Lab 1: Sorting

COSC 3020: Algorithms and Data Structures

Lars Kotthoff
larsko@uwyo.edu

11 September 2019

Instructions

Attempt to finish the tasks below during the lab time. You have until Friday, 13 September 2019, 23:59h to submit the solutions to WyoCourses. You may ask your TA for feedback before submitting, but this feedback will be qualitative only. When submitting Javascript code, submit only the Javascript code, no HTML around it.

You may *not* use external libraries in your code unless explicitly stated.

1 Reverse Insertion Sort

Consider the code for insertion sort we covered in the lectures:

```
function insertionSort(arr) {
  for(var i = 1; i < arr.length; i++) {
    var val = arr[i];
    var j;
    for(j = i; j > 0 && arr[j-1] > val; j--) {
      arr[j] = arr[j-1];
    }
    arr[j] = val;
  }
  return arr;
}
```

Write a function that tests this code, i.e. calls the function with an unsorted array of numbers and checks that it is sorted afterwards.

Now change the function such that it works from the end of the array rather than the beginning, `insertionSortReverse()` – only the direction of iterating over the elements is reversed, the array is still sorted the same way (ascending). Test your new function.

Submit your complete code with the original function, the new function `insertionSortReverse()`, and functions to test your code.

Total 6 points.

2 Average-Case Time Complexity of Insertion Sort

In the lectures, we covered that insertion sort has best-case time complexity of $\Theta(n)$ and worst-case time complexity of $\Theta(n^2)$. What is the average-case time complexity?

Hint: Think about what happens in each iteration of the loop, and how often the loop is executed. Keep in mind that for asymptotic analysis we don't care about constant factors.

Describe your reasoning and the conclusion you've come to. Your reasoning is most important – you can easily find the answer, but you need to demonstrate that you've understood the concept.

Total 4 points.

Testing

You can use `jsverify` (`https://jsverify.github.io/`) to automatically test your code. Assuming that your code is in a file `code.js`, you can use the following code to test it:

```
const fs = require('fs');
const jsc = require('jsverify');

eval(fs.readFileSync('code.js')+'');

const testSort =
    jsc.forall("array nat", function(arr) {
      var a1 = JSON.parse(JSON.stringify(arr));
      var a2 = JSON.parse(JSON.stringify(arr));
      return JSON.stringify(insertionSortReverse(a1)) ==
          JSON.stringify(a2.sort(function(a, b)
              { return a - b; }));
    });
jsc.check(testSort);
```

This is optional, but highly encouraged. How exactly to get this code to run is left as an exercise to the reader. No points, but the satisfaction of knowing how to do property-based testing.