BEN-GURION UNIVERSITY OF THE NEGEV

DATA STRUCTURES
202.1.1031

---

# Assignment No. 1

---

*Responsible staff members:*
Prof. Paz Carmi (carmip@bgu.ac.il)
Jules Zisser (zisserh@post.bgu.ac.il)      Ilan Naiman (naimani@post.bgu.ac.il)

*Authors:*
John Doe (123456789)   Jane Doe (987654321)

Publication date: March 30th, 2025
Submission date: April 20th, 2025

*Update date: March 17th, 2025*

אוניברסיטת בן-גוריון בנגב
Ben-Gurion University of the Negev

# 1 Submission Guidelines

- The submission is in pairs or individuals. We recommend working in pairs to encourage discussion and mutual inspiration. For the avoidance of doubt, if the assignment was submitted in pairs, all of the submitted answers should be the outcome of a joint work.

- Whether you submit as a pair or an individual, you must choose a group in the group assignment resource in Moodle. Groups created for individuals are of the form Assignment1_s_20, and groups for pairs are of the form Assignment1_p_100.

- Your submission should be a single pdf file (composed by LaTeX or any other word processor such as MS Word), and its name should be your group name (as chosen in Moodle), i.e., for the pair 17, the file name should be `Assignment1_p_17.pdf`.

- Questions regarding the assignment should be asked in the dedicated forum in Moodle or during the office hours of the responsible staff members. The forum is intended to allow students to discuss the assignment. The crew will answer questions in the forum when clarification is required in the pinned FAQ thread.

- $\log()$ in the course is in base 2, i.e., $\log_2()$ unless stated otherwise.

- A submission that does not include the integrity statement **will receive a score of 0**.

**Before starting your work, read the following instructions carefully:**

- With each version of your work, submit it so that with each submission a larger part of your work is complete. This way, you won't lose major parts of your work due to a technical issue or unexpected circumstances.

- DO NOT wait until the final hour for submitting your work, because there might be a power stoppage/computer issues/internet issues/Moodle issues/etc.

- save a continuous backup of your work in some cloud platform, so that you can access it from any device if your computer decides to malfunction.

- In case you work in pairs, both partners must verify that the work was submitted timely and fully. This way, you will prevent uncomfortable situations in cases of technical issues.

## 2  Integrity Statement

Each of the students submitting this assignment must sign the following integrity statement:

We, *<Insert your name>*, ID number *<Insert your ID>*, and *<Insert your name>*, ID number *<Insert your ID>* assert that the work we submitted is entirely our own. We have not received any part from any other student in the class, nor did we give parts of it for others to use. We realize that if my work is found to contain code that is not originally our own, a formal case will be opened against us with the BGU disciplinary committee.

## 3  Complexity Hierarchy

**Question 1:** (20 points) For each of the functions below place $f_i$ in the table in the correct place according to the following requirements:

- Functions $f_i$ and $f_j$ will be located on the same line if and if $f_i(n) \in \Theta(f_j(n))$

- For each function $f_i$ in line $k$ and $f_j$ in line $k+1$, $f_i(n) \in O(f_j(n))$

Your task is:

- For each of two functions $f_i, f_j$ you have placed in the same line in the table show constants $c_1, c_2$ and $n_0$ suitable for the definition of $f_i(n) \in \Theta(f_j(n))$.

- For each two functions $f_i$ you have placed in line $k$ in the table and $f_j$ you have placed in line $k+1$ show constants $c, n_0$ suitable for the definition of $f_i(n) \in O(f_j(n))$.

For example, for the following three functions:

1. $f_1(n) = n^2 - 10$

2. $f_2(n) = 6$

3. $f_3(n) = 22n^2 + 10n$

The table will look:

| line number | The functions |
|:-----------:|:-------------:|
| 1 | $f_2$ |
| 2 | $f_1, f_3$ |

- $f_2 \in O(f_1):\ 6 \le c(n^2 - 10)$ for $n \ge n_0 = 4$ and $c = 2$.

- $f_1 \in \Theta(f_3):$

    - $f_1 \in O(f_3):\ n^2 - 10 \le c(n^2 - 10)$ for $n \ge n_0 = 1$ and $c = 1$.
    - $f_1 \in \Omega(f_3):\ n^2 - 10 \ge c(n^2 - 10)$ for $n \ge n_0 = 6$ and $c = \frac{1}{32}$.

**Notice, since there are two functions in Line 2, it is enough to compare $f_2$ to only one of them.**

1. $f_1(n) = 1$

2. $f_2(n) = 1034$

3. $f_3(n) = n^2$

4. $f_4(n) = n^n$

5. $f_5(n) = 2^{n + \log n - 10}$

6. $f_6(n) = 2^{(2^n)}$

7. $f_7(n) = 2^n$

8. $f_8(n) = \log(4^n * n^8)$

9. $f_9(n) = \log\left(n^{\frac{1}{3}}\right)$

10. $f_{10}(n) = \frac{6n^2}{8} - 10$

**Answer 1:** Fill in the table below (**The Proof elaboration in this question is according to the example given above**):

| line number | The functions |
|:-----------:|:-------------:|
| 1 | |
| 2 | |
| 3 | |
| 4 | |
| 5 | |
| 6 | |
| 7 | |
| 8 | |
| 9 | |
| 10 | |

# 4 Properties of asymptotic bounds

**Question 2:** (25 points) Let $f(n)$ and $g(n)$ be asymptotically positive functions.

**Definition:** We say that a function $f : \mathbb{N} \to \mathbb{N}$ is **unbounded increasing monotone function** if:

1. for all $x, y$:

$$x < y \iff f(x) < f(y) \qquad (\iff \text{ stands for if and only if})$$

2. for all constant $M \in \mathbb{R}^+$ there is $n_0 \in \mathbb{N}$ such that for all $n > n_0$:

$$f(n) > M$$

> The logarithm function is an **unbounded increasing monotone function** defined for $x > 0$.

> In conjectures **6** and **7** you can assume that for all constant $c > 0$, there is $x_0 > 0$ such that for all $x > x_0$:
>
> $$c < \frac{x}{\log x}$$

**Prove each of the following conjectures:**

1. (4 points) $n > \log n$ for all $n \in \mathbb{N}$

2. (2 points) Reflexivity: $f(n) \in \Theta(f(n))$

3. (4 points) Transitivity: If $f(n) \in \Theta(g(n))$ and $g(n) \in \Theta(h(n))$ then $f(n) \in \Theta(h(n))$

4. (3 points) Transpose Symmetry: $f(n) \in O(g(n)) \iff g(n) \in \Omega(f(n))$

5. (4 points) For all $f(n) > 1$ and $g(n) > 1$ two **unbounded increasing monotone functions** such that $f(n) \in \Omega(g(n))$, imply
$$\log(f(n)) \in \Omega(\log(g(n)))$$

6. (4 points) For all constant $k \geq 1$ and for all constant $1 > \varepsilon > 0$:

$$\log^k(n) \in O(n^\varepsilon)$$

7. (4 points) For all constant $k \in \mathbb{N}$ and for all constant $\varepsilon > 0$ :

$$n^k \in O((1+\varepsilon)^n)$$

**Answer 2:** Replace with your answer.

# 5 Recurrence Examples

**Question 3:** (15 points) Find and prove an asymptotic tight bound $\Theta$ for $T(n)$ in each of the following recurrences. Assume that $T(c) = c$ for $c \leq 4$.

1. $T(n) = 4T(\frac{n}{2}) + n$

2. $T(n) = T(\frac{3}{7}n) + 1$

3. $T(n) = 6T(\frac{n}{2}) + n^4 \log n$

4. $T(n) = 8T(\frac{n}{2}) + n^4 + 3n^3 \log n$

5. $T(n) = T(\sqrt{n}) + n$

**Answer 3:** Replace with your answer.

# 6 Time Complexity

**Question 4:** (10 points)

Find the time complexity of the following algorithms in terms of $\Theta$ (you are not required to prove this asymptotic bound). You are expected to analyze the asymptotic tight bound of each line in the algorithms, as presented in class for the analysis of insertion sort. In addition, show the calculation of $T(n)$ as done in the lecture.

---
**Algorithm 1** Search $(array[int], key)$
---
1: **return** Foo Search$(array, key, 0, array.length - 1)$

---

---
**Algorithm 2** Foo Search (array[int], key, left, right)
---
1: $output \leftarrow -1$     // *default value for not found*
2: $i \leftarrow left - 1$
3: **if** left $\leq$ right **then**
4:     **for** $j = left$ **to** $right - 1$  **do**
5:         **if** $array[j] \leq array[right]$ **then**
6:             $i \leftarrow i + 1$
7:             exchange $array[i]$ with $array[j]$
8:         **end if**
9:     **end for**
10:     exchange $array[i + 1]$ with $array[right]$
11:     **if** $array[i + 1] = key$ **then**
12:         $output \leftarrow i + 1$
13:     **else if** $array[i + 1] \leq key$ **then**
14:         $output \leftarrow$ Foo Search$(array, key, left, i)$
15:     **else**
16:         $output \leftarrow$ Foo Search$(array, key, i + 2, right)$
17:     **end if**
18: **end if**
19: **return**  output

---

**Answer 4:** Fill in the table and calculate the sum $T(n)$ below (do not fill in the grayed-out entries):

| Line number | Number of times the line is performed | Cost of a single repetition | Total cost of this line |
|:-:|:-:|:-:|:-:|
| 1 | $\Theta(1)$ | $\Theta(1)$ | $\Theta(1)$ |
| 2 | | | |
| 3 | | | |
| 4 | | | |
| 5 | | | |
| 6 | | | |
| 7 | | | |
| 8 | | | |
| 9 | | | |
| 10 | | | |
| 11 | | | |
| 12 | | | |
| 13 | | | |
| 14 | | | |
| 15 | | | |
| 16 | | | |
| 17 | | | |
| 18 | | | |
| 19 | | | |
| Total | | | $\Theta(?)$ |

$$T(n) =?$$

# 7 Algorithm Development

**Question 5:** (10 points) Describe in pseudo-code an efficient algorithm for solving the following problem, and analyze its running time and additional memory usage. The running time should be as asymptotically low as possible.

**Notice:** In this question you can assume that there is a **fast initializing array** that allows initialization of $n$ cells in a constant time, that is, in $\Theta(1)$. This stands in contradiction to the RAM model displayed in class (in which allocation of $n$ cells is done in $\Theta(1)$ and initialization of all cells is $\Theta(n)$), and can be assumed in questions 5 and 6 only.

**Input:** Two unsorted arrays $A$ and $B$ of size $N \in \mathbb{N}$. The value of each element in the arrays is a natural number in the range $[0, 2N]$.

**Output:** The number of elements in Array $B$ that appear in Array $A$.

**Examples:**

Input:

| A | 5 | 2 | 14 | 2 | 2 | 7 | 1 |
|---|---|---|----|---|----|---|---|
| B | 6 | 9 | 9 | 10 | 13 | 0 | 8 |

Output: 0

Input:

| A | 5 | 9 | 7 | 2 | 2 | 7 | 7 |
|---|---|---|---|----|---|----|---|
| B | 6 | 9 | 9 | 10 | 7 | 13 | 8 |

Output: 3      (*9 appears in A and appears twice in B, 7 appears in A and appears once in B*)

Input:

| A | 6 | 9 | 9 | 10 | 7 | 13 | 8 |
|---|---|---|---|----|---|----|---|
| B | 5 | 9 | 7 | 2 | 2 | 7 | 7 |

Output: 4      (*9 appears in A and appears once in B, 7 appears in A and appears three times in B*)

**Answer 5:** You should include the algorithm description, run-time and additional memory analysis:

**Question 6:** (5 points) Solve question 5, where the value of each element in the arrays is a natural number in the range $[0, N^2]$ (instead of $[0, 2N]$).

**Answer 6:** You should include the algorithm description, run-time, and additional memory analysis:

# 8 AVL tree

**Question 7:** (5 points) The following sequence of keys has been added to an AVL tree. $< 5,\ 6,\ 12,\ 10, 8 >$.

  **Answer 7:** Draw the AVL tree after each addition.

**Question 8:** (10 points) Consider the following set $S$ of keys: $\{1, 2, 3, 4, 5, 6, 7\}$

  **Answer 8:** Write an insertion sequence of the keys in $S$ that has **at least two rotations overall** and results in an AVL tree of

(i) maximum height.

(ii) minimum height.