ADS 2021: Week 3 Exercises

Exercises for Algorithms and Data Structures at ITU. The exercises are from *Algorithms, 4th Edition* by Robert Sedgewick and Kevin Wayne unless otherwise specified. Color-coding of difficulty level and alterations to the exercises (if any) are made by the teachers of the ADS course at ITU.

1.4.5, abcd - Green Give tilde approximations for the following quantities:

a. $N + 1$
b. $1 + 1/N$
c. $(1 + 1/N)(1 + 2/N)$
d. $2N^3 - 15N^2 + N$

1.4.6 - Green Give the order of growth (as a function of N) of the running times of each of the following code fragments:

a)

```python
# Python
sum = 0
n = N
while n > 0:
  for i in range(0, n):
    sum += 1
  n //= 2
```

```java
// Java
int sum = 0;
for (int n = N; n > 0; n /= 2)
          for(int i = 0; i < n; i++)
             sum++;

.
```

b)

```python
# Python
sum = 0
i = 1
while i < N:
  for j in range(0, i):
    sum += 1
  i *= 2
```

```java
// Java
int sum = 0;
for (int i = 1; i < N; i *= 2)
            for (int j = 0; j < i; j++)
               sum++;

.
```

c)

```python
# Python
sum = 0
i = 1
while i < N:
  for j in range(0, N):
    sum += 1
  i *= 2
```

```java
// Java
int sum = 0;
for (int i = 1; i < N; i *= 2)
           for (int j = 0; j < N; j++)
              sum++;

.
```

1.4.10 - Green Modify binary search so that it always returns the element with the smallest index that matches the search element (and still guarantees logarithmic running time).

1.4.12 - Green Design a program that, given two sorted arrays of N int values, prints all elements that appear in both arrays, in sorted order. The running time of your program should be proportional to N in the worst case.

1.4.21 - Green *Binary search on distinct values.* Explain how you can change StaticSETofInts (see page 98 of the book) to get the running time of contains() to be guaranteed $\sim lgR$, where R is the number of different integers in the array given as argument to the constructor.

1.4.28 - Green Stack with a queue. Design a stack with a single queue so that each stack operation takes a linear number of queue operations. Hint: To delete an item, get all of the elements on the queue one at a time, and put them at the end, except for the last one which you should delete and return. (This solution is admittedly very inefficient.)

1.4.5, efg - Yellow Give tilde approximations for the following quantities:
e. $lg(2N)/lgN$
f. $lg(N^2+1)/lgN$
g. $N^{100}/2^N$

1.4.1 - Yellow Show that the number of different triples that can be chosen from N items is precisely $N(N-1)(N-2)/6$. Hint: Use mathematical induction.

1.4.24 - Yellow Suppose that you have an N-story building and plenty of eggs. Suppose also that an egg is broken if it is thrown off floor F or higher, and unhurt otherwise. First, devise a strategy to determine the value of F such that the number of broken eggs is ∼lg N when using ∼lg N throws, then find a way to reduce the cost to ∼2 lg F.

1.4.25 - Red Consider the previous question, but now suppose you only have two eggs, and your cost model is the number of throws. Devise a strategy to determine F such that the number of throws is at most $2\sqrt{N}$, then find a way to reduce the cost to $\sim c\sqrt{F}$. This is analogous to a situation where search hits (egg intact) are much cheaper than misses (egg broken).

1.4.18 - Red Design a program that, given an array a[] of N distinct integers, finds a local minimum: an index i such that a[i-1] > a[i] < a[i+1]. Note that for the edge case i = 0 it is only necessary that a[0] be smaller than a[1], and for a[n-1] it only needs to be smaller than a[n-2]. Your program should use $\sim 2lgN$ compares in the worst case.

1.4.29 - Red Steque with two stacks. Design a steque (a stack-ended queue or steque is a data type that supports push, pop, and enqueue) with two stacks so that each steque operation takes a constant amortized number of stack operations.

1.4.30 - Red Deque with a stack and a steque. Design a deque (a double-ended queue that supports pushLeft, pushRight, popLeft and popRight) with a stack and a steque so that each deque operation takes a constant amortized number of stack and steque operations.

1.4.31 - Red Deque with three stacks. Design a deque with three stacks so that each deque operation takes a constant amortized number of stack operations.