# Sabancı University
# Faculty of Engineering and Natural Sciences

## CS301 – Algorithms

## Homework 1

Due: April 2, 2023 @ 23.55 (Upload to SUCourse - **no late submission**)

**PLEASE NOTE**:

Provide only the requested information and nothing more. Unreadable, unintelligible and irrelevant answers will not be considered.

You can collaborate with your TA/INSTRUCTOR ONLY and discuss the solutions of the problems. However you have to write down the solutions on your own.

Plagiarism will not be tolerated.

**Late Submission Policy**:

Your homework grade will be decided by multiplying what you normally get from your answers by a "submission time factor (STF)".

If you submit on time (i.e. before the deadline), your STF is 1. So, you don't lose anything.

If you submit late, you will lose 0.01 of your STF for every 5 mins of delay.

We will not accept any homework later than 500 mins after the deadline.

SUCourse+'s timestamp will be used for STF computation.

If you submit multiple times, the last submission time will be used.

| Question | Points | Score |
|---|---|---|
| 1 | 20 | |
| 2 | 20 | |
| 3 | 50 | |

please proceed to the next page –→

| | | |
|---|---|---|
| 4 | 10 | |
| Total: | 100 | |



**Question 1** [20 points]

(a) [5 points] What is the form of the input array that triggers the worst case of the insertion sort?

> The input array which is sorted in decreased (reverse) order triggers the worst case of insertion sort. Below is an example array:
> [n, n-1, n-2, ..., 3,2,1]

(b) [5 points] What is the complexity of this worst–case behavior in Θ notation?

> Worst case complexity of insertion sort: $\Theta(n^2)$

(c) [10 points] Explain how this particular form of the array results in this complexity.

> Insertion sort executes maximum number of comparisons because input array is sorted in reverse order. 1st element is considered as a sorted subsequence. Algorithm compares $i^{th}$ ($1 < i <= n$) element with elements on the left side of $i^{th}$ element one by one by shifting each larger element to 1 position right.
> Total number of comparisons and shifts in the worst case:
> $1+2+...+(n-2)+(n-1) = n*(n-1)/2 = \Theta(n^2)$

**Question 2** [20 points]

(a) [5 points] What is the form of the input array that triggers the best case of theinsertion sort?

> When the input array is sorted in ascending order
> example array: [1, 2, ..., n-1, n]

(b) [5 points] What is the complexity of this best–case behavior in Θ notation?

> Best-case complexity of insertion sort: $\Theta(n)$

(c) [10 points] Explain how this particular form of the array results in this complexity.

Algorithm makes minimum number of comparisons for each element. Algorithm compares $i^{th}$ element with $i-1^{th}$ element ($1 < i <= n$). Only 1 comparison is made for each element i, since its place is correct. From i=2 to i=n, in total the algorithm makes n-1 comparisons. Which results in Θ(n).

**Question 3** [50 points]

Suppose that you are trying to prove $(5n + 4)^2 = O(n^2)$ by using the formal definition of $O$–notation, where $n \geq 0$.

In order to show that $(5n + 4)^2 = O(n^2)$ by using the formal definition of $O$–notation, we need to pick constants $c$ and $n_0$ such that for any $n \geq n_0$ we have

$$(5n + 4)^2 \leq cn^2 \tag{1}$$

(a) [25 points] If you use $n_0 = 2$, what is the smallest $c$ value that makes the proof go through?

$0 \leq (5n+4)^2 \leq cn^2$ for all $n \geq 2$.
$0 \leq 25n^2 + 40n + 16 \leq cn^2$
$0 \leq 25 + 40/n + 16/n^2 \leq c$
insert n=2: $0 \leq 25 + 20 + 4 \leq c$
$0 \leq 49 \leq c$
smallest c = 49

(b) [25 points] If you use $c = 36$, what is the smallest $n_0$ value that makes the proof go through?

$0 \leq (5n+4)^2 \leq cn^2$
for $c=36$: $0 \leq 25n^2 + 40n + 16 \leq 36n^2$
$25n^2 + 40n + 16 = 36n^2$
$11n^2 - 40n - 16 = 0$ → $n = 4$ (cannot be -4 because n is nonnegative)
smallest $n_0 = 4$

**Question 4** [10 points]

Rank the following functions in descending order with respect to their growth rates.

$\lg(n!)$ $n!$ $n2^n$ $2^{2^n}$ $\lg^2 n$ $(\lg n)!$

$2^{2^\wedge n}$ > n! > $n2^n$ > (lgn)! > lg(n!) > $\lg^2$n

– THE END –