תכנות מערכות 2 –README מטלה 1

תז: 323858324 shannya08@gmail.com

אנחנו נדרשנו לממש את הקבצים הנל:

Graph.cpp,Algorithms.cpp,Test.cpp,Makefile,Algoritms.hpp,Graph.hpp,demo.cpp

בקובץ **Test.cpp** הוספתי עוד טסטים וגם בקובץ **Demo.cpp** הוספתי עוד דוגמאות הרצה.

Graph.hpp:

הקובץ מכיל מחלקה המייצגת גרף כאשר גרף מיוצג על ידי מטריצת שכנויות כך שאם יש מספר שונה מ0 בתא **[i,j]** אז זה אומר שיש צלע בין הקודקוד **i** ל **j** ואם יש 0 אז זה אומר שאין צלע בין הקודקודים.

המחלקה מכילה פונקציה פומבית בשם **loadGraph** ,פונקציה פומבית נוספת בשם **printGraph** ,פונקציה פומבית נוספת **size** מחזירה את מספר הקודקודים בגרף.

,ופונקציה פרייבט בשם **validateGraph** המבצעת בדיקה על תקינות הגרף, היא מוודאת שהמטריצה ריבועית ומחזירה במקרה שלא חריגה של **invalid_argument**.

אנחנו נעבוד במטלה שלנו עם גרפים מכוונים כלומר אם יש לנו צלע מקודקוד **i** לקודקוד **j** בגרף לא מכוון אנחנו נתייחס אליה כאל שתי צלעות כי מתייחסים לגרף כמכוון.

Graph.cpp:

הקובץ מכיל מימוש של הפונקציות loadGraph המקבלת מטריצת שכנויות ומעתיקה אותה לתוך המבנה הנתון של הגרף. לאחר העתקת הגרף החדש, הפונקציה מפעילה פונקצית validateGraph על מנת לוודא את תקינות הגרף.

printGraph: פונקציה זו מדפיסה את הגרף הנוכחי. היא סופרת את מספר הקשתות בגרף ומדפיסה אותו, יחד עם מספר הצמתים.**" Graph with n vertices and m edges."**

validateGraph: פונקציה זו בודקת את תקינות הגרף-מטריצת שכנויות. היא בודקת שכל שורה במטריצת השכנויות בגודל שווה למספר הצמתים בגרף כלומר שהמטריצה ריבועית=חוקית

Algorithms.hpp:

זהו קובץ שבו ישנם כל ההצהרות של הפונקציות שממומשות **Algorithms.cpp .**

Algorithms.cpp:

זהו הקובץ שבו ישנם את כל המימושים של הפונקציות שנדרשנו שממומשות לפי האלגוריתמים שלמדנו באלגוריתמים1

isConnected(g) - פונקציה זו בודקת אם הגרף הינו קשיר, באמצעות אלגוריתם BFS

shortestPath(g,src ,dest) - פונקציה זו מחשבת את המסלול הקצר ביותר בין צומת מקור (src) לצומת יעד (dest) בגרף, באמצעות אלגוריתם Bellman-Ford.

isContainsCycle(g) – פונקציה זו בודקת האם קיימים מעגלים בגרף על ידי בדיקה של כל צומת בגרף ומפעילה את הפונקציה isContainsCycleUtil על כל צומת, הפונק הזו מבצעת DFS כדי לזהות האם קיים בו מעגל.

isBipartite(g) - פונקציה זו בודקת אם הגרף הינו דו-צדדי, כלומר ניתן לצבוע את כל הצמתים של הגרף בשני צבעים כך שכל צומת יהיה מקושר לצמתים בצבע שונה ממנו

negativeCycle(g) - הפונקציה negativeCycle בודקת האם קיים מעגל שלילי בגרף, מעגל שלילי הוא מסלול בגרף שסכום המשקלים של הצלעות בו שלילי.

והיא עושה זאת על ידי שימוש בפונקציה shortestPath , היא עוברת על כל הצמתים בגרף, ומפעילה את shortestPath עם כל צומת כצומת המקור וכצומת היעד. אם shortestPath מחזירה את המחרוזת "**Negative cycle detected in the path**", זה אומר שמצאה מעגל שלילי.

אם אף אחד מהקריאות ל shortestPath לא מחזירה " **Negative cycle detected in the path"** היא מחזירה את המחרוזת " **The graph does not contain negative cycle**", שאומרת שאין מעגלים שליליים בגרף.

אופן ההרצה: כדי להריץ נכתוב את הפקודה בטרמינל make שיריץ את הקובץ demo.cpp כפי שכתוב בmakefile. ואז כדי להריץ את הטסטים נכתוב make test-> ./test.

אם נרצה למחוק את הקבצים שנוצרו נבצע make clean.