

מבני נתונים – פרויקט מספר 1 – עץ מאוזן

הקדמה

בתרגיל זה שני חלקים:

- חלק מעשי: מימוש של עץ AVL. עמודים 1-2 במסמך מתארים חלק זה.
- חלק ניסויי-תאורטי: בהתבסס על המימוש מהחלק המעשי, נבצע מספר ניסויים עם ניתוח תאורטי נלווה. עמודים 3-4 מתארים חלק זה.

שימו לב: בסוף המסמך (עמוד 4) ישנן הוראות הגשה – הקפידו לפעול לפיהן. **תאריך הגשה: 28/12/2025.**

חלק מעשי

דרישות

בתרגיל זה יש לממש עץ AVL, לפי ההגדרות שניתנו בכיתה. לכל איבר בעץ יש ערך (value) שהוא מחרוזת, ומפתח (key) שהוא מספר שלם. כל המפתחות שונים זה מזה, והסדר על צמתי העץ מתייחס כרגיל אך ורק למפתחות. המימוש יהיה בשפת **python 3.13** וצריך להיות מבוסס על קובץ השלד המופיע באתר הקורס. הפעולות שיש לממש הן:

פועלה	תיאור
search(k)	הפונקציה מחפשת איבר בעל מפתח k. היא מחזירה זוג סדור (x, e) , כאשר x הוא מצביע לצומת המתאים (או None אם לא קיים), ו-e הוא אורכו בקשתות של מסלול החיפוש+1.
finger_search(k)	הפונקציה מחפשת איבר בעל מפתח k, החל מהצומת המקסימלי. היא מחזירה זוג סדור (x, e) , כאשר x הוא מצביע לצומת המתאים (או None אם לא קיים), ו-e הוא אורכו בקשתות של מסלול החיפוש+1.
insert(k, v)	הכנסת איבר בעל ערך v ומפתח k לעץ, ניתן להניח שהמפתח לא קיים כבר בעץ. הפונקציה מחזירה שלשה (x, e, h) , כאשר x מצביע לצומת שנוצר, e מספר הקשתות על מסלול ההכנסה, ו-h מספר מקרי שינוי גובה (promote) שנדרשו במהלך האיזון.
finger_insert(k, v)	הכנסת איבר בעל ערך v ומפתח k לעץ החל מהצומת המקסימלי, ניתן להניח שהמפתח לא קיים כבר בעץ. הפונקציה מחזירה שלשה (x, e, h) , כאשר x מצביע לצומת שנוצר, e מספר הקשתות על מסלול ההכנסה, ו-h מספר מקרי שינוי גובה (promote) שנדרשו במהלך האיזון.
delete(x)	מחיקת הצומת x מהעץ בהינתן מצביע.
join(t, k, v)	הפונקציה מקבלת עץ נוסף t שכל המפתחות שלו קטנים ממש, או שכולם גדולים ממש מהמפתחות של העץ הנוכחי, כאשר המפתח k נמצא ביניהם. על הפונקציה לאחד לעץ הנוכחי את העץ הנוסף והאיבר החדש (k, v) . לאחר הפעולה העץ t אינו שמיש, כלומר אסור למשתמש (טסטר) לקרוא לו יותר.
split(x)	הפונקציה מקבלת מצביע לצומת x בעץ. עליה לפצל את העץ לשניים ולהחזיר (t_1, t_2) , כך ש- t_1 יכיל את המפתחות הקטנים מ-x ו- t_2 את הגדולים. לאחר הפעולה העץ הנוכחי וגם x אינם שמישים.
avl_to_array()	הפונקציה מחזירה מערך ממוריד (רשימה של פיתון ממויינת ע"פ המפתחות) של האיברים במילון כאשר כל איבר מיוצג ע"י זוג סדור של (key, value).
max_node()	הפונקציה מחזירה מצביע לצומת בעל המפתח המקסימלי בעץ.
size()	הפונקציה מחזירה את מספר האיברים בעץ. אין לתחזק שדה size בכל צומת אלא רק של העץ כולו. לאחר ביצוע split מותר שהערך לא יהיה תקין, כלומר לאחר split לא תיבדק size על אף אחד משני העצים שהוחזרו ב-split.
get_root()	הפונקציה מחזירה מצביע לשורש העץ.

לצורך מימוש פעולות אלו, ניעזר במחלקה **AVLNode** המופיעה בקובץ. נדרוש שלכל עלה יהיו 2 בנים "וירטואליים", כלומר צמתים ללא מפתח. באופן זה, נוו יותר לממש גלגולים מכיוון שלכל צומת יהיו 2 בנים.

המחלקה AVLNode מכילה את השדות הבאים, שעליכם לתחזק:

key – המפתח של הצומת.

value – הערך של הצומת.

left – הבן השמאלי של הצומת.

right – הבן הימני של הצומת.

parent – ההורה של הצומת.

height – גובה הצומת.

בנוסף, המחלקה תומכת בפעולה הבאה:

is_real_node – מחזירה TRUE אם הצומת מייצג צומת אמיתי בעץ (קרי: צומת שאינו וירטואלי).

עבור צומת וירטואלי יש להחזיר FALSE.

מספר הבהרות והסברים:

1. בפעולות החיפוש אנו סופרים את אורכו בקשתות של המסלול בעץ בין הצומת שבו התחיל החיפוש והצומת שבו הסתיים החיפוש, ומוסיפים לתוצאה 1.
2. בפעולות ההכנסה סופרים את אורכו בקשתות של המסלול בעץ בין הצומת שבו התחיל החיפוש והצומת שהכנסנו מייד לאחר ההכנסה, לפני פעולות איזון. באופן שקול, זהה לאורך המסלול להורה שלו + 1, ולכן אם מחפשים מפתח שלא קיים בעץ לפני שמכניסים אותו מקבלים תוצאה זהה. לדוגמה, אם משתמשים בפעולה finger_search כדי להכניס צומת מקסימלי חדש, מקבלים $e = 1$.
3. עבור פעולות ההכנסה, כאשר מפעילים את אלגוריתם האיזון כפי שנלמד בכיתה, סופרים אך ורק כמה פעמים היינו במקרה 1 (promote) של שינוי שדה גובה. אנו מתעלמים מגלגולים ומשינויי גובה שהתרחשו במקרים 2 או 3.
- לדוגמה, נתבונן בסדרת הכנסה של 3, ואז 1, ואז 2. בהכנסה הראשונה אין פעולות איזון. בהכנסה של 1 מעדכנים את הגובה של 3 (מקרה 1), פעולה אחת. בהכנסה של 2 מעדכנים את הגובה של 1 (מקרה 1) ולאחר מכן בשורש 3 אנחנו נמצאים במקרה 3 של גלגול כפול אותו לא סופרים, לכן שוב מחזירים 1.
4. חיפוש החל מהמקסימום: עולים מהמקסימום עד הצומת הראשון בעל מפתח קטן יותר (או השורש) וממנו יורדים כרגיל. (חיפוש זה יעיל יותר רק עבור מפתחות גדולים שקרובים למקסימום).

הערות חשובות:

1. המימוש יבוצע על ידי מילוי קובץ השלד. מותר להחליף את תוכן הפונקציות הקיימות ולהוסיף פונקציות ושדות חדשים. אסור לשנות את חתימות הפונקציות הקיימות ואת שמות השדות הקיימים כדי לא לפגוע בטסטר (כן מותר להוסיף פרמטרים עם ערך ברירת מחדל). על כל הפונקציות/מחלקות להופיע בקובץ יחיד.
2. אין להשתמש באף מימוש ספרייה של מבנה נתונים.
3. עליכם לממש את כל הפעולות בסיבוכיות המיטבית.

סיבוכיות

יש לציין בקוד ולהסביר בקצרה במסמך התיעוד את סיבוכיות זמן הריצה במקרה הגרוע (האסימפטוטית, במונחי O הדוקים) של כל פונקציה שמכילה לולאות/רקורסיה, כתלות במספר האיברים בעץ n.

פלט

אין צורך בפלט למשתמש.

תיעוד

בנוסף לבדיקות אוטומטיות של הקוד שיוגש, קובץ המקור ייבדק גם באופן ידני. חשוב להקפיד על תיעוד לכל פונקציה, וכמות סבירה של הערות. הקוד צריך להיות קריא, בפרט הקפידו על בחירת שמות משתנים ועל אורך השורות.

יש להגיש בנוסף לקוד גם מסמך תיעוד חיצוני. המסמך יכלול את תיאור המחלקה שמומשה, ואת תפקידו של כל חבר במחלקה. עבור כל פונקציה במחלקה יש לפרט מה היא עושה, כיצד היא פועלת ומה סיבוכיות זמן הריצה שלה. בפרט, אם פונקציה קוראת לפונקציית עזר, יש להתייחס גם לפונקציית העזר בניתוח. עבור פונקציות שעולות זמן קבוע יספיק תיאור קצר ולא לפרט את ניתוח הסיבוכיות.

בדיקות

התרגילים ייבדקו באמצעות תוכנת טסטר שקוראת לפונקציות המפורטות מעלה בתרחישים שונים, ומוודאת את נכונות התוצאות. קובץ הטסטר שלנו לא יפורסם לפני הבדיקות. מומלץ מאוד לממש אוסף בדיקות עבור המימוש, לא בשביל ההגשה, אלא כדי לבדוק שהקוד לא רק רץ, אלא גם נכון!

בקובץ שתגישו לא תהיה פונקציית **main** ולא יהיו הרצות קוד/הדפסות, דבר זה יפגע בטסטר שיבדוק לכם את התרגילים. אין צורך להגיש את הקוד הנוסף שכתבתם לחלק הניסויי.

חלק ניסויי/תאורטי

בשאלה זו נדון ב insertion-sort באמצעות AVL Finger Tree. המיון מתבצע באופן הבא: מכניסים את האיברים לפי הסדר (הלא ממיון) אל העץ, כאשר החיפוש בהכנסת כל איבר חדש מתחיל מהמקסימום הנוכחי, ובסיום מבצעים סריקת in-order לקבלת הסדר הממיון. עבור עלות בניית העץ, ננתח בנפרד את עלות החיפוש ואת מספר פעולות האיזון.

- לצורך הניתוח, נמיין מערכים בגדלים שונים. גודל המערך שנמיין יהיה $n = 300 \cdot 2^i$ כאשר $i = 1, \dots, 10$, ואיבריו יהיו הטבעיים עד n . למשל, עבור $i = 1$ המערך בגודל 600, ועבור $i = 5$ המערך בגודל 9600.

- לכל גודל n של מערך, נבצע 4 ניסויים נפרדים:
 - בניסוי הראשון נמיין מערך ממיון, מקטן לגדול.
 - בניסוי השני נמיין מערך ממיון הפוך, מגדול לקטן.
 - בניסוי השלישי סדר האיברים במערך יהיה אקראי.
 - בניסוי הרביעי ניקח מערך ממיון ועבור כל אינדקס פרט לאחרון $i = 0, \dots, n - 2$, נבצע החלפה עם האיבר הבא $A[i] \leftrightarrow A[i + 1]$ בסיכוי חצי (שימו לב שייתכן שאיבר יוחלף מספר פעמים).

הערה: בסעיפים הבאים, עבור ניסויים אקראיים, יש לקחת את הממוצע על פני 20 ניסויים.

- יש למלא בטבלה הבאה את סך עלויות האיזון **ללא גלגולים** עבור כל אחד מהניסויים. הסבירו מהו החסם העליון התאורטי על סך עלויות האיזון כולל גלגולים, והאם הערכים בטבלה מתאימים. לסיום, נמקו מדוע תוספת הגלגולים לספירה אינה משנה אסימפטוטית.

מספר סידורי i	עלות איזון במערך ממיון	עלות איזון במערך ממיון-הפוך	עלות איזון במערך מסודר אקראית	עלות איזון במערך עם היפוכים סמוכים אקראיים
1				
2				
...				

- בהינתן מערך A בגודל n נגדיר היפוך בתור זוג אינדקסים $i < j$ כך שמתקיים $A[i] > A[j]$, ונסמן את מספר ההיפוכים הכולל במערך ב- I . ניתן לשים לב כי באופן כללי $0 \leq I \leq \binom{n}{2}$ וככל שיש פחות היפוכים כך המערך קרוב יותר לממיון. יש למלא בטבלה הבאה את מספר ההיפוכים במערך הקלט עבור כל אחד מהניסויים (מספיק עד $i = 5$).

מספר סידורי i	מספר היפוכים במערך ממיון	מספר היפוכים במערך ממיון-הפוך	מספר היפוכים במערך מסודר אקראית	מספר היפוכים במערך עם היפוכים סמוכים אקראיים
1				
2				
...				

3. יש למלא בטבלה הבאה את סך עלויות החיפוש עבור כל אחד מהניסויים.

מספר סידורי i	עלות חיפוש במערך ממין-הפוך	עלות חיפוש במערך מסודר אקראית	עלות חיפוש במערך עם היפוכים סמוכים אקראיים
1			
2			
...			

4. כדי לחסום מלמעלה את סך עלויות החיפוש באופן תאורטי, נסמן ב- d_i את מספר האיברים לפני האיבר באינדקס i שגדולים ממנו.

I. הסבירו מדוע $I = \sum d_i$.

II. חסמו את עלות החיפוש בעת הכנסת האיבר ה- i כפונקציה של d_i , והסיקו כי סך עלויות החיפוש לאורך סדרת ההכנסות הוא $O(\log \prod_{i=1}^n (d_i + 2))$.

הערה: כאשר הסיבוכיות היא $O(\max(1, \log d))$ עבור $d \geq 0$, ניתן להמיר לביטוי

$O(\log(d + 2))$ כדי לעבור לביטוי אחד פשוט וחוקי שחוסם את שניהם מלמעלה.

III. השתמשו באי-שוויון הממוצעים כדי לחסום מלמעלה את סך עלויות החיפוש כפונקציה של

n, I . הסבירו מדוע זוהי העלות הדומיננטית מבחינה אסימפטוטית.

IV. השוו בין החסם שהתקבל ותוצאות הניסויים.

הוראות הגשה

הגשת התרגיל תתבצע באופן אלקטרוני באתר הקורס במודל.

הגשת התרגיל היא בזוגות בלבד!

כל זוג יבחר נציג/ה ויעלה רק תחת שם המשתמש של הנציג/ה את קבצי התרגיל (תחת קובץ **zip**) למודל.

על ההגשה לכלול שלושה קבצים:

1. קובץ המקור (הרחבה של קובץ השלד שניתן) תחת השם **AVLTree.py**.
2. קובץ טקסט **info.txt** המכיל את פרטי הזוג: שמות מלאים בעברית, שמות מלאים באנגלית ומספרי ת"ז. יש לרשום גם שמות משתמש (או לחילופין אימיילים אוניברסיטאיים) של המגישים.
3. מסמך תיעוד חיצוני, המכיל גם את תוצאות המדידות. את המסמך יש להגיש בפורמט **pdf**.

שמות קובץ התיעוד וקובץ ה-**zip** צריכים לכלול את שמות המשתמש האוניברסיטאיים של הזוג המגיש לפי הפורמט **AVLTree_username1_username2.pdf/zip**, בתוכן הקבצים יש לציין את שמות המשתמש, תעודות הזהות ושמות המגישים (בכותרת המסמך ובשורת הערה בקובץ המקור).

הגשת שיעורי הבית באיחור - באישור מראש בלבד. הגשה באיחור ללא אישור תגרור הורדת נקודות מהציון. הגשת התרגיל היא חובה לשם קבלת ציון בקורס.

בהצלחה!