**ממ"ן 14 – אופן פעולה וזמני ריצה**

על התוכנית

את התוכנית כתבתי ב-CPP ומקומפלת עם CMake. סביבת ההרצה שלי הייתה לינוקס (wsl) ולכן התייחס פה להרצה מ-terminal של קובץ הרצה elf.

התוכנית מייצא מבנה נתונים MergeableHeap אשר בעצם הינו רשימה מקושרת (linked-list) ולה מצבים ופונקציות שונות.

התוכנית רצה על פקודות הנקלטות כקלט אינטראקטיבי מהמשתמש או כקובץ. הפקודות רצות אחת אחרי השנייה ומצב הרשימה האחרונה מודפס למסך לאחר כל פעולה. בסוף התוכנית, מודפסות הרשימות הסופיות.

פונקציות נתמכות:

* MakeHeap – מייצר רשימה ריקה חדשה.
* Insert X – מכניס ערך מספר X לתוך הרשימה האחרונה שנוצרה.
* Minimum – מדפיס את הערך המינימלי ברשימה האחרונה.
* ExtractMinimum – מדפיס את הערך המינימלי **ומוציא** אותו מתוך הרשימה האחרונה.
* Union – מאחד את שני הרשימות האחרונות שנוצרו לרשימה אחת וממשיך לעבוד איתה. בעת ביצוע פעולה זו שוב, נתייחס לרשימה המאוחדת כרשימה אחת ונאחד אותה את הרשימה שלפנייה (לדוגמה, אם איחדנו את רשימות 2 ו-3, הן יהפכו להיות רשימה 2, ואיחוד נוסף יאחד את 2 ו-1).

מצבי הרשימה – תכונות הרשימה נשמרות לאורך כל התוכנית:

* Sorted – רשימה ממוינת מהערך הנמוך לגבוה.
* Unsorted – רשימה של ערכים בסדר בהם נוצרו.
* Foreign – בדומה ל-Unsorted, **ללא ערכים כפולים**. משמע, פעולת הכנסה לערך קיים אינה תתבצע וכן Union בין 2 רשימות תאחד ערכים הקיימים ב2 הרשימות לערך בודד. לכל פעולה יש פלט מתאים.

Sorted – אלגוריתמים וזמני ריצה

**Insert - הכנסה**

בכדי להכניס איבר לרשימה יש לבצע שינויים על ערכי הצומת ב-next,prev ל-3 צמתים לכל היותר בכדי ליצור את המצב הבא: node\_1 -> node\_2 -> node\_3 כאשר כל צומת יודעת להצביע קדימה ואחורה ברשימה, ולכן ההכנסה עצמה קוראת ב-O(1).

האלגוריתם עובר על הרשימה, מהראש (האיבר הראשון) תחילה וממשיך לטייל ברשימה כל עוד ערך הצומת הנוכחית קטן מערך הצומת שנרצה להכניס.

בסיום רצית הלולאה, ישנם 3 מצבים:

1. רצנו עד סוף הרשימה, ולכן נכניס את הצומת הרצויה לסוף הרשימה ונגדיר אותה כזנב חדש – **המקרה הכי גרוע.**
2. לא זזנו מהמקום (האטורטר לא התקדם כלל), ולכן נכניס את הצומת הרצויה לתחילת הרשימה ונגדיר אותה כראש חדש.
3. מצאנו צומת ברשימה שערכה גדול מערך הצומת להכנסה, ולכן נכניס את הצומת הרצויה מאחורי הצומת שמצאנו.

סך הכל, במקרה הכי גרוע (1), רצנו על כל המערך פעם אחת בזמן לינארי למציאת המקום שנרצה להכניס את הצומת וביצענו את ההכנסה ב-O(1), **ולכן זמן הריצה של כלל הפונקציה הינה לינארי O(n).**

**Minimum – מציאת איבר מינימום**

מאחר והרשימה ממוינת מהערך הנמוך לגבוה, האיבר הראשון הינו המינימלי וניתן להשיג אותו ב-**O(1)** בכך ששמרנו תמיד מצביע לאיבר הראשון.

**ExtractMinimum – הוצאת האיבר המינימלי**

בכדי להוציא איבר מהרשימה, יש לקחת את המצביעים של ה-next,prev לכל היותר ל-2 צמתים שמקיפים את הצומת הרצויה ולעדכן כך שיצביעו אחד על השני, ולכן ההוצאה קוראת ב-O(1).

הוצאת האיבר המינימלי משתמשת בלוגיקה זו לאחר שמבצעת פעולת Minimum אשר גם היא מתבצעת ב-O(1).

**לכן, בסך הכל זמן הריצה של הפעולה הינו O(1).**

**Union – איחוד**

משתמש בפונקצית ה-Insert ובו זמנית מתקדם ב-2 הרשימות. זאת אומרת, אם נאחד את רשימה 2 לרשימה 1, ניקח את האיבר הראשון ברשימה 2 ונשתמש ב-Insert בשביל למצוא את המקום המתאים לו ברשימה 1. נחזיר את המקום הנ"ל מהפונקציה Insert מאותו מקום נחפש את המקום הבא להכניס את האיבר השני ברשימה 2 וכך הלאה.

זאת אומרת, לכל היותר נרוץ עד סוף הרשימה הראשונה ולאחר מכן נכניס את כל איברי הרשימה השנייה אחד אחר אחד.

כל פעולת הכנסה כזאת מתבצעת ב-O(1) והרי אנו רצים סך הכל על אורך שני הרשימות פעם אחת **ולכן זמן הריצה הינו לינארי כסכום אורכי הרשימות ולכן O(n)**.

Unsorted – אלגוריתמים וזמני ריצה

**Insert – הכנסה**

הראנו קודם לכן שזמן הריצה של הכנסה בודדת הינו O(1) ובמקרה זה אנו חוסכים את הריצה על הרשימה וישר מכניסים את האיבר החדש כזנב הרשימה, **ולכן זמן הריצה הינו O(1)**.

**Minimum – מציאת איבר מינימום**

מאחר והרשימה אינה ממוינת, נרוץ איבר אחד אחרי השני ברשימה ונשמור בכל זמן את המינימום האחרון שמצאנו. בסיום הריצה על אורך כל הרשימה, נקבל את האיבר המינימלי ביותר וסך הכל **רצנו בזמן לינארי על הרשימה O(n)**.

**ExtractMinimum – הוצאת האיבר המינימלי**

כמו שנאמר קודם לכן, זמן הריצה של תהליך ההוצאה הינו O(1) ולכן הזמן הקובע פה יהיה של מציאת המינימום **שכן הוא O(n)**.

**Union – איחוד**

מאחר ואין חוקיות על סדר האיברים ברשימה, נחבר את הזנב של רשימה אחת לראש של האחרת ונגדיר זנב חדש לזנב של הרשימה האחרת **וסך הכל ביצענו פעולות קבועות בזמן ריצה של O(1)**.

Unsorted, Foreign – אלגוריתמים וזמני ריצה

**הערה:** בכל רשימה מנעתי ערכים כפולים וכן באיחוד הרשימות איחדתי ערכים כפולים אם היו עם פלט מתאים לכל פעולה.

**Insert – הכנסה**

מאחר והרשימה לא ממוינת, נרוץ תחילה על כל הרשימה על מנת לענות על התשובה האם ערך האיבר החדש נמצא ברשימה – סך הכל בזמן ריצה של O(n). לאחר מכן, אם נבין שהערך לא קיים, נכניס את האיבר בדומה לרשימה לא ממוינת ב-O(1).

**סך הכל זמן ריצה של O(n)**.

**Minimum – מציאת איבר מינימום**

בדומה לרשימה לא ממוינת – O(n).

**ExtractMinimum – הוצאת האיבר המינימלי**

בדומה לרשימה לא ממוינת – O(n).

**Union – איחוד**

אם נאחד את רשימה 2 לתוך רשימה 1, לכל איבר ברשימה 2 נבדוק האם קיים ברשימה אחת בדומה לנעשה ב-Insert ולכן O(n). אם האיבר אינו קיים נבצע הכנסה בדומה למערך לא ממוין ב-O(1).

**סך הכל, על איבר ברשימה 2 רצנו במקרה הכי גרוע על כל האיברים ברשימה 1 ולכן זמן הריצה הינו O()**.