מטלה - מספרים פיסיקליים, העמסת אופרטורים

בשנת 1999, לוויין של נאס"א בשווי של 125 מיליון דולר התרסק בגלל אי-התאמה ביחידות - אחד הצוותים שפיתחו את הלוויין עבד ביחידות מטריות והצוות השני עבד ביחידות בריטיות (ראו דוגמאות נוספות כאן: http://mentalfloss.com/article/25845/quick-6-six-unit-conversion-disasters).

כדי שזה לא יקרה שוב, הם שכרו אתכם וביקשו מכם לכתוב מחלקה המייצגת מספר עם יחידות. במחלקה הזאת אפשר, למשל, לייצג את המספר "3 מטר" ואת המספר "40 סנטימטר", והסכום שלהם לא יהיה 43 אלא 3.4 מטר - המחלקה תדאג לבצע את ההמרה המתאימה. בנוסף, המחלקה לא תאפשר לחבר מספרים עם מימדים לא תואמים, למשל, חיבור של "3 מטר" עם "5 שניות" יגרום לזריקת חריגה.

לצורך המטלה, נתמקד בשלושת הממדים של הפיסיקה הניוטונית: אורך, זמן, ומסה, וביחידות הבאות:

- אורך - מטר (m) קילומטר (km) סנטימטר (cm).
- זמן - שניה (sec) דקה (min) שעה (hour).
- מסה - גרם (g) קילוגרם (kg) טון (ton).

הגדירו מחלקה בשם PhysicalNumber עם הפעולות הבאות (ראו בקובץ המצורף PhysicalNumberDemo.cpp):

- שישה אופרטורים חשבוניים: חיבור (+) הוספה (+=) פלוס אונרי (+), ושלושת האופרטורים המקבילים לחיסור (-). כאמור, חיבור של שני מספרים מאותו מימד יתבצע תוך המרת היחידה של המספר השני ליחידה של המספר הראשון; חיבור של שני מספרים ממימדים שונים יגרום לחריגה.
- שישה אופרטורי השוואה: גדול, גדול-או-שווה, קטן, קטן-או-שווה, שווה, לא-שווה, לפי אותם כללים של האופרטורים החשבוניים.
- הגדלה ב-1 (++) והקטנה ב-1 (--).
- אופרטור קלט (>>) ואופרטור פלט (<<).

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

קבצים

מצורפים לתרגיל זה הקבצים:

- PhysicalNumberDemo.cpp - תוכנית ראשית לדוגמה.
- PhysicalNumberTest.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.

לאחר מכן, יש להרחיב את הקובץ PhysicalNumberTest.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 (במודל ובבדקן האוטומטי) את הקוד במצב זה – קוד שמתקמפל, וקובץ PhysicalNumberTest.cpp הכולל בדיקות-יחידה מפורטות, שעדיין לא כולן עוברות.

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

יש להגיש תוך שבוע נוסף (במודל ובבדקן האוטומטי) את הקוד המלא.

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 PhysicalNumberTest.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.

הרחבות ושאלות

[למחשבה בלבד - לא להגשה]

ניתן להרחיב את המטלה בכמה דרכים:

1. הוספת **אופרטור כפל ואופרטור חילוק**. שימו לב - האופרטורים האלה מתנהגים בצורה שונה מחיבור וחיסור. למשל, 10 קילומטר (אורך) חלקי 2 שעות (זמן) זה 5 קמ"ש (מהירות). איך לדעתכם כדאי לייצג את הצירופים השונים של יחידות כך שיהיה אפשר, למשל, לחשב "10 קמ"ש + 5 מטר לשניה"?
2. הוספת מימד נוסף: **כסף**. לשם כך יש להמיר בין מטבעות שונים לפי השער היציג הנוכחי שלהם, שאפשר למצוא באינטרנט. איך לדעתכם אפשר לקרוא את השער היציג הנוכחי ולהשתמש בו בחישובים?