תרגיל 5

<u>הנחיות חשובות:</u>

- בתרגיל זה מותר לכם להשתמש בקבצים שלכם בספריות הסטנדרטיות הבאות בלבד:

```
#include <cassert>
#include <vector>
#include <cstdlib>
#include <cmath>
```

- בתרגיל זה אסור להשתמש ב new ו- delete או בהקצאה דינמית אחרת באופן ישיר.
- שימו לב שלקבצים מצורפים קבצי קוד וההוראות בקבצים הללו מחייבות!

<u>הנחיות חשובות לכלל התרגילים מעתה והלאה בקורס:</u>

- התרגילים הם לעבודה ביחידים. **מותר להתייעץ אך ורק בעל פה, אסור בתכלית האיסור שחומר כתוב/מודפס/אלקטרוני יעבור בין אנשים. בנוסף, על חלק מהתרגילים תיבחנו פרונטלית ועליכם להבין כל דבר בקוד!**
- המנעו ממספרי קסם: מספרים שמופיעים באמצע הקוד בלי משמעות מיוחדת (לדוגמא נניח שמספר הרשומות בתרגיל אחר הוא מקסימום 50 ואז בכל מקום בקוד כתוב 50. לעומת זאת, 0 לתחילת מערך לא נחשב מספר קסם - הפעילו הגיון בריא) והשתמשו במקום זאת בפקודות מאקרו (#define או אם כבר למדתם על כך ב const).
- אין להשתמש ב-variable length arrays , וכן במשתנים סטאטיים ו / או גלובליים.כל התרגילים בקורס צריכים להתקמפל ולרוץ באתר c9.io (האתר מריץ מערכת הפעלה אובונטו) עם שורת הקמפול:

```
gcc -Wall -Wvla -Werror -g …
```

ושורת הקמפול:

```
g++ -Wall -Wvla -Werror -g -D_GLIBCXX_DEBUG -std=c++11 ...
```

עבור תרגילי C++
או במידה ומצורף Makefile עם ה Makefile המצורף

- יש להקפיד על סגנון תכנות טוב כמו שלמדתם. לדוגמא, להימנע מחזרות קוד (לכתוב פונקציות שצריך), שמות משתנים עם משמעות, בהירות הקוד, להקפיד להשתמש בקבועים שצריך ולא במספרי קסם וכו'.

- עליכם להגיש קובץ ששמו מספר ת.ז. שלכם כמו שהיא מופיעה באתר המודל נקודה zip. לדוגמא, אם מספר ת.ז. שלי הוא 12345678 אז שם הקובץ יהיה:
  12345678.zip
- הקפידו להשתמש בפונקציות של C++ (למשל new, delete, cout) על פני פונקציות של C (למשל malloc, free, printf). בפרט השתמשו במחלקה string ולא במחרוזת של C (כלומר, *char).
- יש להשתמש בספריות סטנדרטיות של C++ ולא של C אלא אם כן זה הכרחי.
- הקפידו על עקרונות Information hiding. לדוגמא, הקפידו להגדיר משתני מחלקות כ private.
- הקפידו לא להעביר אובייקטים גדולים by value אלא by reference או by const reference.
- הקפידו מאד על const correctness. כלומר, על שימוש במילה השמורה const בצורה נכונה.

<u>תרגיל:</u>

משימת תכנות – מטריצה גנרית:
בתרגיל זה תממשו מחלקה גנרית של מטריצה, כלומר איברי המטריצה הם מטיפוס גנרי.
המחלקה תוכל גם לשמש כמטריצה של כל טיפוס שהוא כפי ש std::vector ו std::list הגנריים יכולים להכיל איברים מכל טיפוס שהוא (שמייצגים מספרים עם פעולות חשבון מוגדרות מראש).
המחלקה תדע לבצע פעולות חישוב של מטריצות.
עליכם לכתוב את הקובץ Matrix.hpp שיכיל את ההצהרה והמימוש של המחלקה הגנרית Matrix.

הטיפוסים הספציפיים לאיברי המטריצה:
ניתן ליצור מטריצה מכל טיפוס אשר יש לו מימוש לשלושת האופרטורים = ,* ,+ וכן
אם הטיפוס הוא מחלקה אז יש מימוש של בנאי ברירת מחדל היוצר את איבר האפס.
אם הטיפוס הוא פרימיטיבי אז אם שם הטיפוס הוא לדוגמא int אז ()int הוא איבר האפס מטיפוס int (שימו לב שזה לא בנאי ברירת מחדל).
כמובן שטיפוסים פרימיטיבים כדוגמת int עומדים בכל הקריטריונים הללו.
לכל מחלקה יש דרך שונה לחישוב פעולות החשבון, לייצוג כמחרוזת, ואיבר אפס משלה.
בנוסף יכולות להיות פונקציות ייחודיות לכל מחלקה.
עליכם לממש את המחלקה Rational בקובץ Rational.hpp לפי הממשק הנתון בו.
זוהי מחלקת מספרים רציונליים אשר תוכלו לבדוק בעזרתה את המימוש שלכם למטריצה הגנרית. טיפוס זה מייצג מספר רציונלי באמצעות שני מספרים שלמים מטיפוס long int, המונה והמכנה של השבר. נדרוש שלכל מספר רציונאלי יהיה ייצוג יחיד קאנוני (מספר

רציונאלי צריך להיות תמיד מיוצג בצורתו הכי מצומצמת שאפשר המכנה תמיד יהיה שלם וחיובי), לכן ישנם כמה כללים שמפורטים בקובץ ה-header, שעליכם למלא אותם.
הייצוג המחרוזתי של מספר רציונלי (גם עבור קריאה ממחרוזת בבנאי שמקבל מחרוזת וגם עבור הפונקציה שמחזירה ייצוג מחרוזת) הוא "numerator/denominator", זאת אומרת, עם קו שבר נטוי, ללא רווחים וללא הגרשיים.

ממשק המטריצה:
עליכם להגדיר ולממש את המחלקה הגנרית Matrix בקובץ Matrix.hpp.
המטריצה תהיה גנרית ואבריה לא יהיה בהכרח מספרים שלמים אלא מספרים מטיפוס גנרי.
לצורך נאותות, אתם רשאים להניח שהטיפוסים יהיו בעלי מימוש לאופרטורים הנדרשים כנמצא לעיל. כמו כן אתם יכולים להניח שהטיפוסים שיהיו בשימוש לאברי המטריצה הם כאלה שסדר הפעולות האטומיות של חיבור או כפל (בשרשרת פעולות חישוב ארוכה) לא משנה.
לדוגמא: (a + b) + c == a + (b + c)
בין הפונקציות שיש לממש או שאפשר להשתמש בברירת המחדל שהשפה נותנת (הכל באופן גנרי, כמובן):
• בנאי חסר פרמטרים. הבנאי בונה מטריצה של 1 על 1 עם איבר שערכו הוא ()T כאשר T הוא הטיפוס שמאוחסן במטריצה.
• בנאי שמקבל את מספר השורות, מספר העמודות ווקטור עם ערכי המטריצה למילוי (ראו את הקריאה לבנאי הזה מהטסטר).
• הורס (אין לממש, יש לבנות את המחלקה כך שה default הוא טוב מספיק).
• בנאי העתקה (copy constructor) המקבל מטריצה אחרת (אין לממש, יש לבנות את המחלקה כך שה default הוא טוב מספיק).
• אופרטור השמה (אין לממש, יש לבנות את המחלקה כך שה default הוא טוב מספיק).
• +operator לחיבור מטריצות.
• *operator לכפל מטריצות. המטריצה של האובייקט עליו מפעילים את הפונקציה היא המטריצה השמאלית בכפל.
• *operator לכפל המטריצה בסקלר, שהסקלר מצד ימין (כל איבר במטריצה מוכפל בסקלר).
• פונקצית שחלוף מטריצה (transpose).
• פונקצית עקבה (hasTrace) המקבלת reference לאיבר מהטיפוס הגנרי ומשימה בו את ערך העקבה (אם קיים) של המטריצה ומחזירה ערך בוליאני: true אם המטריצה ריבועית, וfalse אחרת ובמקרה כזה תשים את ערך איבר האפס באיבר שהתקבל כ reference.
• פונקציה getColNum אשר מחזירה את מספר הטורים.
• פונקציה getRowNum אשר מחזירה את מספר השורות.
• פונקציה isSquareMatrix אשר מחזירה true אם ורק אם המטריצה ריבועית (מספר שורות שווה למספר עמודות).

• אופרטור == אשר מחזיר true אם ורק אם המטריצות שוות (גודלן שווה וכל האיברים שווה).
• אופרטור =! אשר מחזיר true אם ורק אם המטריצות שונות (גודלן לא שווה או קיים לפחות איבר אחד שונה).
• 10 הפונקציות הנ"ל (חיבור, כפל במטריצה, כפל בסקלר,transpose, hasTrace, getRowNum, getColNum, אופרטור ==, אופרטור =!) לא משנות את האובייקט עליו הופעלה הפונקציה .

בנוסף תוכלו להוסיף עוד פונקציות ציבוריות או פרטיות כרצונכם, לפי מה שנראה לכם שימושי למחלקה.
בתרגיל זה הממשק (החתימות של הפונקציות) מוכתב לכם חלקית ע"י חלק מקריאות לפונקציות הללו מהטסט, אבל הוא לא מוכתב לגמרי ועדיין יש כמה החלטות שעליכם לעשות. חישבו למשל על:
• לכל פונקציה אם היא צריכה להיות מוגדרת כ const (לא משנה את האובייקט עליו היא מופעלת).
• לכל ארגומנט אם הוא צריך להיות מועבר by reference או by const reference או מועתק, או אולי כדאי להשתמש במצביע.
• לערך ההחזרה, האם הוא צריך להיות מוגדר כ const, והאם הוא מועבר by reference
או מועתק.
שימו לב: חלק מהמטודות לא דורשות מימוש מכיוון שמימוש ברירת המחדל של השפה הוא טוב מספיק. נקודות יורדו על מימוש של מטודות שאין צורך לממש!

טסטרים:
בקובץ zip יש כרגיל קבצי Test שצריך להשלים ולהגיש

בדיקת פרמטרים של פונקציות בעזרת myassert:
בתרגיל זה עליכם לבדוק את הפרמטרים של הפונקציות בעזרת המקרו myassert המוגדר בקובץ myassert.hpp. שימו לב שמותר לכם להשתמש ב assert רגיל לבדיקת לוגיקה של התוכנית שלכם אך אתם מחוייבים להשתמש ב myassert על מנת לבדוק פרמטרים של פונקציות ואנו נשנה את המקרו על מנת לבדוק את התוכנית שלכם.
פרטים שעליכם לבדוק:
המכנה של שבר הוא אף פעם לא אפס.
מימדי מטריצות בחיבור וכפל הם נכונים.
מספר השורות והעמודות במטריצה צריך להיות גדול או שווה לאחד.

<u>הנחיות נוספות</u>

אתם יכולים להוסיף מחלקות נוספות כדוגמת Complex שראיתם בכיתה, ולנסות בעזרתה את מימוש המטריצה.
זכרו שגם הבדיקה של התרגילים תוכל לבדוק את מחלקת Matrix שלכם עם טיפוסי איברים נוספים.
הערה: מצורף קובץ Makefile שיש כרגיל לעמוד בדרישות שלו!

- חובה כרגיל להגיש גוגל טסט של התוכנית
- בדיקת הקוד לפני ההגשה, גם על ידי קריאתו וגם על ידי כתיבת בדיקות אוטומטיות (כמו שכתוב בסעיף הקודם, חובה להגיש גוגל טסט) היא אחריותכם. חישבו על מקרי קצה, חלק מהציון ניתן על עמידה בבדיקות אוטומטיות.

<u>הנחיות הגשה</u>

כרגיל עליכם להגיש קובץ ששמו מספר ת.ז. שלכם כמו שהיא מופיעה באתר המודל נקודה zip. לדוגמא, אם מספר ת.ז. שלי הוא 12345678 אז שם הקובץ יהיה:
12345678.zip
הקובץ יכיל את הקבצים הבאים בלבד (שנו לתעודת הזהות שלכם את ID ב Makefile והשתמשו ב make zipfile וב make checkzipfile על מנת ליצור את קובץ הזיפ ולבדוק אותו):

Matrix.hpp
Rational.hpp
MatrixTest.cpp
RationalTest.cpp

בהצלחה!!!