גרפים ואלגוריתמים

פרויקט זה הוא עיצוב פשוט של מבנה נתונים של גרף ב־C++, יחד עם אלגוריתמים בסיסיים לגרפים. הפרויקט תומך בגרפים כמו גם בגרפים מכוונים.

תכונות

יצירת אובייקט גרף וציון האם הוא גרף מכוון או לא.

טעינת גרף ממטריצת שכנויות דו־ממדית.

מטריצת הסמיכות חייבת להיות מרובעת (מספר השורות שווה למספר העמודות).

עבור גרפים לא מכוונים, מטריצת הסמיכות חייבת להיות סימטרית.

מחלקת גרף

Graph.cpp

פונקציות המחלקה:

loadGraph: טוענת גרף ממטריצת שכנויות.

printGraph: מדפיסה את מספר הקודקודים והקשתות בגרף.

size: מחזירה את מספר הקודקודים של הגרף.

getMatrix: מחזירה את מטריצת השכנויות של הגרף כווקטור דו מימדי.

isDirected: מחזירה האם הגרף הוא גרף מכוון.

קובץ Algorithms.cpp

מכיל מימושים לפונקציות שונות על גרפים.

הפונקציות:

isConnected:

בודקת אם הגרף מחובר.

אופן המימוש:

הפונקציה משתמשת בdfs ומבצעת חיפוש עומק מתוך צומת התחלה נתון.

הפונקציה dfs מסמנת צמתים כמבוקרים במהלך החיפוש.

הפונקציה isConnected מבצעת חיפוש dfs מכל צומת בגרף ובודקת לכל צומק אם כל השכנים בוקרו. אם לא כולם בוקרו הגרף אינו קשיר.

shortestPath:

מוצא את הנתיב הקצר ביותר בין שני קודקודים בהתבסס על סוג הגרף:

בלמן־פורד לגרפים עם ממושקלים.

BFS לגרפים לא משוקללים.

הפונקציה משתמשת בפונקציה **PrintShortestPath** כדי להדפיס את המסלול הקצר.

isBipartite:

פונקציה זו בודקת אם גרף נתון הוא דו-צדדי. גרף דו-צדדי הוא גרף שניתן לחלק לשתי קבוצות זרות כך שאין קשת בין שני קודקודים באותה קבוצה.

הפונקציה מקבלת את מטריצת השכנות של הגרף.

מאתחלת וקטור color עם הערך 1- כדי לסמן שקודקודים לא צבועים עדיין.

משתמשת בשני וקטורים, A ו-B, כדי לאחסן את הקודקודים של כל קבוצה.

שימוש בחיפוש לרוחב (BFS):

עבור כל קודקוד, אם הוא לא צבוע, צובעים אותו בצבע 0 ומתחילים חיפוש לרוחב מהקודקוד הזה.

במהלך ה-BFS, צובעים את הקודקודים הסמוכים בצבע הנגדי. אם נמצאת קשת שבה שני הקודקודים צבועים באותו צבע, הפונקציה מחזירה "-1", מה שמעיד שהגרף אינו דו-צדדי.

isContainsCycle:

הפונקציה בודקת אם גרף מכיל מעגל ולהחזיר את המעגל במידה וקיים.

dfs – פונקציית עזר: מבצעת חיפוש לעומק בגרף. אם נמצא מעגל, הפונקציה מחזירה את המסלול של המעגל בפורמט טקסטואלי.

isContainsCycle מפעילה את הפונקציה dfs על כל צומת בגרף כדי לבדוק אם קיים מעגל.

מהלך האלגוריתם:

הפונקצייה מגדירה מערך צבעים לבדיקת מצב כל צומת (לא ביקרנו, בביקור, ביקרנו).בנוסף מגדירה מערך הורים לעקוב אחרי הצמתים במהלך החיפוש.

לאחר מכן נבצע חיפוש לעומק לכל צומת שלא ביקרנו בו.

אם במהלך החיפוש נמצא מעגל, הפונקציה מחזירה את המעגל בפורמט טקסטואלי.

negativeCycle:

הקוד מממש את אלגוריתם בלמן-פורד למציאת מעגל שלילי בגרף מכוון ממושקל.

האלגוריתם מבצע רלקסציה לכל הקשתות בגרף V פעמים, כאשר V הוא מספר הקודקודים.

במהלך כל איטרציה, אם נמצא מסלול קצר יותר, האלגוריתם מעדכן את המרחק ואת ההורה של הקודקוד הנוכחי.

בסיום האיטרציות, אם עדיין מתבצעת רלקסציה, זה מצביע על קיומו של מעגל שלילי.

אם התבצעה רלקסציה באיטרציה האחרונה, האלגוריתם מוצא את הקודקוד המעורב במעגל השלילי על ידי מעקב אחורה דרך מערך ההורים.

האלגוריתם בונה את המסלול של המעגל השלילי ומחזיר אותו כמחרוזת.

קבצים נוספים

מצורף קובץ demo המכיל main המדגים שימוש בפונקציות

בנוסף מצורף קובץ טסטים הבודק מקרי קצה של כלל הפונקציותץ

מצורף גם קובץ makefile. על מנת להריץ את התוכנית ואת קובץ הדמו יש לכתוב בטרמינל make run

על מנת להריץ ולהדוק את הטסטים יש להריץ make test