שפת C++ – תרגיל 1

הכרות עם השפה, classes, const , references,
dynamic allocation, operators overloading

<u>תאריך הגשה:</u> יום רביעי 28.12.16 עד שעה 23:55

*** הגשה מאוחרת (בהפחתת 10 נקודות): יום חמישי 29.12.16 עד שעה 23:55**

**** הגשה מאוחרת (בהפחתת 20 נקודות): מוצאי שבת 31.12.16 עד שעה 23:55**

<u>תאריך ההגשה של הבוחן:</u> יום רביעי 28.12.15 עד שעה 23:55

1. הנחיות חשובות:

1. בכל התרגילים יש לעמוד בהנחיות הגשת התרגילים וסגנון כתיבת הקוד. שני המסמכים נמצאים באתר הקורס – הניקוד יכלול גם עמידה בדרישות אלו.
2. 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.
3. במידה ואתם משתמשים בעיצוב מיוחד או משהו לא שגרתי, עליכם להוסיף הערות בקוד המסבירות את העיצוב שלכם ומדוע בחרתם בו.
4. בכל התרגילים במידה ויש לכם הארכה ואתם משתמשים בה, **<u>חל איסור להגיש קובץ כלשהוא בלינק הרגיל (גם אם לינק ההגשה באיחור טרם נפתח)</u>**. מי שיגיש קבצים בשני הלינקים מסתכן בהורדת ציון משמעותית.
5. אין להגיש קבצים נוספים על אלו שתדרשו.
   a. אין להגיש קובץ README אלא אם צוין במפורש שיש צורך בכך (לדוגמא, בתרגיל זה אין צורך להגיש).
   b. עליכם לקמפל עם הדגלים Wall -Wextra- ולוודא שהתוכנית מתקמפלת ללא אזהרות, **תכנית שמתקמפלת עם אזהרות תגרור הורדה בציון התרגיל.** למשל, בכדי ליצור תוכנית מקובץ מקור בשם ex1.cpp יש להריץ את הפקודה:

```
g++ -std=c++11 -Wextra -Wall ex1.cpp -o ex1
```

6. עליכם לוודא שהתרגילים שלכם תקינים ועומדים בכל דרישות הקמפול והריצה במחשבי בית הספר מבוססי מעבדי bit-64 (מחשבי האקוריום, לוי, השרת river). **<u>חובה להריץ את התרגיל במחשבי בית הספר לפני ההגשה</u>**. (ניתן לוודא שהמחשב עליו אתם עובדים הנו בתצורת bit-64 באמצעות הפקודה "uname -a" ווידוא כי הארכיטקטורה היא 64, למשל אם כתוב x86_64)
7. לאחר ההגשה, בדקו הפלט המתקבל בקובץ ה-PDF שנוצר מה presubmission script בזמן ההגשה. באם ישנן שגיאות, תקנו אותן על מנת שלא לאבד נקודות.
   <u>שימו לב ! תרגיל שלא יעבור את ה presubmission script ציונו ירד משמעותית</u> (הציון יתחיל מ-50, ויוכל לרדת) **<u>ולא יהיה ניתן לערער על כך.</u>**
8. בדיקת הקוד לפני ההגשה, גם על ידי קריאתו וגם על ידי כתיבת בדיקות אוטומטיות (tests) עבורו היא אחריותכם. חישבו על מקרי קצה לבדיקת הקוד.

9. **הגשה מתוקנת -** לאחר מועד הגשת התרגיל ירוצו הבדיקות האוטומטיות ותקבלו פירוט על הטסטים בהם נפלתם. לשם שיפור הציון יהיה ניתן להגיש שוב את התרגיל לאחר תיקוני קוד קלים ולקבל בחזרה חלק מהנקודות - <u>פרטים מלאים יפורסמו בפורום ואתר הקורס</u>.

2. הנחיות חשובות לכלל התרגילים בקורס C++

1. 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 (char *) C!
2. יש להשתמש בספריות סטנדרטיות של C++ ולא של C אלא אם כן הדבר הכרחי (וגם אז עליכם להוסיף הערה המסבירה את הסיבות לכך).
3. הקפידו על עקרונות Information Hiding – לדוגמא, הקפידו כי משתני המחלקות שלכם מוגדרים כמשתנים פרטיים (private).
4. הקפידו לא להעתיק by value משתנים כבדים, אלא להעבירם (היכן שניתן) by reference.
5. **<u>הקפידו מאוד</u>** על שימוש במילה השמורה const בהגדרות הפונקציות והפרמטרים שהן מקבלות.
   פונקציות 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.
6. הקפידו על השימוש ב- static, במקומות המתאימים (הן במשתנים והן במתודות)
7. הקפידו לשחרר את כל הזיכרון שאתם מקצים (**השתמשו ב-valgrind כדי לבדוק שאין לכם דליפות זיכרון**).

3. הנחיות ספציפיות לתרגיל זה:

1. חל איסור להשתמש במבני נתונים מוכנים בתרגיל (כדוגמת STL) שימוש כזה יוביל לפסילת הסעיף הרלוונטי.
2. בתרגיל זה אתם רשאים להוסיף קבצים נוספים ולהוסיף דברים לקבצי ה-h (<u>אך אין לשנות את חתימת המתודות שמופיעות שם</u>).
3. בתרגיל זה אנו מניחים שאין שגיאות (הקצאות זיכרון מצליחות תמיד, הפרמטרים לפונקציות חוקיים וכו').

4. מידע חשוב נוסף:

1. ניתן להתחבר באמצעות SSH למחשבי בית הספר (למשל לשם בדיקת הקוד לפני הגשה מהבית)
   http://wiki.cs.huji.ac.il/wiki/Connecting_from_outside
2. עליכם להכיר את ספריית הקלט-פלט של שפת CPP ובייחוד את השימוש בפונקציות cout ו-cin
   http://www.cplusplus.com/doc/tutorial/basic_io/

5. Set:

1. בשאלה זאת מבנה נתונים מסוג Set - הכוונה למבנה נתונים שבו כל מפתח (key) מופיע לכל היותר פעם אחת.
2. את ה-Set עליכם לממש בקבצים MySet.h ו-MySet.cpp.
3. מבנה הנתונים יכיל מחרוזות ומספרים ממשיים

   (כל צומת יחזיק מחרוזת (key) ומספר ממשי (data)).
4. בנוסף לבנאי והורס עליכם לממש את השיטות הבאות:
   - המתודה remove מקבלת מחרוזת, ומוציאה את כל האיברים (כלומר 0 או 1 איברים) בקבוצה המכילים מחרוזת זאת. המתודה מחזירה את מספר האיברים שהוסרו.
   - המתודה isInSet מקבלת שני פרמטרים: מפתח (מחרוזת) ו reference ל- double. המתודה מחזירה ערך בוליאני true אם המפתח מופיע בקבוצה ו- false אחרת. אם המפתח הופיע ברשימה היא תעדכן את המשתנה של ה- reference להיות ה- data של המפתח שנמצא. ראו קריאה למתודה בקובץ SetExample.cpp.
   - המתודה printSet תדפיס את כל איברי הקבוצה בסדר ממוין על פי ערך ה-Hash של המפתח מהנמוך לגבוה. כל איבר בשורה נפרדת. כל שורה תכיל זוג ערכים המופרדים ע"י פסיק יחיד, ללא רווחים:

     `<string value>,<double value>`

     אם הקבוצה ריקה, תודפס המילה "EMPTY". זכרו בסוף כל הדפסת קבוצה (בין אם היא מכילה מחרוזת ובין אם היא מכילה את המילה EMPTY) להדפיס תו ירידת שורה.

     (עבור ערכים בעלי אותו ה- Hash, אתם חופשיים לבחור את דרך ההדפסה)
   - המתודה sumSet מחזירה את סכום המספרים הממשיים בקבוצה (סכום כל ה- data של אברי הקבוצה).
   - המתודה totWeight מחזירה את סכום הערכים שמחזירה myHashFunction על כל המפתחות שנמצאים ב-Set.
   - המתודה myHashFunction היא פונקצית גיבוב על מחרוזות (ראו פירוט בקובץ MySet.h).
5. <u>הנחיות וטיפים -</u>
   - **אין להשתמש במבני נתונים מוכנים (כמו STL) לצורך פתרון התרגיל.**
   - שימו לב למספר הפעמים שאיבר יכול להופיע.
   - יש לממש את המתודות add, remove ו- isInSet בסיבוכיות של לכל היותר O(מספר האיברים ברשימה).
   - וודאו שהקוד שלכם עובד עם קובץ הדוגמא ושהפלט שלו תואם.
   - אתם מוזנים לממש את ה-Set בכל דרך שתבחרו. אנו ממליצים לממש זאת באמצעות רשימה מקושרת.
   - ניתן לממש את הקוד ע"י מחלקה יחידה.

   ניתן (אך לא חובה) להשתמש במחלקה נוספת (MyNode) שתייצג איבר בקבוצה. אם בחרתם לממש את המחלקה MyNode, הקפידו להגיש אותה. שימו לב כי ניתן לממש את MyNode

כולה (כולל הצהרת ה-class) בקובץ MySet.cpp, ולא לאזכר את קיום המחלקה בקובץ ה-header (כך יותר נכון מבחינת information hiding) או לכל היותר לאזכר את קיומה כמחלקה פנימית של MySet ללא תיאור של התוכן שלה.

- טיפול בשגיאות ב-C++ מבוצע ע"י מנגנון חריגות (exceptions). אתם תלמדו על הנושא בהמשך הקורס. בתרגיל זה אנו מניחים שאין שגיאות (הקצאות זיכרון מצליחות תמיד, הפרמטרים לפונקציות חוקיים וכו').

  בפרט, פעולת הוספת איבר לקבוצה מצליחה תמיד (אחרי כל קריאה לפונקציה של הוספת איבר לקבוצה, באם המפתח אינו קיים בקבוצה, גודל הקבוצה יגדל ב-1).
- מסופקים לכם קובץ בדיקה בסיסי SetExample.cpp וקובץ הפלט הצפוי SetExample.out.

6. מימוש אופרטורים עבור MySet:

1. עליכם לממש את האופרטורים הבאים עבור המחלקה MySet :

- אופרטורי השוואה:

  a. operator>
  b. operator<
  c. operator==

  אופרטורים אלו יערכו את ההשוואות בין שתי Sets כאשר הערך הקובע בהשוואה הוא משקלן (כפי שמוחזר מהפונקציה totWeight שהוגדרה בחלק הקודם).
  ערך ההחזרה של האופרטורים הוא bool.
- operator -

מאחר ו-MySet הינו Set (אף מפתח אינו מופיע יותר מפעם אחת), אופרטור זה יתנהג כמו חיסור על Set. הפונקציה תחזיר קבוצה חדשה המכילה את האיברים מה-קבוצה השמאלית אשר המפתחות שלהם אינם מופיעים בקבוצה הימנית[1].

operator |

זהו אופרטור המבצע את פעולת ה-Union על sets. הפונקציה תחזיר קבוצה חדשה המכילה את כל האיברים מהקבוצה השמאלית וכן את כל האיברים מהקבוצה הימנית אשר אינם מופיעים בקבוצה השמאלית. זאת אומרת שאם היו במקור איברים עם מפתח שמופיע גם ב-left וגם ב-right, אז בקבוצה החדשה הערך של המפתחות הללו יהיה הערך שהופיע ב-left והערכים של מפתחות מ-right שלא היו ב-left כלל ישמרו[2].

- operator &

זהו אופרטור המבצע את פעולת ה-Intersection על sets. הפונקציה תחזיר קבוצה חדשה המכילה את כל האיברים שמפתחותיהם מופיעים גם בקבוצה השמאלית וגם בימנית. הערכים שישמרו במפה החדשה יהיו הערכים שהיו בקבוצה השמאלית. זאת אומרת שאם היו במקור איברים עם מפתח שמופיע גם ב-left וגם ב-right, אז בקבוצה החדשה הערך של המפתחות הללו יהיה הערך שהופיע ב-left בלבד[3].

---

[1] מסומנת בדרך כלל "left\right"
[2] מסומנת בדרך כלל "left∪right"
[3] מסומנת בדרך כלל "left∩right"

- אופטורי השמה

עליכם לממש פונקציות (בנאי העתקה) ואופרטורים בכדי לתמוך בפעולות השמה.

למשל - "mySet1=mySet2".

אתם רשאים לשנות את MySet.h להעביר מתודות מ-private ל-public במידת הצורך. עליכם להוסיף תיעוד המסביר מהי הסיבה בגללה השינוי בוצע.

2. עליכם לכתוב גם קובץ בשם SetBinaryOperations.cpp המכיל פונקצית main ויוצר קובץ הרצה המדגים את השימוש בכל האופרטורים שהוגדרו כאן.
3. <u>הנחיות וטיפים -</u>
   - יש לממש את האופרטורים במחלקה עצמה בקבצים MySet.h ו-MySet.cpp כפונקציות מחלקה. הבנת החתימות הנדרשות של פונקציות האופרטורים הן חלק מהדרישות.
   - בשאלה זו אין צורך לממש מחלקה כלשהי, אלא רק לממש את המתודות המבוקשות (ניתן לכתוב גם מתודות עזר).
   - מותר לכם להוסיף מתודות למחלקה MySet מלבד האופרטורים.
   - הקפידו שהפונקציות תעבודנה ללא דליפות זיכרון.
   - הקפידו שלא לשכפל קוד בין האופרטורים השונים.

7. חומר עזר:

1. את קבצי התרגיל ניתן למצוא ב:

```
~labcpp/www/ex1/ex1_files.tar
```

2. את קבצי הבדיקה ניתן למצוא ב:

```
~labcpp/www/ex1/ex1_tests.tar
```

עיינו בהם כדי לראות איך התכנית צריכה להתנהג

בדקו את תכניתכם וודאו שהפלטים שלכם זהים לאלה של פתרון בית הספר. אתם יכולים לייצר קבצי קלט רבים נוספים כדי לבדוק מקרים נוספים, ולהשוות את הפלט של התכנית שלכם עלטים של תלמידים אחרים, או עם הפלט שנוצר כשאתם נותנים את הקלט הזה לקובץ הריצה של פתרון בית הספר.

3. ביצוע overloading ב-++C:

http://www.cprogramming.com/tutorial/operator_overloading.html
http://en.wikipedia.org/wiki/Operators_in_C_and_C%2B%2B
http://www.cplusplus.com/reference/set/set/operators/

8. עבודה עם valgrind:

1. בתרגיל זה (כמו ביתר התרגילים בקורס) תידרשו להשתמש בניהול זיכרון דינמי.
2. ישנו מבחר די גדול של תוכנות בשוק שמטרתם לסייע באיתור בעיות זיכרון בקוד לפני שחרורו אל הלקוח. אנו נשתמש בתוכנת valgrind, שיחסית לתוכנה חינמית, נותנת תוצאות מעולות.
3. כדי להריץ את valgrind עליכם לבצע קומפילציה ו-linkage לקוד שלכם עם הדגל '-g' (הן בשורת הקומפילציה והן בשורת ה-linkage). לאחר מכן הריצו valgrind:

```
> valgrind --leak-check=full --show-possibly-lost=yes
     --show-reachable=yes –undef-value-errors=yes  IntMatrixMainDriver
```

4. אם קיבלתם הודעת שגיאה, יתכן שתצטרכו לבצע שינוי הרשאות:

```
> chmod 777 IntMainDriver
```

5. כמובן שאם valgrind דיווח על בעיות עם הקוד שלכם, עליכם לתקן אותן.
6. היעזרו ב-tutorial הקצרצר של valgrind שבאתר הקורס.

9. הגשה:

1. עליכם להגיש קובץ tar בשם ex1.tar המכיל לפחות את הקבצים הבאים:
   - MySet.h ו-MySet.cpp
   - SetBinaryOperations.cpp
   - קובץ Makefile התומך בפקודות הבאות:
     - make tar - יצירת קובץ tar בשם ex1.tar, המכיל רק את הקבצים שצריך להגיש בתרגיל.
     - make clean - ניקוי כל הקבצים שנוצרו באמצעות פקודות ה-makefile.
     - הרצת make ללא פרמטרים תבנה קובץ הרצה בשם **ex1**
   - extension.pdf - רק במקרה שההגשה היא הגשה באיחור.
2. ניתן ליצור קובץ tar כדרוש על ידי הפקודה:

```
tar cvf <tar_name> <files>
```

3. לפני ההגשה, פתחו את הקובץ ex1.tar בתיקיה נפרדת וודאו שהקבצים מתקמפלים ללא שגיאות וללא אזהרות.
4. מומלץ מאוד גם להריץ בדיקות אוטומטיות וטסטרים שכתבתם על הקוד אותו אתם עומדים להגיש.
5. 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)

6. דאגו לבדוק לאחר ההגשה את קובץ הפלט (submission.pdf) וודאו שההגשה שלכם עוברת את ה-presubmission script ללא שגיאות או אזהרות.

```
~labcpp/www/ex1/presubmit_ex1
```

בהצלחה!