תרגיל בית 2

מגישים:

נעה אנגלנדר 307875302

איתי קינן 305186900

שאלה 1-

א)

מתיאור הבעיה, הבנו ש-graph צריך לאפשר אכסון צמתים מטיפוס כלשהו, וכמו כן הוא צריך לאכסן (או לתעד) קשתות מכוונות בגרף.

על מנת ליצור גרף ולאכלס אותו בצמתים וקשתות, ראינו צורך ליצור:

* קונסטרקטור שמייצר גרף ריק
* מתודה AddNode להוספת צמתים לגרף
* מתודת AddEdge להוספת קשת לגרף.

לצורך בדיקות תקינות קלטים למתודות הנ"ל, ראינו צורך ב:

* מתודה NodeInGraph על מנת לאפשר לנו לבדוק בנוחות האם צומת כבר קיים בגרף.
* מתודה EdgeInGraph לאותה המטרה.

כמו כן, היות וראינו ב-TestDriver קראיה למתודות GetNodes ו- ListChildren, יצרנו גם מתודות כאלו, על מנת להציד צמתים הקיימים בגרף, ואת כל הבנים של צומת נתון.

ב)

בחרנו לשמור את הצמתים בגרף בתוך HashMap, כאשר ה-Key הוא צומת מטיפוס N כלשהו, וה-Value הוא HashSet שמכיל את כל הצמתים שהם בנים של אותו צומת.

בחרנו ב-HashMap כי הוא מאפשר לגשת לכל הבנים של צומת נתון בזמן קבוע, מה שיהיה יעיל עבור אלגוריתם ה-DFS בהמשך.

בחרנו לשמור את הבנים של כל צומת ב-HashSet מכיוון שכל צומת מופיע פעם אחת בדיוק בגרף.

מימוש אלטרנטיבי היה יכול להיות לשמור שני ArrayList: אחד שיכיל את כל המצתים בגרף, ואחד שיכיל את כל הקשתות. מימוש זה אינו יעיל עבוד DFS מכיוון שסיבוכיות הזמן שנדרשת על מנת לגשת לכל הבנים של צומת מסוים לאו דווקא תהיה קבועה.

ג)

ד) ב package של התרגיל מוגדר מפרט בשם Path. אם היינו עושים import java.nio.file.Path ואז מנסים לקרוא ל java.nio.file.Path בשם המקוצר Path , היה גורם ל-name conflict.

שאלה 2-

א)

האלגוריתם מתחיל מצביעת כל הצמתים בלבן, לסמן שעוד לא ביקר בהם.

לאחר מכן האלגוריתם מתחיל חיפוש בגרף מצומת נתון start.

1. הוא צובע את הצומת הנוכחי באפור, לסמן שהוא בבדיקה.

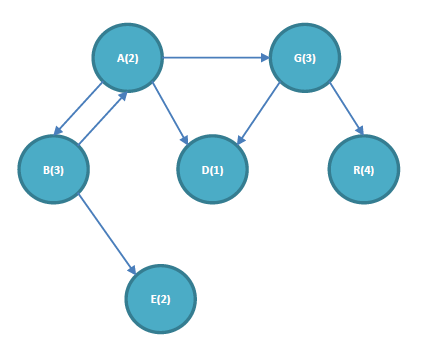
2. אם start = end סיימנו. נחזיר true.

3. מוסיפים את כל בניו של הצומת הנוכחי ל-priority queue.

4. בוחרים צומת בעל ה-priority הגבוה ביותר ב-queue, וכעת חוזרים לשלב 1 כאשר הצומת הנבחר הוא ה-start.

5. אם סיימנו לעבור על priority queue של בנים מבלי שהגענו ל-end, צובעים את ה-start של אותו queue בשחור (לסמן שסיימנו לבדוק אותו) וחוזרים אחורה ל-priority queue "עליון" יותר, שעוד לא ביקרנו את כל הצמתים בו.

ממשיכים באותו האופן עד שמגיעים ל-end, או עד שמסיימים לעבור על כל ה-priority queues. אם סיימנו לעבור על כולם מבלי שהגענו ל-end, נחזיר false.



DFS(A(2), D(1))

At First, all nodes are white, and none are active or visited. We begin from node A.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Black nodes | White nodes | Visited nodes (gray) | Current active node | Step |
|  | B, D, E, G, R | A | A | 1 |
|  | B, D, E, R | A, G | G | 2 |
|  | B, D, E | A, G, R | R | 3 |
| R | B, D, E | A, G | G | 4 |
| R | B, E | A, G, D | D | 5 |

DFS(B(3))

At First, all nodes are white, and none are active or visited. We begin from node B.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Black nodes | White nodes | Visited nodes (gray) | Current active node | Step |
|  | A, D, E, G, R | B | B | 1 |
|  | A, D, G, R | B, E | E | 2 |
| E | A, D, G, R | B | B | 3 |
| E | D, G, R | A, B | A | 4 |
| E | D, R | A, B, G | G | 5 |
| E | D | A, B, G, R | R | 6 |
| E, R | D | A, B, G | G | 7 |
| E, R, G |  | A, B, D | D | 8 |
| E, G, D, R |  | A, B | G | 9 |
| E, G, D, R |  | A, B | A | 10 |
| E, G, D, R, A |  | B | B | 11 |
| A, B, D, E, G, R |  |  |  | 12 |