

הוראות המבחן:

- משך המבחן 4 שעות.
- בבחן 4 שאלות, עלייר לענות על כלן. פירוט הניקוד של כל שאלה מופיע בcoturneta.
- השאלות מופיעות תחילת באנגלית ולאחר מכן בעברית.
- ניתן לענות בעברית.
- חומר עדיף מותר: הדפסות חומר העזר המותר לבחן כפי שהוא מופיע באתר הקורס בפלטפורמה.

בהצלחה ☺

שאלה 1 (25 נקודות: 15 נק' לסעיף א, 10 נק' לסעיף ב)

When given a queue Q containing n natural numbers, you are required to transfer the n numbers into stack S . The numbers must appear in the stack as an ordered series, with the smallest number at the bottom of the stack, and the largest number at the head of the stack.

Assume that operations ***Enqueue***, ***Dequeue*** and ***Empty*** are defined for queue Q . And that operations ***Push***, ***Pop***, ***Empty*** and ***Top*** (that returns, but does not delete, the value at the top of the stack) are defined for stack S .

נתון תור Q המכיל n מספרים טبויים. ברצוננו להעביר את n המספרים למחסנית S , כך שיהיו סדרה ממוינת מהבסיס (המספר הקטן ביותר) אל הראש (המספר הגדול ביותר).

הנichi שעל התור מוגדרות, הפעולות ***Empty***, ***Dequeue***, ***Enqueue***, ו-***Empty***. על המחסנית, מוגדרות הפעולות ***Top*** (המחזירה אך לא מוחקת את הערך הנמצא בראש המחסנית).

שאלה 1 - סעיף א' (15 נק')

- a. Present an algorithm (in pseudocode) that implements the transfer (from Q to S , in a run-time complexity of $O(n^2)$). The algorithm may use only $O(1)$ additional memory.

א. כתבי אלגוריתם (פסאודוקוד) לביצוע העברת המספרים (מ- Q ל- S), כנדרש לעיל בזמן (n^2).
האלגוריתם ישמש רק ב- $O(1)$ זיכרון נוספים.

שאלה 1 - סעיף ב' (10 נק')

- b. Explain why the algorithm you suggested works correctly and prove that it meets the runtime requirements defined in this question.

ב. הסבירי מדוע האלגוריתם שכתבת פועל נכון, והrai שזמן הריצה עומד בדרישות השאלה.

שאלה 2 (25 נקודות: 13 נק' סעיף א' ; 12 סעיף ב')

שאלה 2 מורכבת משני תת-סעיפים.

יש לשים לב שבסעיף א' אתן מתבקשות רק לתאר את המימוש של מבנה הנתונים,
ובסעיף ב' אתן מתבקשות לפרט את אופן המימוש של הפעולות השונות.

שאלה 2 – סעיף א' (13 נק')

- a. Suggest a data structure S that can be used to implement all the following operations in the requested runtime complexity, where n is the number of keys that differ from each other in S (the number of records in S can be much greater than n):

The operation	Runtime complexity
Insert(k, R, S): insert record R , with a key-value k , into S .	$O(\lg n)$
Delete(k, S): delete a record (maybe one of many), with a key-value k , from S .	$O(\lg n)$
Find(k, S): find a record (maybe one of many, with a key-value of k , in S .	$O(\lg n)$
MostFrequent(S): return the key value with the highest frequency in S .	$O(1)$

Note: the data structure, S , can consist of several basic data structures.

א. הצעי מבנה נתונים S שבאמצעותו ניתן למשתכל אחות מהפעולות הבאות בסיבוכיות המבוקשת, כאשר n הוא מספר המפתחות השונים זה מזה ב- S (מספר הרשומות יכול להיות הרבה יותר מ- n):

הזמן הנדרש	הפעולה
$O(\lg n)$	הכנסת הרשומה R בעלת המפתח k ל- S : Insert (k, R, S)
$O(\lg n)$	מחיקת רשותה כלשהי בעלת מפתח k מ- S : Delete (k, S)
$O(\lg n)$	חיפוש רשותה כלשהי בעלת מפתח k ב- S : Find (k, S)
$O(1)$	החזרת ערך המפתח בעל השכיחות הגבוהה ביותר ב- S : MostFrequent (S)

הערה: מבנה נתונים יכול להיות מורכב מכמה מבני נתונים בסיסיים.

שאלה 2 – סעיף ב' (12 נק')

b. Explain briefly how each of the 4 operations will be performed.

ב. הסבירי בקצרה כיצד מתבצע כל אחת מארבעת הפעולות בסיבוכיות הנדרשת.

שאלה 3 (20 נקודות: 12 נק' סעיף א' ; 8 נק' סעיף ב')

שאלה 3 – סעיף א' (12 נק')

You are given a hash table where collision resolution is through chaining, and the chaining is done using a doubly-linked list. In addition to the usual basic hash-table operations: search, insert and delete, we would like to add the functionality of **return the minimum value** in the hash table.

Assume that:

- The hash tables' size is m .
- The number of elements in the hash table is n .
- When performing the delete operation, you are given a pointer to the item to be deleted.

Answer the following question:

Is there a possible implementation that enables us to perform the three following operations: return minimum value, insert and delete in $O(1)$ (worst case) time complexity, and search in $O(1)$ (average case) time complexity?

If so, explain how to do so (in the required time complexities).

If not, what is the maximum number of operations (from the 4 discussed operations) that can be performed in $O(1)$ (worst case) time complexity? Explain what will be the time complexity of each of the four operations, and how you would implement them (in the time complexity you stated).

נתונה טבלת גיבוב בשיטת השרשור, כאשר כל רשימה בה היא רשימה מקוושת דו-כיוונית. בנוסח לפעולות הרגילות על טבלת גיבוב: חיפוש, הכנסה ומחיקה, בראצוננו לאפשר ביצוע פעולה **החזירת מינימום** (החזירת ערך המפתח המינימלי בטבלה).
הנחות:

- גודל הטבלה הוא m .
- מספר האיברים בטבלה הוא n .
- לצורך ביצוע פעולה מחיקה, מקבלים מצביע לאיבר שרצו למחוק.

ענין על השאלה הבאה:

אם יש שימוש המאפשר לבצע את הפעולות: החזרת המינימום, הכנסה, מחיקה – כל אחת מהן ב-(1) O (במקרה הגרוע), ופועלות החיפוש ב-(1) O בumnoץ? אם כן, הסבירי כיצד תבוצע כל פעולה (בסיבוכיות שצויינה לעיל). אם לא, מהו המספר המקסימלי של פעולות (מבין ארבע הפעולות הניל') הניתנות לביצוע ב-(1) O במקרה הגרוע? צייני באיזו סיבוכיות תבוצע כל פעולה, וכי怎 היא תוממש (בסיבוכיות שצויינה).

שאלה 3 – סעיף ב' (8 נק')

It is known that when deleting a value X, from a BST, if X is stored in a node with both a non-empty left child and a non-empty right child, we must find X's in-order successor (or predecessor).

Fill in the missing sections in the following sentence, so that the following statement is correct:

Which of the following statements may occur when discussing X's in-order successor:

1. X's in-order successor is a leaf node.
2. X's in-order successor has 2 non-empty children (a right child and a left child).
3. X's in-order successor has no left child.
4. X's in-order successor has no right child.
5. X's in-order successor has a non-empty left child.
6. X's in-order successor has a non-empty right child.

ידוע שכשר רוצחים למחוק מעץ חיפוש בינרי צומת X שיש לו חן בן שמאלית וחן בן ימנית (ששתיהן אינם ריקים), אנו נדרשים למצוא את העוקב (או הקודם) שלו בסדר תוכי.

עבור הצומת העוקב בסדר תוכי של צומת X (כמפורט לעיל), אילו מבין המשפטים הבאים יכולים להתקיים:

1. הצומת העוקב בסדר תוכי הוא עלה.
2. לצומת העוקב בסדר תוכי יש שני בניים (לא ריקים).
3. לצומת העוקב בסדר תוכי אין בן שמאלית.
4. לצומת העוקב בסדר תוכי אין בן ימני.
5. לצומת העוקב בסדר תוכי יש בן שמאלית.
6. לצומת העוקב בסדר תוכי יש בן ימני.

שאלה 4 (30 נקודות: 18 נק' סעיף א' ; 12 סעיף ב')

You are given a Graph $G=(V, E)$ which is an undirected and unweighted graph, and two nodes $s, t \in V$ and an arc $e \in E$ in the given graph.

נתון גרף $G=(V, E)$ לא מכוון ולא ממושקל, ונתונים שני צמתים $s, t \in V$ וקשת $e \in E$ בגרף.

שאלה 4 – סעיף א' (18 נק')

- Describe an efficient as possible algorithm, that checks whether the arc $e=(u, v)$ (where $u, v \in V$) is on all the shortest paths between s and t in G .

Prove the correctness of your algorithm and analyze its run-time complexity

a. תאריך אלגוריתם יעיל ככל האפשר הבודק האם הקשת (v, u) ($v \in V$, $u \in V$) נמצאת על כל המסלולים הקצרים ביותר בין s ל- t ב- G .

הוכיח את נכונות האלגוריתם ונתחיה את סיבוכיות זמן הריצה של האלגוריתם.

שאלה 4 – סעיף ב' (12 נק')

- Describe an efficient as possible algorithm, that checks whether the arc $e=(u, v)$ (where $u, v \in V$) is on any of the shortest paths between s and t in G .

Analyze the algorithm's run-time complexity.

b. תאריך אלגוריתם יעיל ככל האפשר הבודק האם הקשת (v, u) ($v \in V$, $u \in V$) נמצאת על מסלול קצר ביותר בלשחו בין s ל- t ב- G .

נתחיה את סיבוכיות זמן הריצה של האלגוריתם.