אלגוריתמים, סמסטר ב' תשפ"ב
המכללה האקדמית של תל-אביב-יפו

פרוויקט תכנותי

תאריך הגשה: יום חמישי ה- 12.5.22 עד השעה 23:59.

הנחיות כלליות

1. הפרויקט ניתן להגשה בבודדים או בזוגות, אך לא בקבוצות גדולות יותר. מומלץ להגיש את הפרויקט בזוגות.
2. איחור בהגשה יאושר רק במקרה של מילואים, מחלה ממושכת או לידה, וגם זאת רק בתנאי שהפנייה למרצה בנושא נעשתה לפני מועד ההגשה המקורי של הפרויקט.

מטרת הפרויקט

עליכם לכתוב תכנית בשפת C++ (ובה בלבד. בחירת השפה אינה נתונה לשיקול דעת הסטודנטים), שפותרת את **בעיית מציאת עץ פורש מינימלי** בגרפים לא מכוונים פשוטים בכמה דרכים, לפי שני אלגוריתמים שלמדנו בקורס.

השלבים לביצוע הפרויקט

א. קראו תחילה היטב את ההנחיות של הפרויקט והבינו מה נדרש בדיוק.
ב. תכננו את ה-design של התוכנית שלכם: בחרו אילו מחלקות תממשו, החליטו על data members מתאימים ועל methods רלוונטיים לכל מחלקה.
ג. כתבו מימוש מלא לכל המחלקות שייעשה בהן שימוש במסגרת התוכנית.
ד. ממשו את האלגוריתמים הנדרשים ואת פונקציית ה-main של התוכנית.

תיאור מפורט של מטרת הפרויקט:

מטרת הפרויקט היא לממש מספר פתרונות ל**בעיית מציאת עץ פורש מינימלי**, לחקור ולנתח את ההבדלים בין הפתרונות השונים.

העפ"מים יחושבו על ידי שני אלגוריתמים מרכזיים:

1. אלגוריתם קרוסקל כאשר המיון ממומש על ידי אלגוריתם quick-sort וקבוצות זרות ממומשות בעזרת עצים.

2. אלגוריתם פרים כאשר תור העדיפויות ממומש על ידי ערימה בינארית.

3. בהמשך, המשתמש יזין קשת אחת שאותה יש להסיר מהגרף, ובמידה והקשת שייכת לעפ"מ שמצאתם בעזרת קרוסקל, יש למצוא עפ"מ חדש שלא יכיל אותה.

<u>תיאור אלגוריתם למציאת עפ"מ לאחר שהוסרה קשת</u>

- במידה והקשת הייתה גשר בגרף המקורי, אז לאחר הסרתה הגרף כבר לא קשיר. יש לבדוק זאת ולהוציא הודעה No MST.
- במידה והקשת אינה גשר, ניתן להריץ שוב את האלגוריתם של קרוסקל כאשר הקשתות כבר ממוינות ולכן אין צורך למיין אותן מחדש.

<u>תיאור התכנית הראשית :</u>

פונקציית main של התכנית תקבל שני ארגומנטים שהם שמות קבצים ותבצע:

1. קליטת נתוני הגרף מקובץ טקסט ששמו נמצא בארגומנט הראשון ([1]argv) ויכיל את הקלט הבא:
   1.1. מספר שלם, n, שמסמן את מספר הקדקודים בגרף.
   1.2. מספר שלם, m, שמסמן את מספר הצלעות בגרף.
   1.3. שלשות של מספרים שלמים, כל שלשה בשורה נפרדת, כאשר השלשות מייצגות שמות של שני קדקודים i,j (כל קדקוד מיוצג על ידי מספר בין 1 ל- n) ואת משקל הקשת מ- i ל- j.
   משקל הקשת אמור להיות מספר שלם (לא בהכרח חיובי).
   1.4. זוג i,j שמסמל את הקשת שיש להוריד מהגרף בשביל ההרצה השלישית.

בעזרת נתונים אלה התוכנית תבנה גרף ותריץ את האלגוריתמים המתוארים למעלה (סה"כ 3 הרצות).

2. תדפיס למסך את משקל העץ בכל אחד מהמקרים.

פורמט הפלט יהיה (בהנחה שמשקל העפ"מ הראשון הוא <x> ומשקל העפ"מ לאחר הסרת הקשת הוא <x'>):

```
Kruskal <x>
Prim <x>
Kruskal <x'>
```

<u>כך למשל קלט תקין ייראה כך:</u>

```
6
9
1 2 16
1 3 13
2 3 10
2 4 12
4 3 9
3 5 14
5 4 7
5 6 4
4 6 20
1 3
```

<u>שימו לב – הקשת (1,3) היא הקשת שמסירים מהגרף</u>

3. תדפיס לקובץ שהוא הארגומנט השני שהתוכנית קיבלה את אותו הפלט שהודפס למסך.

<u>במקרה זה הפלט יהיה:</u>

```
Kruskal 43
Prim 43
Kruskal2 46
```

<u>מבני הנתונים</u>

לצורך ביצוע הפרויקט, הנכם <u>נדרשים</u> לממש את המחלקות הבאות ללא שימוש ב- STL:

1. **גרף מכוון פשוט עם משקלים שממומש על ידי רשימות שכנויות:**
הגרף ימומש כרשימות שכנויות, כאשר כל קדקוד ברשימה יכיל בנוסף למצביע לקדקוד הבא ברשימה גם שדה של משקל הקשת המתאימה.

<u>פעולות בסיסיות:</u>
MakeEmptyGraph(n) – יצירת גרף ריק מקשתות עם n קדקודים.
IsAdjacent(u,v) – מחזיר כן אם הקשת (u,v) שייכת לגרף, ואחרת לא.
GetAdjList(u) – החזרת רשימה מקושרת של השכנים של קדקוד u.
AddEdge(u,v,c) – הוספת קשת (u,v) בעלת משקל c.
RemoveEdge(u,v) – הסרת הקשת (u,v) מהגרף.

2. **תור קדימויות מינימום שממומש על ידי ערימת מינימום.**
תור הקדימויות ימומש בעזרת ערימת מינימום בינארית שתישמר במערך, כפי שנלמד בקורס מבני נתונים.
כל איבר בתור יהיה זוג מהצורה (data, key) כאשר העדיפות נקבעת על פי ה-key.

<u>פעולות בסיסיות:</u>
Build – בניית ערימה מתוך מערך על ידי שימוש באלגוריתם של פלויד.
DeleteMax – שליפת האיבר עם המפתח המקסימלי מהערימה ותיקון הערימה.
IsEmpty – בדיקה האם הערימה ריקה.
DecreaseKey(place, newKey) – הקטנת ערך המפתח (עדיפות) של האיבר שנמצא במקום place בערימה לערך newKey (וכמובן תיקון הערימה לפי הצורך).

3. **קבוצות זרות (מבנה union find) שממומש על ידי עצים עם מצביע להורה, שמאוחסן במערך.**
יש לממש את השיפורים של union by size ושל כיווץ מסלולים שנלמדו בקורס מבני נתונים.

עליכם לממש לכל אחד מטיפוסי הנתונים המופשטים האלה את הפעולות הבסיסיות המפורטות לעיל. מעבר לכך אתם יכולים להוסיף פעולות ו- data members נוספים, כראות עיניכם.

כמו כן עליכם לממש כמובן כל מחלקה אחרת שלה תזדקקו במהלך התכנית.

<u>בדיקת שגיאות</u>

הקפידו לבדוק שגיאות אפשריות בקלט (למשל לבדוק שקשת מחברת בין קדקודים קיימים, שהמשקלים אכן שלמים, וכך הלאה). במקרה של שגיאה, כתבו הודעת שגיאה למסך invalid input וצאו מהתכנית באמצעות הפונקציה exit(1).

(יש לבצע #include <stdlib.h> על מנת להשתמש בה).

מותר להניח שיתקבל בקלט גרף פשוט ואין צורך לבדוק זאת.

<u>הנחיות הגשה</u>

יש להגיש במערכת mama במקום המיועד להגשה את הקבצים הבאים:

1. קובץ readme שיכיל את כל פרטי ההגשה הבאים:

כותרת – <u>פרויקט בקורס אלגוריתם.</u>

שורה מתחת - שמות המגישים, מספרי ת.ז. שלהם ומספר הקבוצה של כל אחד מהם (מותר להגיש עם בן זוג מקבוצה אחרת).

שימו לב: קובץ טקסט פשוט – לא word.

2. את **<u>תיקיית</u>** הsolution שבתוכו נמצא הפרויקט שכתבתם, אחרי שהרצתם את הפרויקט על מחשב אחר ובדקתם שהכול מתפקד בדיוק כמו שאתם רוצים. ניתן ורצוי לכווץ את התיקייה הנ"ל.
3. **רק אחד מבני הזוג יגיש את הפרויקט**.

<u>שימו לב!</u> הגשה שאינה בפורמט הנדרש תידחה אוטומטית.

<u>הערות הכרחיות נוספות</u> (שימו לב, למרות שזה קורס באלגוריתמים התכנית צריכה להיות כתובה לפי כל הכללים של תכנות נכון!)

- הקפידו על תיעוד, שמות בעלי משמעות למשתנים, מודולאריות וכל מה שנדרש מתכנות נכון. בתיעוד בראש התוכנית כתבו גם הוראות הפעלה <u>מדויקות</u> <u>וברורות</u>.
- הקפידו על חלוקה נכונה לקבצים (קבצי cpp וקבצי h לכל מחלקה).
- תכנתו Object Oriented והימנעו מאלמנטים מיותרים של קוד פרוצדוראלי.
- הקפידו לשחרר את כל הזיכרון אשר הקציתם דינמית לאחר שהשתמשתם בו.
- במקרה של תקלה בריצת התוכנית (מסיבה כלשהי), עליה לדווח זאת למשתמש <u>טרם</u> סיימה לרוץ.
- בדקו את תכניותיכם על קלטים רבים ככל האפשר - כולל קלטים חוקיים ולא חוקיים, וזאת בנוסף לקלטים לדוגמה שהוכנו עבורכם במאמא.
- תנאי הכרחי (אך כמובן לא מספיק) לקבלת ציון עובר על הפרויקט, הוא שהקוד יעבור **<u>קומפילציה בוויז'ואל סטודיו 2019</u>**. ציונו של פרויקט אשר אינו עובר קומפילציה יהיה 0.

צוות אלגוריתמים, סמסטר ב' תשפ"ב.