|  |  |  |  |
| --- | --- | --- | --- |
| מתודה | סיבוכיות | מחלקה | הסבר |
| addEdge | O(logn) | Graph | בזמן קבוע הפונק' ניגשת למיקום של כל אחד מהאיברים שמחוברים בקשת החדש, ומתווסף לרשימת השכנים בעת העפולה בעצם מתעדכנת ערימת המקסימום עבור כל אחד מהצמתים, משום שנוסף לו משקל. עלות של כל הפעפועים כלפי מעלה. בנוסף, משום שטבלת ההאש שלנו היא משפחה אוני' אנחנו צופים כי בתוחלת בכל תא יהיה כמות נמוכה של צמתים (פקטור העומס נמוך מאוד – טווח המספרים האפשריים עבור המפתחות והראשוני שהוגדר הם באותו סדר גודל) |
| maxNeighborhoodWeight | O(1) | Graph | שדה שמתוחזק במחלקה. שליפה של מידע מאוחסן בו תהיה בזמן קבוע. |
| getNeighborhoodWeight | O(1) | Graph | שדה שמתוחזק במחלקה של Node. שליפה של מידע מאוחסן בו תהיה בזמן קבוע. |
| deleteNode |  | Graph | מתודה מאתגרת. מספר פעולות קורות בה – היא מקבלת צומת וניגשת אליו בזמן קבוע בתוחלת. היא פונה לרשימת השכנים שלו, ובעצם עבור כל שכן (סה"כ d) היא ניגשת לשכן ומוחקת את עצמה מרשימת השכנים שלו. לאחר כל פעולה כזו, מתעדכנת ערימת המקסימום בהתאם לכך שמשקל השכנויות של השכנים שלה קטן. לבסוף ברגע שנגמר רשימת השכנים, הצומת נמחק ומתעדכנים מספר הצמתים החדשים בגרף. |
| getNumNodes | O(1) | Graph | שדה שמתוחזק במחלקה. שליפה של מידע מאוחסן בו תהיה בזמן קבוע. |
| getNumEdges | O(1) | Graph | שדה שמתוחזק במחלקה. שליפה של מידע מאוחסן בו תהיה בזמן קבוע. |
| Connect | O(1) | Neighbor | מקבלת שני צמתים, יוצרת ייצוגים עבורם ברשימת השכנים זה של זה ומקשרת ביניהם בעזרת מצביע. |
| addNeighbor |  | Node | מקבלת צומת שאמור להתווסף בתור שכן, מעדכנת את רשימת השכוניות של כל אחד, מעדכנת את ערימת המקסימום. |
| updateNeighborhoodWeight | O(1) | Node | מקבלת צומת, ניגשת אליו בתוחלת של זמן קבוע, ניגשת לשדה משקל השכנויות שלו ומעדכנת אותו בהתאם להוראה שקיבלה. |
| disassociateFromNeighbors | O(d) | Node | המתודה רצה על כל רשימת השכנים של צומת שאותו נרצה למחוק ובעצם מוחקת פיזית את החיבור בין השכנים לבין הצומת שנרצה למחוק. היא תעבור על כל d השכנים ועבור כל אחד תנתק את החיבור. |
| retreiveNode | O(1) | DoublyLinkedList | מחפשת את הצומת ברשימה המקושרת שנמצאת בכל תא בטבלת ההאש הגדולה. מצפים כי יהיה צומת אחד בכל תא ולכן הזמן קבוע. |
| Insert | O(1)\O(n) | DoublyLinkedList | הכנסה של צומת חדש לרשימה מקושרת שנמצאת בתא המיועד עבור האינדקס שלו בטבלת ההאש, או לחלופין להכנסת שכן חדש לרשימה מקושרת שמחזיקה את השכנים של כל צומת. |
| initializeHeap | O(n) | MaxHeap | אתחול סטנדרטי של ערימת מקסימום. |
| Delete | O(logn) | MaxHeap | מוחקת צומת (עם קבלת מצביע) מהערימה. ממקמת את האיבר האחרון במערך שמייצג את הערימה (כלומר את האיבר ברמה הנמוכה ביותר במיקום הימני ביותר, ומשם מתחילה פעפוע עד לקבלת ערימה תקינה שוב. |
| heapifyUp | O(1)-O(n) | MaxHeap | מבצעת חילוף במידת הצורך עם ההורה, במידה והצומת החדש גדול ממנו. יכולה להמשיך ברקורסיה עד להגעה אל השורש. |
| heapifyDown | O(1)-O(n) | MaxHeap | מבצעת חילוף במידת הצורך עם הבן בעל הערך הגדול יותר, במידה והצומת החדש קטן ממנו. יכולה להמשיך ברקורסיה עד להגעה אל העלים. מתבצע לאחר שנמחק צומת וצומת אחר נקרא לדגל להחליף אותה. |
| populateTable | O(N) | MyHashtable | הפונקציה המאתחל לטבלת ההאש שמחזיקה את כל Graph. מקבלת את הערכים, ויוצרת מערך שמסמל את טבלת ההאש, כך שכל תא בטבלת מצביע לאיבר רשימה מקושרת דו כיוונית ריקה וממתין לקראת הצמתים שיוכנסו. |
| accessNode | O(1) | MyHashtable | בתוחלת אנו צופים שנוכל לגשת לכל אחד מהתאים בזמן קבוע. |

\*מתודות שהוספנו והן רק שולפות מידע משדה לא ציינו בטבלה (היא נראתה לנו עמוסה גם ככה).

טבלת ההאש – מחזיקה אינדקס ומצביע לרשימה מקושרת דו כיוונית של צמתים, על פי גיבוב רנדומי בכל אתחול

בכל רשימה מקושרת יש מבנה שמחזיק את הצומת. המבנה מאפשר לצומת להחזיק מצביע אל רשימה מקושרת שתחזיק את כל הצמתים השכנים שלו. בנוסף, המבנה שמכיל את הצומת גם מצביע לאיבר המיקום בערימת המינימום מה שיאפשר לנו לגשת בזמן קבוע למיקום שלו בערימה.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 |
|  |  |  |  |  |  |  |

Id:3

Id:9

Id:7

Id:23

Id:23

Id:9

NW

NW

NW

NW

Id:23

Id:3

Id:9

Id:7

|  |  |
| --- | --- |
| מחלקה | תפקיד |
| Graph | כפי שהוגדרה בתרגיל - עבור יצירה של גרף. |
| Neighbor | יורשת מNode – על מנת להבדיל בין צומת אמיתית לבין צומת שמופיעה ברשימת השכנים (ואותה גם נוכל למחוק או להוסיף במידת הצורך מבלי לגרום לנזק לצומת האמיתי). מתחזקת את אותם השדות של Node ובנוסף מחזיקה מצביע לעבר הצומת שהוכנס ביחד איתה (כלומר הצד השני של הקשת שהוכנסה ביחד איתו) |
| Node | מחלקה המתארת את האובייקטים הראשיים בתוכנית – קודקודי הגרף. |
| doublyLinkedList | מימוש עבור רשימה דו כיוונית מקושרת. |
| dllNode | צומת אשר "עוטף" את הצומת המקורי (או השכן). מחזיק מצביע קדימה ואחורה ובנוסף ליבת האובייקט הוא הNode שהוא מחזיק. |
| MaxHeap | מימוש עבורה ערימת מקסימום |
| MyHashtable | מימוש עבור טבלת האש (בכל אתחול מחדש מגרילה את המשתנים a וb) |