loop and one while-loop. The for loop runs from 1 to the number of elements in the given array. And the while loop runs depending on how the array is sorted at start.
The for loop start looking at the first element then next and so on. The while loop pushes elements that are smaller then the previous element in the array to the left. So to speak to the head of the array. This is done by using a swap method.

```
let tmp = a.[j]
a.[j] <- a.[j-1]
a.[j-1] <- tmp
```

The function is given by:

```
let arraySortD (a : 'a []) =
    for i = 1 to a.Length - 1 do
        let mutable j = i
        while j >= 1 && a.[j] < a.[j-1] do
            let tmp = a.[j]
            a.[j] <- a.[j-1]
            a.[j-1] <- tmp
            j <- j - 1
```