אוניברסיטת תל אביב

סמסטר א' תשפ"ג

מבני נתונים – פרויקט מספר 2 – ערימת פיבונאצ'י

הקדמה:

בתרגיל זה שני חלקים:

א. חלק המעשי: מימוש של ערימת פיבונאצ'י. עמודים 1-3 במסמך זה מתארים את החלק הזה.

ב. חלק ניסויי-תיאורטי: בהתבסס על המימוש מהחלק המעשי, נבצע מספר "ניסויים" עם ניתוח תיאורטי נלווה. עמודים 3-5 מתארים את החלק הזה.

שימו לב: בסוף המסמך (עמוד 5) ישנן הוראות הגשה – הקפידו לפעול לפיהן.

תאריך הגשה: 22.1.23

חלק מעשי

דרישות

יש לממש ערימת פיבונאצ'י (Fibonacci Heap), לפי ההגדרות שניתנו בכיתה. ניתן למצוא את הפרטים במצגות הקורס ובספר של Cormen. **שימו לב לתת-פרק "הבהרות מימוש" להלן, ועיקבו אחרי השרשור הרלוונטי בפורום התרגיל באתר הקורס.**

בתרגיל זה נניח שהאיברים בערימה הם תמיד מספרים שלמים שונים זה מזה. המימוש צריך להיות מבוסס על קובץ השלד המופיע באתר הקורס. הפעולות שמופיעות בקובץ:

פעולה	תיאור
isEmpty()	הפונקציה מחזירה ערך TRUE אם ורק אם הערימה ריקה.
insert(int i)	הפונקציה יוצרת צומת מסוג HeapNode שמכיל את המפתח i ומכניסה אותו לערימה. פעולה זו מחזירה את הצומת שנוצר (שמכיל את המפתח i).
deleteMin()	מחיקת הצומת שהמפתח שלו מינימלי מבין המפתחות שבערימה (אין צורך להחזיר אותו).
findMin()	החזרת הצומת (מטיפוס HeapNode) שהמפתח שלו מינימלי מבין המפתחות שבערימה. מערימה ריקה יוחזר null.
meld(FibonacciHeap heap2)	מיזוג ערימה נוספת heap2 עם הערימה הנוכחית.
size()	הפונקציה מחזירה את מספר האיברים בערימה.
countersRep()	הפונקציה מחזירה מערך מונים כך שבאינדקס i מופיע מספר העצים בערימה שהסדר (rank) שלהם הוא i. כלומר, היא מחזירה מערך של integers, כך שלכל אינדקס i בין 0 עד הדרגה המקסימלית של עץ שקיימת בערימה, הערך שמוחזר במערך הוא מספר העצים שקיימים בערימה מסדר i. עבור ערימה ריקה יוחזר מערך ריק. **חשוב**: זו הפונקציה היחידה שאין צורך לממש ביעילות, ממשו איך שנוח.
delete(HeapNode x)	מחיקת הצומת x מהערימה.
decreaseKey(HeapNode x, int d)	ערכו של המפתח של הצומת x יופחת בערך $d \geq 0$. כלומר, מתבצע $x.key \leftarrow x.key - d$, וכמובן שיש לעדכן את מבנה הערימה בהתאם לשינוי זה (למשל: Cascading Cuts במידת הצורך).
nonMarked()	הפונקציה מחזירה את כמות האיברים שאינם marked בערימה.
potential()	הפונקציה מחזירה את ערך הפוטנציאל הנוכחי של הערימה. הפונטנציאל, כפי שהוגדר בשיעור, הינו: Potential = #trees + 2*#marked
totalLinks()	פונקציה סטטית זו מחזירה את מספר כל פעולות החיבור שבוצעו מתחילת ריצת התוכנית. פעולת חיבור היא הפעולה שמקבלת שני עצים מאותו סדר ומחברת אותם.
totalCuts()	פונקציה סטטית זו מחזירה את מספר כל פעולות החיתוך שבוצעו מתחילת ריצת התוכנית. פעולת חיתוך מתרחשת עקב decreaseKey, כאשר מנתקים תת-עץ מאביו (כולל cascading cuts).
kMin(FibonacciHeap H, int k)	פונקציה סטטית זו מקבלת ערימה H שהיא עץ (יער של עץ יחיד) שדרגתו $\deg(H)$, ומספר חיובי $k < size(H)$. הפונקציה מחזירה מערך

	ממוין של k הצמתים הקטנים ב-H. על הפעולה לרוץ בסיבוכיות $O(k \cdot \deg(H))$. אין לבצע שום שינוי בערימת הקלט (למשל לבצע deleteMin).

<u>הערות חשובות:</u>

1. בקובץ השלד מופיעים ה header ים של כל הפונקציות. **המימוש יבוצע על ידי מילוי קובץ השלד. במידת הצורך, ניתן להרחיב את המימוש** (למשל להוסיף פונקציות עזר שאינן מופיעות בשלד), אך **אסור לשנות את הגדרות הפונקציות לעיל**. על כל הפונקציות/מחלקות להופיע בקובץ יחיד.
2. **אין להשתמש באף מימוש ספרייה של מבנה נתונים.**

הבהרות מימוש

במימוש של ערימת פיבונאצ'י שנלמד בהרצאה ישנן דרגות חופש שעלולות לגרום למבנה העצים להיות שונה בין מימושים שונים. בשביל אחידות בהתנהגות עבור הניסויים והבדיקות, נוסיף אילוצים נוספים:

(1) לעצים בערימה יש סדר: מ"שמאל" (חדש) ל"ימין" (ותיק).
(2) שורשי עצים לעולם אינם מסומנים.

להלן תיאור מפורט יותר של אופן העדכון בהתאם לכל פעולה:

- **הערה כללית**: אם סדר העצים היה חופשי, ניתן לדאוג שהעץ הראשון (השמאלי) יהיה זה ששורשו הוא איבר המינימום, וכך לשמור מצביע רק אליו. מכיוון שהעצים מסודרים בסדר כרונולוגי, דרוש פתרון מעט שונה, כמו זה המופיע בציורים שבמצגת (כך ש- min ו- first אינם בהכרח שקולים).
- **insert**: האיבר החדש ייכנס לראש הערימה (משמאל לרשימת העצים).
- **cut**: הצומת שנחתך מאביו ייכנס לראש הערימה (שמאל). בנוסף, אם לצומת y הייתה רשימת בנים $x_1, \dots, x_k$ וביצענו $cut(x_i, y)$ אז לאחר החיתוך סדר הבנים של y אינו משתנה פרט לבן שנחתך. כלומר, רשימת הבנים תהיה $x_1, \dots, x_{i-1}, x_{i+1}, x_k$ (בשונה מהפסאודו-קוד בהרצאה שאחריו הסדר יוצא $x_{i+1}, \dots, x_k, x_1, \dots, x_{i-1}$).
- **meld (עצל)**: משרשרים את היער של הערימה השנייה אחרי רשימת העצים של הערימה המקורית (מימינה). (מכאן והלאה, לא מתבצעות פעולות על הערימה ש"נבלעה".)
- **consolidation**: כאשר יתבצע consolidation (עקב delete-min), הוא יבוצע במלואו, ולא בצורת one-pass (אפשרות נוספת שמופיעה במצגת). רשימת העצים הסופית (החדשה), מסודרת לפי דרגות בסדר עולה. כלומר, העץ השמאלי ביותר הוא בעל הדרגה הכי נמוכה.
- **decrease-key**: אם לאחר עדכון מפתח אין הפרה של תנאי הערימה בין הקודקוד לאביו, לא מתבצע חיתוך. אם התבצע חיתוך, תת-העץ שנחתך ייכנס לראש הרשימה (משמאל).
- **delete-min**: הצאצאים של איבר המינימום יישארו עם אותו סדר-יחסי ביניהם. בנוסף, הם גם יישארו באותו מיקום יחסי ברשימה, ולא יזוזו להתחלה. כלומר: אם סדרת העצים היא y_1, y_2, y_3 כאשר y_2 המינימום ובניו $x_1, \dots, x_k$, נקבל את הסדר $y_1, x_1, \dots, x_k, y_3$. או בפסאודו-קוד: min.child.prev=min.prev, min.child.prev.next=min.next.
- **mark**: יש לדאוג ששורשי עצים לעולם אינם מסומנים (אי אפשר לתלוש שורשים). זה רלוונטי גם במקרה של קודקוד מסומן שאביו נמחק עקב delete min (לפני ביצוע consolidate).

<u>חשוב</u>: אם יצוצו עניינים נוספים, יוקדש עבורם שרשור הבהרות מימוש בפורום התרגיל, באתר הקורס. יש לעקוב אחרי השרשור כדי להתעדכן.

סיבוכיות

יש לתעד בקוד ובמסמך נפרד (ביתר פירוט) את סיבוכיות זמן הריצה במקרה הגרוע (האסימפטוטית, במונחי O הדוקים) של כל פונקציה, כתלות במספר האיברים בעץ n. יש להשיג סיבוכיות זמן ריצה (במקרה הגרוע ביותר) נמוכה ככל הניתן עבור כל אחת מהפונקציות.

יש לתעד את סיבוכיות זמן הריצה (במקרה הגרוע) של כל פונקציה, כתלות במספר האיברים בערימה. צריך להשיג סיבוכיות זמן ריצה זהה לזאת שנלמדה בכיתה עבור פעולות של מבנה הנתונים (זמני worst case (WC) וגם amortized).
לגבי פונקציות שלא נלמדו, יש להשיג זמן ריצה אסימפטוטי טוב ביותר במקרה הגרוע (WC), בלי לפגוע בזמני הריצה WC/amortized של הפעולות שנלמדו.

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.

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 מתקמפל, אלא גם נכון!

בקובץ שתגישו **לא תהיה פונקציית main**, דבר זה יפגע בטסטר שיבדוק את התרגילים. אם הפרויקט מתקמפל לבדו (ללא טסטר), זה סימן שמשהו לא נכון במימוש.
הקוד ייבדק על מחשבי בית הספר על גירסא Java8.

הנחיות להשמשת סביבת העבודה בבית (ג'אווה+אקליפס):
http://courses.cs.tau.ac.il/software1/1415b/misc/workenv.pdf
מדריך לעבודה עם Eclipse (סעיפים 9-5, 15):
http://www.vogella.com/
הנחיות לפתיחת חשבון מחשב, למי שמעוניינ/ת לעבוד במעבדת בית הספר:
http://cs.tau.ac.il/system/accounts0
שימוש בג'אווה 8 במעבדות האוניברסיטה:
http://courses.cs.tau.ac.il/software1/1415b/misc/lab-eclipse.pdf

חלק ניסויי/תיאורטי
בשאלות שלהלן נניח, כמובן, את ההתנהגות האחידה שהוגדרה במסגרת הבהרות המימוש.

שאלה 1:
בשאלה זו נציג סדרת פעולות, ונשאל מה יקרה אם נבצע/לא נבצע פעולות מסוימות. **מומלץ מאוד** להיעזר בקוד שנכתב ולכתוב תוכנית שתבצע את הפעולות והשינויים המתוארים כדי לוודא את נכונות התשובות (אין צורך להגישה).

נתון m שהוא חזקה של 2 (לדוגמה: $m = 2^{10}$). נבצע את הפעולות הבאות:

```
for k=m-1, m-2, ..., 0, -1: insert(k)
deleteMin()
for i=log2(m), ..., 2, 1: decreaseKey(m - 2^i + 1 , m + 1)
```

המלצות:

1. מומלץ לבנות ידנית את המבנה המתקבל עבור m=2,4,8,16 כדי להבין לאילו איברים בחרנו להקטין את המפתח... "למה דווקא הם".
2. כדי לקרוא ל-decreaseKey כראוי, מומלץ לשמור מערך מצביעים לכל הקודקודים בשלב הבנייה של הערימה (שורה #1).

שאלות:

א. מהו זמן הריצה האסימפטוטי (במונחי "Big O") של סדרת הפעולות כפונקציה של m?
ב. מלאו את הטבלה הבאה עבור $m = 2^5, 2^{10}, 2^{15}, 2^{20}$. זמן-הריצה יימדד במילישניות (אלפיות שנייה). ערכי ה-total והפוטנציאל הם הערכים שיתקבלו בסוף הריצה על הסדרה.

m	Run-Time (ms)	totalLinks	totalCuts	Potential
2^5				
2^{10}				
2^{15}				
2^{20}				

ג. כמה **פעולות link** וכמה **פעולות cut** מתבצעות במהלך סדרת הפעולות, ומהו **הפוטנציאל** של המבנה בסוף הסדרה? יש לתת ביטוי מדויק התלוי ב-m.

בשלושת הסעיפים הבאים, ענו מחדש על הסעיף הקודם תחת השינויים שיתוארו. השינויים בלתי-תלויים, וכל סעיף משנה בנפרד ובאופן שונה את הפעולות שלעיל:

ד. אם בשורה #3 נבצע decreaseKey על המפתחות $m - 2^i$ (בלי ה-"+1").
ה. אם נמחק את שורה #2.
ו. אם נוסיף שורה נוספת: "decreaseKey(m-2,m+1)". במקרה זה, ציינו גם מהי העלות היקרה ביותר של פעולת decreaseKey (במונחים של m. עלות = מספר חיתוכים).

(תזכורת: מומלץ לוודא את התשובות התיאורטיות באמצעות מדידות.)

חשוב: כדי להקל על הבדיקה, יש לסכם את תשובות סעיפים ג'-ו' בטבלה כמו זו שלהלן. טבלה זו אינה מחליפה את הנימוקים הנדרשים עבור הביטויים(!):

case	totalLinks	totalCuts	Potential	decreaseKey max cost
(c) original				(skip)
(d) decKey(m-2^i)				(skip)
(e) remove line #2				(skip)
(f) added line #4				

שאלה 2:

כִּתבו תוכנית (לא להגשה) שתפעיל את הפעולות שמומשו, כדי לבצע את הסדרה הבאה ולענות על השאלות. בצעו הרצות **נפרדות** עבור $m = 3^i - 1$ כאשר $i = 6,8,10,12,14$ של הפעולות:

```
for k=0, 1, ..., m: insert(k)
for i=1, ..., 3m/4: deleteMin()
```

(כלומר: הכנסנו את הערכים ואז מחקנו מספר שלם מתוכם, $\frac{3m}{4}$.)

שאלות:

א. מלאו את הטבלה הבאה. זמן-הריצה יימדד במילישניות (אלפיות שנייה). ערכי ה-total והפוטנציאל הם הערכים שיתקבלו בסוף הריצה על הסדרה.

m	Run-Time (ms)	totalLinks	totalCuts	Potential
728				
...				

ב. מהו זמן הריצה האסימפטוטי (במונחי "Big O") של סדרת הפעולות כפונקציה של m?

ג. כמה **פעולות link** וכמה **פעולות cut** מתבצעות במהלך סדרת הפעולות, ומהו **הפוטנציאל** של המבנה בסוף הסדרה? יש לתת ביטוי מדויק התלוי ב-m.

(הערה: לשניים מתוך השלושה אין "נוסחה סגורה" על-ידי פונקציות אלמנטריות ופעולות חשבון, אך יש תיאור מאוד ברור כתלות ב-m. רצוי לשחק עם ערכים קטנים כדי להבחין בקשר, $m = 4,8,12, \ldots$. שימו לב לבחור כפולות של 4, כדי שמספר המחיקות יהיה שלם.)

(וודאו שהתשובות התיאורטיות תואמות למדידות שהתקבלו בסעיף א...)

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 FibonacciHeap.java.
2. קובץ טקסט info.txt המכיל את פרטי הזוג: מספר ת"ז, שמות, ושמות משתמש.
3. מסמך תיעוד חיצוני, המכיל גם את תוצאות המדידות. את המסמך יש להגיש באחד הפורמטים הבאים: doc, docx או pdf.

שמות קובץ התיעוד וקובץ הzip צריכים לכלול את שמות המשתמש האוניברסיטאיים של **הזוג המגיש** לפי הפורמט .../FibonacciHeap_idstudent1_ idstudent2.pdf/doc/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!