**תיעוד FibonacciHeap.HeapNode**

כל node ייוצג על ידי:

int Key – המפתח של הצומת.

int rank – הדרגה של הצומת – מספר הצמתים שצומת זה הוא ההורה שלהם.

boolean mark – הסימן של הצומת – סימון הצומת ב-true במידה ואחד הילדים שלו נחתך (כמו שנלמד בשיעור).

HeapNode child – מצביע לילד של הצומת.

HeapNode next – מצביע לצומת הבא.

HeapNode prev – מצביע לצומת הקודם.

HeapNode parent – מצביע להורה של הצומת.

HeapNode pointerKmin – שדה זה משתנה רק בקריאה לפונקציה KMin(FibonacciHeap H, int k). השדה מצביע לצומת בעלת אותו מפתח בערימה המקורית.

להלן הפונקציות במחלקה:

|  |  |  |
| --- | --- | --- |
| **סיבוכיות זמן ריצה** | **תיאור הפונקציה** | **הפונקציה** |
| סיבוכיות זמן ריצה . | הבנאי של המחלקה. | public HeapNode(int key) |
| סיבוכיות זמן ריצה . | הפונקציה מחזירה את ה-key של הצומת. | public int getKey() |
| סיבוכיות זמן ריצה . | הפונקציה מחזירה את ה-rank של הצומת. | Public int getRank() |
| סיבוכיות זמן ריצה . | הפונקציה מחזירה את ה-mark של הצומת. | Public boolean getMark() |
| סיבוכיות זמן ריצה . | הפונקציה מחזירה את ה-child של הצומת. | Public HeapNode getChild() |
| סיבוכיות זמן ריצה . | הפונקציה מחזירה את ה-next של הצומת. | Public HeapNode getNext() |
| סיבוכיות זמן ריצה . | הפונקציה מחזירה את ה-prev של הצומת. | Public HeapNode getPrev() |
| סיבוכיות זמן ריצה . | הפונקציה מחזירה את ה-parent של הצומת. | Public HeapNode getParent() |
| סיבוכיות זמן ריצה . | הפונקציה מחזירה את ה-pointerKmin של הצומת. | Public HeapNode getpointerKmin() |
| סיבוכיות זמן ריצה . | הפונקציה מקבעת את ה-key של הצומת. | Public void setKey(int key) |
| סיבוכיות זמן ריצה . | הפונקציה מקבעת את ה-rank של הצומת. | Public void setRank(int rank) |
| סיבוכיות זמן ריצה . | הפונקציה מקבעת את ה-mark של הצומת. | Public void setMark(boolean mark) |
| סיבוכיות זמן ריצה . | הפונקציה מקבעת את ה-child של הצומת. | Public void setChild(HeapNode child) |
| סיבוכיות זמן ריצה . | הפונקציה מקבעת את ה-next של הצומת. | Public void setNext(HeapNode next) |
| סיבוכיות זמן ריצה . | הפונקציה מקבעת את ה-prev של הצומת. | Public void setPrev(HeapNode prev) |
| סיבוכיות זמן ריצה . | הפונקציה מקבעת את ה-parent של הצומת. | Public void setParent(HeapNode parent) |
| סיבוכיות זמן ריצה . | הפונקציה מקבעת את ה-pointerKmin של הצומת. | Public void setpointerKMin(HeapNode node) |

**תיעוד FibonacciHeap:**

כל FibonacciHeap (ערימת פיבונצ'י) תיוצג על ידי:

HeapNode min – מצביע לצומת בעלת המפתח המינימלי בערימה.

HeapNode first – מצביע לצומת הראשונה בערימה (הצומת הכי שמאלית/הצומת הכי חדשה).

Int size – מספר הצמתים בערימה.

Int numOfTrees – מספר העצים בערימה.

Int nonMarked – מספר הצמתים בערימה שה-mark שלהם הוא false.

Int highestDeg – הדרגה של העץ הכי גדול בערימה.

כמו כן, עבור כל הערימות פיבונצ'י שנוצרו במהלך ריצת התוכנית ישמרו הפרמטרים הסטטיים:

Static int cuts –מספר כל פעולות החיתוך שבוצעו מתחילת ריצת התוכנית. (תהליך זה יתבצע רק בקריאה ל-decreaseKey() כאשר מנתקים תת עץ מאביו)

Static int links – מספר כל פעולות החיבור שבוצעו מתחילת ריצת התוכנית. (תהליך זה יתבצע רק בקריאה ל-link(HeapNode x, HeapNode y))

להלן הפונקציות במחלקה:

|  |  |  |
| --- | --- | --- |
| **סיבוכיות זמן ריצה** | **תיאור הפונקציה** | **הפונקציה** |
| סיבוכיות זמן ריצה . | הבנאי של המחלקה. | public FibonacciHeap() |
| סיבוכיות זמן ריצה . | הפונקציה מחזירה את ה-min של הערימה. | Public HeapNode getMin() |
| סיבוכיות זמן ריצה . | הפונקציה מחזירה את ה-first של הערימה. | Public HeapNode getFirst() |
| סיבוכיות זמן ריצה . | הפונקציה מחזירה את ה-size של הערימה. | Public int getSize() |
| סיבוכיות זמן ריצה . | הפונקציה מחזירה את ה-numOfTrees של הערימה. | Public int getNumOfTrees() |
| סיבוכיות זמן ריצה . | הפונקציה מחזירה את ה-nonmarked של הערימה. | Public int getNonMarked() |
| סיבוכיות זמן ריצה . | הפונקציה מחזירה את ה-highestDeg של הערימה. | Public int getHighestDeg() |
| סיבוכיות זמן ריצה . | הפונקציה מחזירה את cuts. | Public static int getCuts() |
| סיבוכיות זמן ריצה . | הפונקציה מחזירה את ה-links. | Public static int getLinks() |
|  |  |  |
| סיבוכיות זמן ריצה . | הפונקציה מקבעת את ה- min של הערימה. | Public void setMin(HeapNode min) |
| סיבוכיות זמן ריצה . | הפונקציה מקבעת את ה- first של הערימה. | Public void setFirst(HeapNode first) |
| סיבוכיות זמן ריצה . | הפונקציה מקבעת את ה- size של הערימה. | Public void setSize(int size) |
| סיבוכיות זמן ריצה . | הפונקציה מקבעת את ה- numOfTrees של הערימה. | Public void setNumOfTrees(int numOfTrees) |
| סיבוכיות זמן ריצה . | הפונקציה מקבעת את ה-nonMarked של הערימה. | Public void setNonMarked(int nonMarked) |
| סיבוכיות זמן ריצה . | הפונקציה מקבעת את ה-highestDeg של הערימה. | Public void setHighestDeg(int highestDeg) |
| סיבוכיות זמן ריצה . | מעלה את הערך של cuts ב-1. | Public void incCuts() |
| סיבוכיות זמן ריצה . | מעלה את הערך של links ב-1. | Public void incLinks() |
|  |  |  |
| סיבוכיות זמן ריצה . | הפונקציה מחזירה true אם הערימה ריקה ואחרת false. | Public Boolean isEmpty() |
| הפונקציה רק משנה מצביעים ולכן סיבוכיות זמן הריצה שלה הוא . | הפונקציה יוצרת צומת שמפתחו הוא key, מכניסה את הצומת לערימה ומחזירה אותו. | Public HeapNode insert(int key) |
| הפונקציה קוראת ל-removeMin() שרצה בסיבוכיות ולאחר מכן קוראת לפונקציה consolidate() שרצה בסיבוכיות ב-W.C ו- באמורטייזד. לכן הפונקציה deleteMin() רצה בסיבוכיות ב-W.C ו- באמורטייזד. | הפונקציה מוחקת את הצומת בעלת המפתח המינימלי מהערימה. | Public void deleteMin() |
| רצה על כל הילדים של הצומת המינימלית ושנה את המצביעים שלהם – סיבוכיות . כל שאר הפעולות בפונקציה הן שינוי מצביעים ולכן סיבוכיות זמן הריצה של הפונקציה הוא . | הפונקציה מבצעת את עדכון המצביעים כך שלא יהיה ניתן להגיע לצומת המינימלית דרך הערימה. | private void removeMin() |
| הפונקציה רצה על כל השורשים של העצים בערימה – סיבוכיות ב-W.C ו- באמורטייזד, וקוראת לפעולות  link(HeapNode x, HeapNode y) ו- meldNode(HeapNode node) שרצות ב-. כל שאר הפעולות מתבצעות ב- ולכן סיבוכיות זמן הריצה של הפונקציה consolidate() הוא ב-W.C ו- באמורטייזד. | הפונקציה מאחדת את כל העצים בערימה כנלמד בשיעור (הגרסה שמבצעת את התהליך במלואו). | Private void consolidate() |
| הפונקציה רק משנה מצביעים ולכן סיבוכיות זמן הריצה שלה הוא . | הפונקציה מאחדת את שני העצים שבהם x ו-y הם השורשים של העצים (כנלמד בשיעור) ומחזירה את השורש של העץ הנוצר מהמיזוג. | Private HeapNode link(HeapNode x, HeapNode y) |
| סיבוכיות זמן ריצה . | הפונקציה מחזירה את ה-min של הערימה. | Public HeapNode findMin() |
| הפונקציה רק משנה מצביעים ולכן סיבוכיות זמן הריצה שלה הוא . | הפונקציה ממזגת את הערימה עם heap2 בצורת meld עצל (כפי שנלמד בשיעור). הערימה החדשה תיכנס מימין לערימה המקורית. | Public void meld(FibonacciHeap heap2) |
| הפונקציה רק משנה מצביעים ולכן סיבוכיות זמן הריצה שלה הוא . | הפונקציה ממזגת את הצומת (שיכולה להיות גם שורש של עץ) לערימה כמו בפונקציה הנ"ל. | Private void meldNode(HeapNode node) |
| סיבוכיות זמן ריצה . | הפונקציה מחזירה את ה-size של הערימה. | Public int size() |
| תחילה הפונקציה יוצרת מערך בגודל highestDeg שהוא לכל היותר . לאחר מכן הפונקציה רצה על כל השורשים של העצים ברשימה – סיבוכיות ב-W.C. סה"כ נקבל כי הסיבוכיות של הפונקציה היא . | הפונקציה מחזירה מערך מונים כך שבאינדקס מופיע מספר העצים בערימה שהסדר (rank) שלהם הוא . כלומר, היא מחזירה מערך של integers ,כך שלכל אינדקס בין 0 עד הדרגה המקסימלית של עץ שקיימת בערימה, הערך שמוחזר במערך הוא מספר העצים שקיימים בערימה מסדר . עבור ערימה ריקה יוחזר מערך ריק. | Public int[] countersRep() |
| הפונקציה בהכרח קוראת ל-deleteMin() שרצה ב- ב-W.C ו- באמורטייזד. כמו כן אם האיבר המינימלי אינו x, הפונקציה קוראת ל- decreaseKey(HeapNode x, int delta) שרצה בסיבוכיות בסיבוכיות ב-W.C ו- אמורטייזד. סה"כ נקבל כי הפונקציה רצה בסיבוכיות ב-W.C ו- באמורטייזד. | הפונקציה מוחקת את הצומת x מהערימה. | Delete(HeapNode x) |
| הפונקציה קוראת ל-cascadeCut(HeapNode x) שרצה בסיבוכיות ב-W.C ו- אמורטייזד. כל שאר הפעולות קורות ב-. לכן הפונקציה רצה בסיבוכיות ב-W.C ו- אמורטייזד. | הפונקציה מפחיתה את ערכו של המפתח של הצומת x בערך delta. כלומר, מתבצע . | Public void decreaseKey(HeapNode x, int delta) |
| הפונקציה פועלת בדומה לפונקציה שנלמדה בשיעור ולכן זמן הריצה שלה ב-W.C הוא וזמן האמורטייזד שלה הוא . | הפונקציה מבצעת cascadeCut כמו שנלמד בשיעור. | Private void cascadeCut(HeapNode x) |
| הפונקציה רק משנה מצביעים ולכן סיבוכיות זמן הריצה שלה הוא . | הפונקציה חותכת את הצומת x מהעץ של אביו ומכניסה את התת עץ ש-x הוא שורשו לראש הערימה (משמאל). לבסוף הפונקציה מחזירה את ההורה של x (לפני ביצוע החתך). | Private HeapNode cutAndInsertHeap(HeapNode x) |
| סיבוכיות זמן ריצה . | הפונקציה מחזירה את כמות האיברים שאינם marked בערימה. | Public int nonMarked() |
| סיבוכיות זמן ריצה . | הפונקציה מחזירה את ערך הפוטנציאל הנוכחי של הערימה. | Public int potential() |
| סיבוכיות זמן ריצה . | הפונקציה מחזירה את מספר כל פעולות החיבור שבוצעו מתחילת ריצת התוכנית. | Public static int totalLinks() |
| סיבוכיות זמן ריצה . | הפונקציה מחזירה את מספר כל פעולות החיתוך שבוצעו מתחילת ריצת התוכנית. | Public static int totalCuts() |
| בשיעורי בית 4 שאלה 3 הוכחנו כי סיבוכיות זמן הריצה של הפונקציה הוא .  נשים לב כי . לכן, מכיוון ש- הוא העץ היחיד בערימה, מתקיים:  .  מכאן נקבל כי סיבוכיות זמן הריצה של הפונקציה הוא . | פונקציה סטטית זו מקבלת ערימה H שהיא עץ שדרגתו , ומספר חיובי . הפונקציה מחזירה מערך ממוין של הצמתים הקטנים ב-. | Public static int[] kMin(FibonacciHeap H, int k) |