סדנת תכנות בשפת **C**++, מס׳ קורס 67317 - 2018

תרגיל 1

תאריך הגשה: יום חמישי 20.12.18 עד שעה 23:55

הגשה מאוחרת (בהפחתת 10 נקודות): יום שישי 21.12.18 עד שעה 23:55

תאריך ההגשה של הבוחן: יום חמישי 20.12.18 עד שעה 23:55

1. הנחיות כלליות:

- בכל התרגילים יש לעמוד בהנחיות הגשת התרגילים וסגנון כתיבת הקוד. שני המסמכים נמצאים באתר הקורס – הניקוד יכלול גם עמידה בדרישות אלו.
- בכל התרגילים עליכם לכתוב קוד ברור. בכל מקרה בו הקוד שלכם אינו ברור מספיק עליכם להוסיף הערות הסבר בגוף הקוד. יש להקפיד על תיעוד (documentation) הקוד ובפרט תיעוד של כל פונקציה.
- במידה ואתם משתמשים בעיצוב מיוחד או משהו לא שגרתי, עליכם להוסיף הערות בקוד המסבירות את העיצוב שלכם ומדוע בחרתם בו.
- עבור כל פונקציה בה אתם משתמשים, עליכם לוודא שאתם מבינים היטב מה הפונקציה עושה גם במקרי קצה (התייחסו לכך בתיעוד). ובפרט עליכם לוודא שהפונקציה הצליחה.
- עליכם לקמפל עם הדגלים Wall -Wextra -std=c++17 -pthread -g ולוודא שהתוכנית מתקמפלת ללא אזהרות, תכנית שמתקמפלת עם אזהרות תגרור הורדה משמעותית בציון התרגיל.
- עליכם לוודא שהתרגילים שלכם תקינים ועומדים בכל דרישות הקימפול והריצה במחשבי בית הספר מבוססי מעבדי 64-bit (מחשבי האקווריום, לוי, השרת river). חובה להריץ את התרגיל במחשבי בית הספר לפני ההגשה. (ניתן לוודא שהמחשב עליו אתם עובדים הנו בתצורת 64-bit באמצעות הפקודה "uname -a" ווידוא כי הארכיטקטורה היא 64, למשל אם כתוב 86x_64)
- לאחר ההגשה, בדקו את הפלט המתקבל בקובץ ה-PDF שנוצר מהpresubmission script בזמן ההגשה. באם ישנן שגיאות, תקנו אותן על מנת שלא לאבד נקודות.
שימו לב ! תרגיל שלא יעבור את ה **presubmission script** ציונו ירד משמעותית (הציון יתחיל מ-50, ויוכל לרדת) ולא יהיה ניתן לערער על כך.
- בדיקת הקוד לפני ההגשה, גם על ידי קריאתו וגם על ידי כתיבת בדיקות אוטומטיות (tests) עבורו היא אחריותכם. בדקו מקרי קצה.

2. הנחיות חשובות לכלל התרגילים בקורס **C**++

- הקפידו להשתמש בפונקציות ואובייקטים של C++ (למשל new, delete, cout) על פני פונקציות של C (למשל malloc, free, printf).
בפרט השתמשו במחלקה string (ב-std::string) ולא במחרוזת של C (char *).
- יש להשתמש בספריות סטנדרטיות של C++ ולא של C אלא אם כן הדבר הכרחי (וגם אז עליכם להוסיף הערה המסבירה את הסיבות לכך).
- הקפידו על עקרונות Information Hiding – לדוגמא, הקפידו כי משתני המחלקות שלכם מוגדרים כמשתנים פרטיים (private).
- הקפידו לא להעתיק by value משתנים כבדים, אלא להעבירם (היכן שניתן) by reference.
- הקפידו על שימוש במילה השמורה const בהגדרות המתודות והפרמטרים שהן מקבלות:
המתודות שאינן משנות פרמטר מסויים – הוסיפו const לפני הגדרת הפרמטר.

מתודות של מחלקה שאינן משנות את משתני המחלקה – הוסיפו const להגדרת המתודה.
שימו לב: הגדרת משתנים / מחלקות ב- C++ כקבועים הוא אחד העקרונות החשובים בשפה.

וקטורים ומטריצות

1. <u>בתרגיל זה נממש שתי מחלקות: Matrix3D and Vector3D</u>

כל המספרים הם מסוג **double**

המחלקה Vector3D שומרת וקטורים ממימד 3.

עליכם לתמוך בבנאים הבאים:
- בנאי ברירת מחדל שמאתחל לאפס
- בנאי שמקבל שלוש מספרים
- בנאי שמקבל מערך בגודל 3
- בנאי העתקה

עליכם לתמוך באופרטורים הבאים:
- אופרטורים +/- שמקבלים שני וקטורים ומחזירים וקטור שהוא תוצאת החיבור/חיסור ביניהם
- אופרטורים =+/=- שמקבלים וקטור ומחברים/מחסירים לוקטור הנוכחי
- אופרטורים =+/=- שמקבלים מספר ומחברים/מחסירים לוקטור הנוכחי
- אופרטור - שמקבל וקטור ומחזיר וקטור שהוא המינוס שלו (הכפלה ב -1)
- אופרטורים * ו- / שמקבלים וקטור ומספר ומחזירים וקטור שמוכפל/מחולק במספר
- אופרטור * שמקבל מספר ווקטור(שימו לב לסדר) ומחזיר וקטור שמוכפל במספר.
- אופרטורים =* ו- =/ שמקבלים מספר ומכפילים/מחלקים בו את הוקטור הנוכחי
- אופרטור | שמקבל שני וקטורים ומחזיר מרחק ביניהם
- אופרטור * שמקבל שני וקטורים ומחזיר תוצאה של מכפלה סקלרית ביניהם
- אופרטור ^ שמקבל שני וקטורים ומחזיר את הזווית ביניהם ברדיאנים
- אופרטור << שקורה וקטור מ stdin
- אופרטור >> שמדפיס וקטור ל stdout
- אופרטור =
- אופרטור [] לגישה קואורדינטת x, y, and z של הוקטור
- אופרטור [] להשמה קואורדינטת x, y, and z של הוקטור

עליכם לתמוך בפונקציות הבאות:
- norm המחזירה את אורך הוקטור המחושב לפי מרחק מאפס
- dist שמקבלת וקטור ומחזירה מרחק בינו לבין הוקטור הנוכחי

המחלקה Matrix3D שומרת מטריצות 3x3

עליכם לתמוך בבנאים הבאים:
- בנאי ברירת מחדל שמאתחל לאפס
- בנאי שמקבל מספר ומאתחל מטריצה עם מספר באלכסון ו 0 בשאר המקומות
- בנאי שמקבל תשע מספרים ומאתחל מטריצה
- בנאי שמקבל מערך באורך 9
- בנאי שמקבל מערך 3x3
- בנאי שמקבל שלוש Vector3D
- בנאי העתקה

עליכם לתמוך באופרטורים הבאים:
- אופרטורים =+/=- שמקבלים מטריצה ומחברים/מחסירים למטריצה הנוכחית

- אופרטור *= שמקבל מטריצה ומחזיר תוצאה של כפל במטריצה הנוכחית
- אופרטורים */+/- שמקבלים שתי מטריצות ומחזירים מטריצה שהיא תוצאת הכפל/חיבור/חיסור ביניהם
- אופרטורים =* ו- =/ שמקבלים מספר ומכפילים/מחלקים בו את המטריצה הנוכחית
- אופרטור * שמקבל מטריצה ווקטור ומחזיר את תוצאת הכפל ביניהם
- אופרטור << שקורה מטריצה מ stdin
- אופרטור >> שמדפיס מטריצה ל stdout עם newline אחרי כל שורה
- אופרטור =
- אופרטור [] לגישה לשורה במטריצה. מחזיר Vector3D
- אופרטור [] להשמה לשורה במטריצה.

עליכם לתמוך בפונקציות הבאות:

- row - המקבלת short בין 0 ל- 2 ומחזירה את השורה הרלוונטית במטריצה בתור Vector3D
- column - המקבלת short בין 0 ל- 2 ומחזירה את העמודה הרלוונטית במטריצה בתור Vector3D
- trace - המחזירה את סכום איברי האלכסון במטריצה
- determinant - המחשבת ומחזירה את הדטרמיננט של המטריצה

2. <u>דוגמאות הרצה:</u>

```
int main() {
  Vector3D a(3.0, 2.0, 5.0);
  std::cout << "a.norm = " << a.norm() << std::endl;
  Vector3D b(7.0, 1.0, 0.0);
  std::cout << "a.dist(b) = " << a.dist(b) << std::endl;
  Vector3D c(5.0, 1.0, 4.0);

  Matrix3D m(a, b, c);
  std::cout << "Matrix m = \n" << m;
  std::cout << "m.determinant = " << m.determinant() << std::endl;
  std::cout << "m*a = " << m*a << std::endl;
  return 0;
}
```

התוכנית מדפיסה את הפלט הבא:

```
a.norm = 6.16441
a.dist(b) = 6.48074
Matrix m =
3 2 5
7 1 0
5 1 4
m.determinant = -34
m*a = 38 23 37
```

- שתי המחלקות שלכם יתקמפלו לספריה, קובץ Makefile שבונה הכל מצורף
- הטסטים יהיו תוכניות שיתקמפלו עם הספריה שלכם, כמו קובץ ex1.cpp שמצורף גם

4. הערות:

- בתרגיל זה אתם רשאים להניח כי הקלט תקין
- במידה ונתקלתם בשגיאה עליכם להדפיס הודעת שגיאה ל- cerr
- התכניות יבדקו גם על סגנון כתיבת הקוד וגם על פונקציונאליות, באמצעות קבצי main שונים (תרחישים שונים להרצת התכניות).
- עליכם להקפיד על פורמט הדפסה מדויק, כדי למנוע שגיאות מיותרות והורדת נקודות. ספציפית בתרגיל זה מדובר באופרטורים >> של שתי המחלקות.
- סיפקנו לכם קובץ פלט לדוגמה (ex1.out), עליכם לוודא שהתכנית שלכם נותנת את אותו הפלט בדיוק.

חומר עזר:

1. קובץ Makefile , ex1.cpp, ו- ex1.out:

~labcpp/www/ex1/files/

הגשה:

1. עליכם להגיש קובץ tar בשם ex1.tar המכיל בדיוק את הקבצים הבאים:
   - ex1.cpp
   - Vector3D.h
   - Vector3D.cpp
   - Matrix3D.h
   - Matrix3D.cpp
   - Makefile

   - שימו לב! - על אף שאתם יכולים להוסיף קבצים נוספים כרצונכם, הימנעו מהוספת קבצים לא רלוונטים (גם בכדי להקל על הבודקים, וגם בכדי שציונכם לא יפגע מכך).

2. לפני ההגשה, פתחו את הקובץ ex1.tar בתיקיה נפרדת וודאו שהקבצים מתקמפלים ללא שגיאות וללא אזהרות.
3. מומלץ מאוד גם להריץ בדיקות אוטומטיות וטסטרים שכתבתם על הקוד אותו אתם עומדים להגיש. בנוסף, אתם יכולים להריץ בעצמכם בדיקה אוטומטית עבור סגנון קידוד בעזרת הפקודה:

~labcpp/www/codingStyleCheck <file or directory>

כאשר <directory or file> מוחלף בשם הקובץ אותו אתם רוצים לבדוק או תיקייה שיבדקו כל הקבצים הנמצאים בה (שימו לב שבדיקה אוטומטית זו הינה רק חלק מבדיקות ה codingStyle)

4. דאגו לבדוק לאחר ההגשה את קובץ הפלטsubmission.pdf)) וודאו שההגשה שלכם עוברת את ה- presubmission script ללא שגיאות או אזהרות.

~labcpp/www/ex1/presubmit_ex1

בהצלחה!