מטלה - עץ בינארי

מטרת המטלה היא לאמן אתכם במחלקות, בניה, פירוק, ניהול זיכרון, מבני-נתונים ובדיקות-יחידה.

המשימה: כיתבו מחלקה בשם Tree, המייצגת עץ-חיפוש בינארי לשמירת מספרים בסדר עולה, כפי שלמדתם בקורס מבני נתונים (לא חייב להיות עץ מאוזן).

פעולות

המחלקה צריכה לאפשר את הפעולות הבאות:

- insert - מקבלת כקלט מספר i ומכניסה את i לעץ במקום המתאים.
- remove - מקבלת כקלט מספר i ומסירה את i מהעץ.
- size - מחזירה את כמות המספרים בעץ.
- contains - מקבלת כקלט מספר i ומחזירה "אמת" אם ורק אם i נמצא בעץ.
- root - מחזירה את המספר הנמצא בשורש העץ.
- parent - מקבלת כקלט מספר i ומחזירה את המספר שנמצא מעל i בעץ.
- left - מקבלת כקלט מספר i ומחזירה את המספר שהוא הבן השמאלי של i בעץ.
- right - מקבלת כקלט מספר i ומחזירה את המספר שהוא הבן הימני של i בעץ.
- print - הדפסת מבנה העץ, לצורך הצגה וניפוי שגיאות. פורמט ההדפסה - לשיקולכם.

יש לזרוק חריגות במקרים הבאים:

- insert - כשמנסים להכניס מספר שכבר נמצא בעץ.
- remove - כשמנסים למחוק מספר שלא נמצא בעץ.
- בשאר הפקודות - לזרוק חריגה בכל מצב שנראה לכם חריג – תפעילו שיקול דעת.

בנוסף לפתרון עצמו, עליכם לכתוב קובץ בשם **TreeTest.cpp** הכולל בדיקות-יחידה (unit-test) מפורטות.

קבצים

מצורפים לתרגיל זה הקבצים:

- TreeDemo.cpp - תוכנית ראשית לדוגמה.
- TreeTest.cpp - תוכנית ראשית הכוללת בדיקות-יחידה לדוגמה.
- Makefile – קובץ ליצירת תוכנית הדוגמה ותוכנית הבדיקה.

שלבי העבודה

בשלב ראשון, עליכם לכתוב את הקבצים הדרושים על-מנת שהפקודות הבאות ירוצו בלי שגיאות קימפול:

```
make demo && ./demo
make test && ./test
```

בשלב זה אין לשנות את הקבצים הנתונים – עליכם לוודא שהתוכנית שלכם עובדת עם הקבצים הנתונים כמו שהם. כמו כן, לא חייבים לכתוב תוכנית המקבלת 100 בכל הבדיקות – רק שתתקמפל בלי שגיאות.

בשלב שני, יש להרחיב את הקובץ TreeTest.cpp ולהוסיף בדיקות-יחידה נוספות באותו סגנון של הבדיקות הקיימות (לא למחוק את הקיימות). יש לכתוב בדיקות-יחידה מפורטות. שימו לב – בשלב זה הקוד שכתבתם כנראה לא יעבור את כל הבדיקות – זה בסדר. העיקר שהבדיקות שלכם יהיו מלאות.

יש להגיש תוך שבוע (במודל ובתירגול הרביעי) את הקוד במצב זה – קוד Tree שמתקמפל, וקובץ TreeTest.cpp הכולל בדיקות-יחידה מפורטות, שעדיין לא כולן עוברות.

בשלב שלישי, יש לשפר את מימוש המחלקה Tree שלכם כך שתעבור את כל הבדיקות – גם הבדיקות שלכם וגם הבדיקות האוטומטיות שלנו.

יש להגיש תוך שבוע נוסף (במודל ובתירגול החמישי) את הקוד המלא.

בתירגולים תתבקשו להציג תוכנית-דוגמה היוצרת עץ בינארי ומדפיסה אותו (בעזרת print שכתבתם). בנוסף תתבקשו להסביר על הקוד שלכם ולהראות שהוא מקיים כללים בסיסיים של הנדסת תוכנה: חלוקה לקבצים, תיעוד, שמות משמעותיים למשתנים, בדיקות תקינות ומניעת דליפת-זיכרון.

הגשה לבדיקה אוטומטית

צרו מאגר (repository) חדש בגיטהאב והעלו לשם את הקבצים בתיקיה הראשית.

הגישו בטופס-ההגשה קישור-שיבוט למאגר - הקישור שרואים כשלוחצים על הכפתור clone בגיטהאב.

אנחנו נבצע את הפקודות הבאות ממחשב עם לינוקס:

1. git clone <הקישור שלכם>
2. נעתיק לתוך התיקיה שלכם תוכנית TreeTest.cpp משלנו, עם בדיקות אוטומטיות נוספות.
3. make test && ./test

אתם יכולים לפתור את התרגיל בכל סביבת-פיתוח שאתם רוצים, אבל לפני ההגשה, וודאו שהפקודות האלו רצות בלי שגיאות על מחשב לינוקס אחר כלשהו.

דגשים

- יש לחזור על החומר של ההרצאות לפני שמתחילים לכתוב, ולהשתמש בו לפי הצורך.
- מותר להשתמש בתכונות מתקדמות של שפת C++ גם אם עדיין לא נלמדו בהרצאות.
- אין להעתיק תרגילים שלמים מסטודנטים אחרים. מותר להיעזר בקטעי קוד מהאינטרנט, אולם **יש לציין בבירור את המקור**, לוודא שהקוד עובד, ולוודא שאתם מבינים למה הוא עובד.