ממ"ן 15 – מבני נתונים ומבוא לאלגוריתמים

**מגישה: סיניה נודל 203191663**

**תיאור התכנית**

התוכנית מייצגת כלי עזר בניהול בית מלאכה לאריזת מתנות. היא מקבלת מהמשתמש פקודות שיש לבצע על מלאי המתנות ולפיכך מדפיסה פלט עם מידע על הפעולה שבצעה ועל תוצאתה.

התוכנית כתובה בשפת JAVA וחלק גדול מהפונקציות הוא מימוש של פסאודו קוד לפונקציות שנלמדו בקורס עבור עץ חיפוש בינארי.

קלט מהמשתמש מתקבל על ידי תיבות דו שיח, על ידי פעולות ישירות שיש לכתוב בתיבה או על ידי הכנסת שם קובץ ממנו יש לקרוא פקודות.

התכנית משתמשת במבנה נתונים של עץ חיפוש בינארי בשתי דרכים:

1. עץ לפי מפתח נפח האריזה, כאשר כל צומת מכיל מצביע לעץ עם גבהי החבילות שיש במלאי עבור אותו הנפח.
2. עץ לפי מפתח צלע הבסיס, כאשר כל צומת מכיל מצביע לעץ עם גבהי החבילות שיש במלאי עבור אותו הבסיס. כל צומת צלע בסיס גם מכיל ערך של גובה מקסימלי שיש בעץ המקושר אליו.

המבנים שהרחבתי יורשים ממחלקת עץ בינארי בסיסית שמימשתי ותומכים בפעולות של הוספת קופסא, הוצאת קופסא, החזרת ממדי קופסא בנפח מינימלי ובדיקה האם קיימת קופסא מתאימה במלאי.

המבנה הראשון תוכנן כדי לתמוך במציאת קופסא בנפח מינימלי – שכן סריקה תוכית של עץ חיפוש בינארי מאפשרת מעבר על הצמתים באופן ממוין מהערך הקטן לגדול.

המבנה השני תוכנן כדי לתמוך בבדיקה יעילה יותר עבור קיום של קופסא עם ממדים מתאימים כלשהם. נעשית סריקה של צמתי עץ צלעות הבסיס תוך בדיקה של ערך מקסימום הגובה הקיים במלאי.

לכל צומת בעצים ישנו ערך count שמאפשר כפילות ערכים וקופסאות במלאי.

**שיטות וסיבוכיות**

**הכנסה:**

מכיוון שיש שימוש ב2 מבני נתונים, יש לבצע הכנסה של קופסא על כל אחד מהם בנפרד.

שני המבנים משתמשים במחלקה של עץ חיפוש בינארי ובפונקציית הכנסת הצומת שלה.

הפונקציה הבסיסית מקבלת ערך מפתח ובודקת אם העץ ריק (=שורש לא מוגדר) ואם כן, הצומת החדש יוכנס כשורש.

אם לא, ישנה לולאה שמאפשרת ניווט בעץ למיקום ההכנסה הנכון כאשר לבן ימני יוכנס ערך גדול מצומת נתון, ולבן שמאלי יוכנס ערך נמוך. אם הערך זהה – ערך הcount של אותו צומת יעלה.

הסריקה מתחילה בשורש וממשיכה בלולאה בכל השורשים של תתי העצים עד מציאת המקום הפנוי.

הפונקציה מחזירה את הצומת החדש שהוכנס או את הצומת הקיים שערך הcount שלו עודכן.

מכיוון שהסריקה של העץ עבור ההכנסה היא מהשורש למורד העץ – זמן הריצה במקרה הגרוע הוא כגובה העץ.

הכנסה לעץ הנפחים:

insertValues(double height , double side) במחלקה VolumesTree פונקציית ההכנסה מקבלת ערך צלע בסיס וערך גובה ומשתמשת פעמיים בפונקציית ההכנסה הבסיסית של עץ החיפוש הבינארי insertNode במחלקה BinaryTree.

בפעם הראשונה ישנו חישוב נפח הקופסא לפי הנתונים, וכך הכנסת הערך לעץ.

בפעם השנייה משתמשים בצומת הנפח שעודכן, ומכניסים לעץ הגבהים המקושר אליו את הגובה הנתון.

זמן ריצה: עבור עץ הנפחים nm זהו מספר הנפחים האפשרי, ומכיוון שזהו עץ בינארי אז הגובה יהיה log(nm). לכל נפח יכולים להיות במקסימום n גבהים ולכן גובה כל עץ גבהים הוא log(n).

מכיוון שכל פעולת הכנסה לכל עץ מתאים תתבצע פעם אחת סה"כ העלות עבור פעולת הכנסה היא log(nm)+log(n).

הכנסה לעץ הצלעות:

insertValues(double height , double side) במחלקה SizesTree

מקבלת ערך צלע בסיס וערך גובה ומשתמשת פעמיים בפונקציית ההכנסה הבסיסית של עץ החיפוש הבינארי insertNode במחלקה BinaryTree.

בפעם הראשונה ישנה הכנסה של צלע הבסיס לעץ הצלעות ובדיקה אם ערך המקסימום של הגובה דרוש עדכון (אם הגובה הנוכחי שהוכנס גבוה מהערך הקיים). לאחר מכן יש הכנסה של ערך הגובה לעץ הגבהים של צומת צלע הבסיס הנתון.

זמן ריצה: יכולים להיות במקסימום m צלעות בסיס ולכן גובה כל עץ צלעות הוא log(m). לכל עץ כזה יש עץ גבהים כאשר במקסימום הגובה הוא log(n).

מכיוון שכל פעולת הכנסה לכל עץ מתאים תתבצע פעם אחת סך העלות עבור פעולת הכנסה היא log(m)+log(n).

סך זמן ריצה עבור הפעולה הכוללת לשני מבני הנתונים:

log(n) + log(m) + log(mn)2

כלומר, log(nm) אם מתייחסים לסדר הגודל המשמעותי.

**מחיקה:**

מכיוון שיש שימוש ב2 מבני נתונים, יש לבצע מחיקה של קופסא על כל אחד מהם בנפרד.

שני המבנים משתמשים במחלקה של עץ חיפוש בינארי ובפונקציית מחיקת הצומת שלה removeNode(TreeNode node).

קוד פונקציית מחיקת הצומת עבור עץ החיפוש הבינארי הוא מימוש של הפסאודו קוד המתואר בספר הלימוד בעמ' 221 עם שינוי עבור תמיכה בעדכון count של צומת.

הפונקציה מקבלת מצביע לצומת אותו יש למחוק ומחסירה 1 מערך הcount. אם לאחר ההחסרה, ערך הcount שווה ל0 אז יש למחוק את הצומת מהעץ באותו מימוש של הפסאודו קוד הנלמד בקורס. במקרה בו יש מחיקת צומת זמן הריצה הוא גובה העץ.

הפונקציה משתמשת בפונקציה למציאת עוקב המופיעה בעמ' 218 בספר הלימוד, כאשר הפסאודו קוד תורגם לשפת ג'אווה.

הפונקציה מחזירה את הצומת שערך הcount שלו עודכן (כאשר עדיין יש במלאי לאחר ההחסרה) או null אם כבר אינו במלאי.

מחיקה מעץ הנפחים:

הפונקציה removeValues מקבלת את מידת צלע הבסיס וגובה הקופסא, מחשבת את נפח הקופסא, משתמשת בפונקציית חיפוש צומת בעץ לפי מפתח הנפח, ומעבירה לפונקציית המחיקה הבסיסית את המצביע לצומת שנמצא. אם הצומת שעבר החסרה לא שווה לnull לאחר הפעולה, זה אומר שאותו צומת שערך הcount שלו הופחת צריך לעבור בנוסף מחיקת צומת בעץ הגבהים המקושר לצומת הנפח.

לכן ישנו חיפוש צומת לפי ערך הגובה בעץ המקושר לצומת, ושוב הפעלת פונקציית המחיקה הבסיסית על צומת זה.

זמן הריצה: פעולת חיפוש בעץ הנפחים תעלה log(nm) ובמקרה בו צומת נמצא והופחת יידרש לבצע גם חיפוש ומחיקה מעץ הגבהים המקושר שיעלה בנוסף log(n) עבור חיפוש וlog(n) עבור מחיקת צומת אם נדרש. סך סדר הגודל בהתייחסות לגורם המשמעותי הוא log(nm).

מחיקה מעץ הצלעות:

הפונקציה removeValues מקבלת את מידת צלע הבסיס וגובה הקופסא, משתמשת בפונקציית חיפוש צומת בעץ לפי מפתח צלע הבסיס, ומעבירה לפונקציית המחיקה הבסיסית את המצביע לצומת שנמצא. אם הצומת שעבר החסרה לא שווה לnull לאחר הפעולה, זה אומר שאותו צומת שערך הcount שלו הופחת צריך לעבור בנוסף מחיקת צומת בעץ הגבהים המקושר לצומת צלע הבסיס.

לכן ישנו חיפוש צומת לפי ערך הגובה בעץ המקושר לצומת, ושוב הפעלת פונקציית המחיקה הבסיסית על צומת זה.

ישנה בדיקה של מקרה בו צומת הגובה שנמחק היווה את ערך המקסימום בעץ. במקרה כזה, יש לבדוק מקסימום חדש ע"י פונקציית מציאת מקסימום בעץ בינארי. בעץ חיפוש בינארי בהכרח ערך המקסימום הוא הבן הימני ביותר מפני שכל בן ימני בעל ערך גדול מזה של צומת ההורה. עלות חיפוש ערך המקסימום היא כגובה העץ. ערך המקסימום החדש יושם בנתוני צומת צלע הבסיס אליה עץ הגבהים הנתון מקושר.

זמן הריצה: פעולת חיפוש בעץ צלעות הבסיס תעלה log(m) ובמקרה בו צומת נמצא והופחת יידרש לבצע גם חיפוש ומחיקה מעץ הגבהים המקושר שיעלה בנוסף log(n) עבור חיפוש וlog(n) עבור מחיקת צומת אם נדרש. אם צומת הגובה שהיווה את המקסימום נמחק- עלות מציאת מקסימום היא עוד log(n). סך סדר הגודל בהתייחסות לגורם המשמעותי הוא log(n)+log(m).

סך זמן ריצה עבור הפעולה הכוללת לשני מבני הנתונים:

סדר גודל של log(nm) בהתייחסות לגורם המשמעותי.

**החזרת ממדי הקופסא בנפח המינימלי:**

הפונקציה getValues בעץ הנפחים מקבלת את מידת צלע הבסיס ומידת גובה הקופסא ומחזירה את ממדי הקופסא בנפח המינימלי שצלעה לפחות side וגובהה לפחות height. אם לא נמצאה קופסא, יוחזר NULL.

תחילה, מחשבת את נפח הקופסא המינימלי האפשרי לפי הצלעות, ומשתמשת במחסנית עזר על מנת לדמות סריקה תוכית בעץ חיפוש בינארי. הפסאודו קוד הנלמד בקורס עבור סריקה תוכית בעץ חיפוש בינארי בעמ' 214 משתמש ברקורסיה במקום במחסנית עזר, אך הדרך מתבצעת באופן זהה כשם שפונקציה רקורסיבית משתמשת במחסנית הזיכרון עבור ערך הפונקציה מאחורי הקלעים. הסריקה מכניסה את צמתי העץ למחסנית לפי הסדר צומת הורה ואז בן שמאלי. לאחר מכן מוציאה מראש המחסנית את הבן השמאלי, ההורה ורק אז יש בדיקה של בן ימני. מפני שעץ חיפוש בינארי שומר על תכונת בן שמאלי קטן מהורה ובן ימני גדול מהורה – הסריקה תתבצע בסדר ממוין מהערך הקטן לגדול.

עם כל בדיקה של צומת מהקטן לגדול, אם נפח הצומת גדול או שווה לנפח המינימלי האפשרי אז יש בדיקה בעץ הגבהים המקושר לצומת אם נמצא גובה בטווח מתאים. כאשר נפח נמצא מתאים, לא בהכרח ממדי הקופסא תואמים לדרישה.

הגובה צריך להיות לפחות height וצלע הבסיס לפחות side. לכן ישנו חישוב של ערך גובה הקופסא המקסימלי האפשרי לפי ערך הנפח הנבדק ולפי מצב בו side הוא מינימלי. לאחר החישוב יש שימוש בפונקציית עזר isNodeExistBetween המקבלת מצביע לשורש עץ, ערך מינימלי וערך מקסימלי. כך מחפשת בעץ צומת המקיים ערך כלשהו בטווח. פונקציה זו משתמשת בלולאה שבודקת אם שורש תת העץ מתאים לטווח, אם ערך השורש נמוך מהמצופה – בודקת את תת העץ של הבן הימני, ואם הערך גבוה- בודקת את תת העץ של הבן השמאלי. אם נמצא צומת בטווח – הצומת מוחזרת. אם לא נמצא צומת תואם כלל, מוחזר NULL. במקרה הגרוע תערך סריקה לכל גובה העץ וזה יהיה זמן הריצה.

אם הפונקציה החזירה צומת קיים – כלומר, ישנו גובה בעץ הנפחים שתואם לדרישה, אזי ישנו חישוב של צלע הבסיס האקטואלית והחזרת ממדי הקופסא כאובייקט box המכיל את הנתונים שנמצאו.

אם לא נמצא גובה בטווח, הסריקה נמשכת עבור שאר צמתי עץ הנפחים , עם סריקה נוספת של עץ הגבהים עבור כל נפח.

הנכונות של הפונקציה למציאת קופסא מתאימה בנפח מינימלי היא ע"י הסריקה הממוינת של עץ הנפחים מערך קטן לגדול, כך שאם נמצא נפח מתאים – בהכרח נבדק שערכי הנפחים הקטנים ממנו לא התאימו.

ע"י חישוב של ערך גובה מינימלי ומקסימלי לקופסא, ובדיקה בעץ הגבהים המקושר לאותו נפח, בהכרח הנפח המתאים גם כולל ערכי צלע בסיס וגובה לפי הדרישה.

זמן ריצה:

* סריקה תוכית של עץ הנפחים במקרה גרוע תבדוק כל צומת בעץ ולכן מס' הצמתים בעץ, כמספר הנפחים הוא nm.
* עבור כל צומת נפח מתאים תבוצע בדיקה על עץ הגבהים עם הפונקציה isNodeExistBetween שזמן ריצתה log(n) כגובה עץ הגבהים.

סך כל העלות היא nm\*log(n).

**בדיקה עבור המצאות קופסא מתאימה כלשהי במלאי:**

הפונקציה checkValues בעץ הצלעות מקבלת את מידת צלע הבסיס ומידת גובה הקופסא ומחזירה ערך בוליאני TRUE אם נמצאה קופסא כלשהי במידות לפחות side וגובה לפחות height. אם לא נמצאה קופסא, יוחזר FALSE.

גם פונקציה זו משתמשת במחסנית עבור סריקה תוכית של עץ צלע הבסיס לפי ערך קטן לגדול. אם נמצא ערך side מתאים – ישנה בדיקה שערך גובה המקסימום בנתוני הצומת הוא לפחות ערך הגובה שניתן כפרמטר לפונקציה. הנכונות נובעת מכך שבהכרח בדיקה של ערך גובה מקסימלי מחליפה סריקה של כל צמתי הגבהים עבור צלע הבסיס, מכיוון שאם ערך המקסימום נמוך מהגובה המצופה, לא קיים בעץ צומת שערכו גבוה מהנתון הרלוונטי.

זמן ריצה:

* סריקה תוכית של עץ הנפחים במקרה גרוע תבדוק כל צומת בעץ ולכן מס' הצמתים בעץ, כמספר צלעות הבסיס הוא m.
* עבור כל צומת צלע בסיס מתאים תבוצע פעולה בזמן קבוע עבור בדיקת ערך מקסימום גובה.

סך כל העלות היא m.