Module 0: Taste of Algorithmic Concepts **[RAD 09/2023]**

Instruction: The following exercises are to be performed in groups (each with approximately 6 students). Each group is expected to post the solution to each problem in the shared Google doc. These inputs will be used as a basis for providing feedback and a recap of key concepts.

1. **Algorithmic Problems:**
   a. Select and discuss a real-world problem in which only the best solution will do. Then come up with one in which approximately the best solution is good enough.
   b. Describe a real-world problem in which sometimes the entire input is available before you need to solve the problem, but other times the input is not entirely available in advance and arrives over time.

a. Sum two numbers algorithm - Route planning
b. Sum two numbers algorithm - Real-time route planning

2. **Algorithm Characteristics/Requirements:**
Which characteristics of an algorithm (see slide 4) do the procedures a - d below have and which do they lack?

a) **procedure** *double*(n: positive integer)
while $n > 0$
$n := 2n$

b) **procedure** *divide*(n: positive integer)
while $n \geq 0$
$m := 1/n$
$n := n - 1$

c) **procedure** *sum*(n: positive integer)
$sum := 0$
while $i < 10$
$sum := sum + i$

d) **procedure** *choose*(a, b: integers)
$x :=$ either a or b

a. As this is a while loop the “algorithm” would never end as the integer would always be greater than 0. Not a finite algorithm!
b. This algorithm is unambiguous as it merely divides the positive integer until it becomes negative. Downside is that it is stored as a variable *m* that will be rewritten every time. At the same time it could also divide by zero, which would produce an error.
c. Undefined pre-condition. It will never sum any valid input, since *i* is undefined.
d. Ambiguous. Will choose either a or b!?!?!?!?!?!

3. **Algorithm Problem Specification:**
   a. Which of the following are valid instances of the search problem?
      i. sequence $\langle 2, 8, 10, 9, 15, 20, 14\rangle$, and an integer, $\langle -20\rangle$
      ii. sequence $\langle -2, 800, 10, 90, 14\rangle$, and an integer, $\langle 100\rangle$
      iii. sequence $\langle 5, a, 4, 9, 15\rangle$, and an integer, $\langle a\rangle$
      iv. sequence $\langle 5, 98, 4, 9, 15\rangle$
      v. sequence $\langle\rangle$ and integer, $\langle 5\rangle$

I. Invalid
II. Valid (notFoundException)
III. Invalid due to string
IV. Missing request in the form of **v**
V. Invalid as the sequence is empty and needs a length of n

   b. Consider the problem of adding two n-bit binary integers, stored in two n-element arrays A and B. The sum of the two integers should be stored in binary form in an (n+1)-element array C. Specify the problem formally.

Sum two n-bit binary integers

Preconditions:
n: a positive integer
A: An n-bit binary integer array with n elements
B: An n-bit binary integer array with n elements

Postconditions:
C: An (n+1)-bit binary integer array with (n+1) elements

Constrains:
Inputs (and other variables) have not changed

   c. Specify each of the following tasks as an algorithmic problem (including pre-conditions, post-conditions, and constraints). Indicate whether a potential solution to the problem involves: sorting or searching operations.
      i. **Route optimization in GPS navigation**: Calculate optimal routes in real-time, considering factors like traffic, road closures, and shortest distance.

I. Route optimization in GPS navigation

Preconditions:
L (location) : a float array A with 2 elements, such that
-90 < A[0] < 90, -180 < A[1] < 180 (latitude, longitude)
D (destination): a float array B with 2 elements, such that
-90 < A[0] < 90, -180 < A[1] < 180 (latitude, longitude)

n (number of intersections): a positive integer
DM (distance matrix) : an array of length n of positive float arrays with n elements, such that
TM (time matrix) : an array of length n of positive float arrays with n elements, such that

Postconditions:
I (number of intersections the route passes through) : a positive integer or zero
R (route) : an (I+2)-length tuple/integer array, such that
Output is a tuple at position R[0] and R[i+2-1]

Constrains:
Inputs (and other variables) have not changed

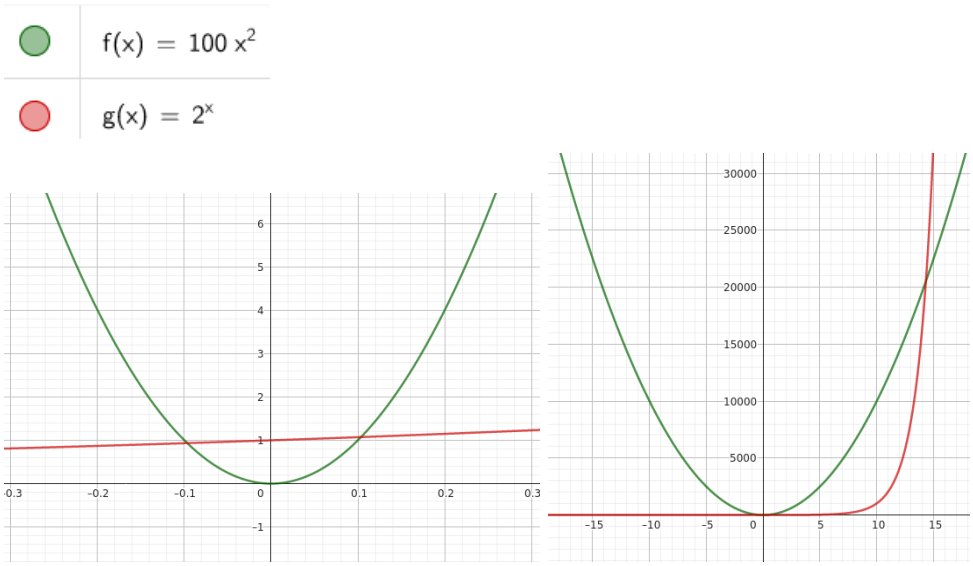
4. Algorithmic Efficiency

a. What is the smallest value of n such that an algorithm whose running time is $100n^2$ runs faster than an algorithm whose running time is 2^n on the same machine?

b. Suppose that for inputs of size n on a particular computer, algorithm A runs in $8n^2$ steps and Algorithm B runs in $64n \log_2(n)$ steps. For which values of n does algorithm A beat algorithm B?

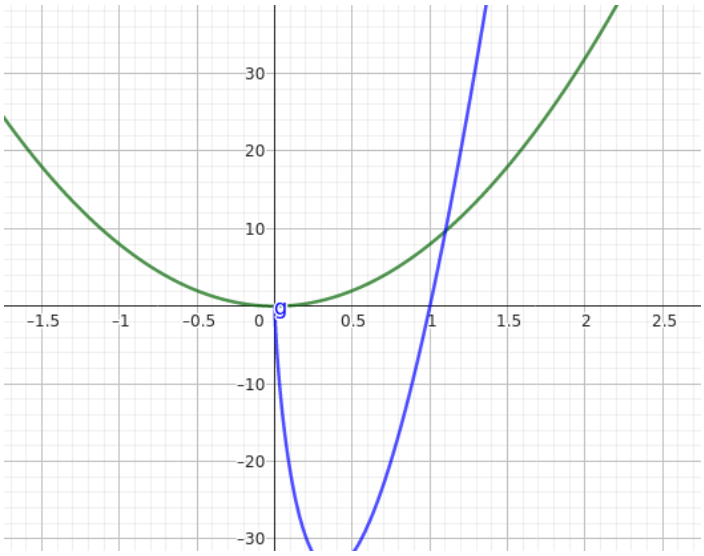
c. For each function $f(n)$ and time t in the following table, determine the largest size n of a problem that can be solved in time t , assuming that the algorithm to solve the problem takes $f(n)$ microseconds.

a. Between 0<n<0.10366



b. Between 1.1<n<43

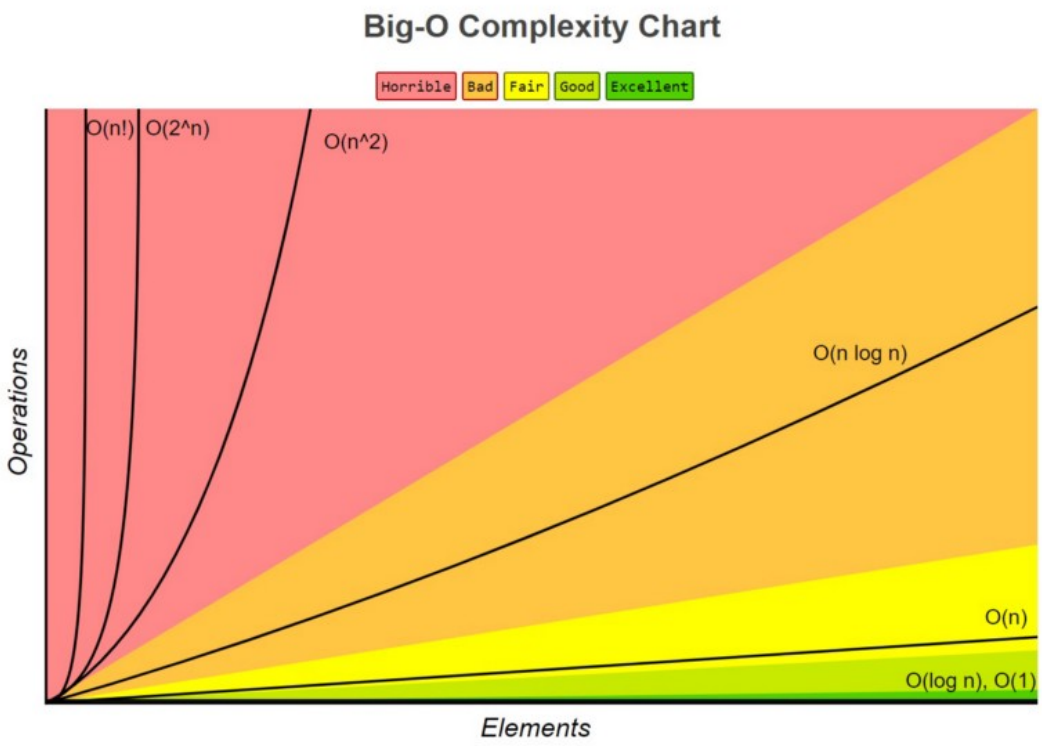
	$f(x) = 8x^2$
	$g(x) = 64 \times \log_2(x)$



Den grønne er bedre end den blå efter 2 steps

c. The largest size *n* of a problem that can be solved in time *t* is

	1 second	1 minute	1 hour	1 day	1 month	1 year
$\lg n$	2^{10^6}	$2^{6\times10^7}$	$2^{36\times10^8}$	$2^{864\times10^8}$	$2^{2592\times10^9}$	$2^{315\times10^{11}}$
$\sqrt{n}$	10^{12}	36×10^{14}	1296×10^{16}	746×10^{19}	672×10^{22}	992×10^{24}
n	10^6	6×10^7	36×10^8	864×10^8	2592×10^9	315×10^{11}
$n \lg n$	627×10^2	28×10^5	133×10^6	276×10^7	72×10^9	796×10^9
n^2	10^3	7746	6×10^4	294×10^3	161×10^4	561×10^4
n^3	10^2	391	1533	4421	13737	31582
2^n	20	25.8	31.7	36.3	41.2	44.8
$n!$	9	11	12	13	15	16


Big-O Complexity Chart
Horrible
Bad
Fair
Good
Excellent
O(n!)
O(2^n)
O(n^2)
O(n log n)
O(n)
O(log n), O(1)
Operations
Elements