סדנת תכנות בשפת C++, מס' קורס 67319 - 2018
תרגיל 3

Templates, Iterators, Smart pointers, Move semantics

תאריך הגשה: 23:55 17/09/2018
הגשה מאוחרת (בהפחתת 10 נקודות): 23:55 18/09/2018
תאריך ההגשה של הבוחן: 23:55 18/09/2018

הנחיות חשובות לכלל התרגילים:

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. 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.
5. בכל התרגילים במידה ויש לכם הארכה, או שאתם מגישים באיחור. **חל איסור להגיש קובץ כלשהו בלינק הרגיל (גם אם לינק overdue טרם נפתח). מי שיגיש קבצים בשני הלינקים מסתכן בהורדת ציון משמעותית.**
6. אין להגיש קבצים נוספים על אלו שתדרשו. ובפרט אין להגיש קובץ README אלא אם צוין במפורש שיש צורך בכך. (לדוגמא, בתרגיל זה אין צורך להגיש).
7. עליכם לקמפל עם הדגלים Wall -Wextra -Wvla -std=c++14 -g -pthreads ולוודא שהתוכנית מתקמפלת ללא אזהרות, **תכנית שמתקמפלת עם אזהרות תגרור הורדה משמעותית בציון התרגיל.** למשל, בכדי ליצור תוכנית מקובץ מקור בשם ex1.cpp יש להריץ את הפקודה:

```
g++ -Wextra -Wall -Wvla -std=c++14 -g -pthread  ex1.cpp -o ex3
```

8. עליכם לוודא שהתרגילים שלכם תקינים ועומדים בכל דרישות הקימפול והריצה במחשבי בית הספר מבוססי מעבדי bit-64 (מחשבי האקוריום, לוי, השרת river). **חובה להריץ את התרגיל במחשבי בית הספר לפני ההגשה**. (ניתן לוודא שהמחשב עליו אתם עובדים הנו בתצורת bit-64 באמצעות הפקודה "uname -a" ווידוא כי הארכיטקטורה היא 64, למשל אם כתוב x86_64)
9. 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.**
10. בדיקת הקוד לפני ההגשה, גם על ידי קריאתו וגם על ידי כתיבת בדיקות אוטומטיות (tests) עבורו היא אחריותכם. בדקו מקרי קצה.
    במידה וסיפקנו לכם קבצי בדיקה לדוגמא, השימוש בהם יהיה על אחריותכם. **במהלך הבדיקה הקוד שלכם ייבדק מול קלטים נוספים לשם מתן הציון.**
11. **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 ולקבל בחזרה חלק מהנקודות - פרטים מלאים יפורסמו בפורום ואתר הקורס.

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 C (char *).
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. הקפידו מאוד על שימוש במילה השמורה const בהגדרות המתודות והפרמטרים שהן מקבלות:
המתודות שמקבלות משתנה ייחוס (reference) או מצביע ואינן משנות אותו – 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 לשחרר את כל הזיכרון שאתם מקצים (השתמשו ב-valgrind כדי לבדוק שאין לכם דליפות זיכרון).
7. שימו לב שהאלגוריתמים שלכם צריכים להיות יעילים.
8. אתם רשאים (ולעתים אף נדרשים) להגדיר פונקציות נוספות לשימושכם הפנימי.

הנחיות ספציפיות לתרגיל זה:

1. בתרגיל זה אסור לכם להקצות זיכרון בעצמיכם (כלומר שאין להשתמש בפונקציות new, delete או במקבילותיהן מ- C).
2. עליכם לוודא שהקוד שלכם רץ באופן תקין וללא דליפות זכרון (גם שאינן נובעות מפעולה ישירה שלכם). לשם כך עליכם להשתמש בתוכנת valgrind.
3. אתם רשאים (ולעתים אף נדרשים) להגדיר פונקציות נוספות לשימושכם הפנימי.
4. שימו לב שאתם מכירים כל פונקציה בה אתם משתמשים ושאתם בודקים עבור כל פונקציה שהיא הצליחה.
5. בתרגיל זה עליכם להתמודד עם מצבים לא צפויים באמצעות מנגנון החריגות (exceptions). וודאו כי אתם ממשים את הכללים אותם למדנו בקורס בנוגע לשימוש נכון בחריגות.

קבוצה מתמטית כללית באמצעות עץ בינארי

Generic set implemented as a binary tree

בתרגיל זה תכתבו מבנה נתונים של קבוצה מתמטית הממומשת כעץ בינארי, אשר יכולה להכיל כל טיפוס נתונים בדומה למבני הנתונים התקניים של השפה – ספריות STL. העץ יהיה ממוין, ולכן יתמוך בטיפוסי נתונים שמוגדר עליהם יחס סדר.
על הקבוצה לתמוך בהוספה והסרה של איברים, מלוא יכולות ההעברה והעתקה של השפה, וכן במעבר על העץ באמצעות איטרטורים. כל יכולות הקבוצה וחתימות הפונקציות שלה להתאים לספריה התקנית, ולשם כך עליכם להיעזר בקישור הבא שבו ניתן למצוא פירוט מלא.

פונקציות ותכונות נדרשות:

וודאו כי אתם ממשים את הפונקציות לפי ההערות בהמשך.

1. בנאים מכל הסוגים והמינים: בנאי ברירת מחדל, בנאי העתקה, בנאי העברה, אופרטור השמה, ובנאי המקבל איטרטור התחלה ואיטרטור סיום ומוסיף את האיברים לעץ.
2. פונקציות תמיכה תקניות: size, empty.
3. פונקציות גישה תקניות: insert, clear, find, erase.
4. פונקציות גישה על ידי איטרטורים: cbegin, cend, crbegin, crend.
5. פונקציית swap על פי תקן std.
6. בונוס (5 נקודות): פונקציית בניה במקום, emplace.

בעלות על זיכרון:

העץ מוגדר כבעלים על כל האיברים שנמצאים בו, ולתכנון זה יש את ההשלכות הבאות:

לכל סוגי האיברים:

- קבלת איברים: על הפונקציות המקבלות איברים לתמוך בקבלה על ידי const reference ולייצור עותק מקומי, וכן בקבלה על ידי r-value reference (move semantics).
- בגישה לאיברים, האיברים יוחזרו כ- const reference.

הערות:

- במקרה של הוספת איבר שכבר נמצא בקבוצה, יש להשאיר את האיבר הקיים.
- לשם הפשטות, אין צורך לתמוך באיטרטורים המאפשרים שינוי האיברים אלא רק באיטרטורים קבועים. שימו לב שיש צורך לתמוך באיטרטורים דו-כיווניים ובאיטרטורים העוברים מהסוף להתחלה.
- שימו לב כי עליכם לממש את הפונקציות לפי החתימות בספריה התקנית, כלומר:
  - לפונקציה insert יש מספר גרסאות: הוספה של איבר, הוספה של איבר בעזרת רמז, הוספה של רצף (אין צורך לתמוך ב- initializer_list). עליכם להחזיר טיפוסים וערכים מתאימים לפי הספריה התקנית.
  - לשאר הפונקציות: שימו לב במיוחד לחריגות ולסימוני const ו- noexcept.
- אין צורך לאזן את העץ.

הערות כלליות למשימות התכנות:

1. התכניות יבדקו גם על סגנון כתיבת הקוד וגם על פונקציונאליות, באמצעות קבצי קלט שונים (תרחישים שונים להרצת התכניות). הפלט של פתרונותיכם יושווה (השוואת טקסט) לפלט של פתרון בית הספר. לכן עליכם להקפיד על פורמט הדפסה מדויק, כדי למנוע שגיאות מיותרות והורדת נקודות.

2. קבצי הבדיקה שסיפקנו לכם מכילים דוגמאות מועטות לבדיקה של חלק קטן מהקוד. עליכם לכתוב בדיקות נוספות על מנת לוודא את תקינות המימוש שלכם. מותר (ואף מומלץ) לשתף את הבדיקות שכתבתם בפורום המיועד לכך, כל עוד הן אינם חושפות פרטי מימוש. בדיקות טובות יזכו את כותביהן בנקודות בונוס.

הגשה:

1. עליכם להגיש קובץ tar בשם ex3.tar המכיל **רק** את הקבצים הבאים:
- my_set.h
- my_set.cpp
- קובץ Makefile התומך לפחות בפקודות הבאות:
  - make testset – הידור, יצירה והרצה של תוכנית testset (עם קובץ בשם my_set_tester.cpp שעשוי להיות שונה מזה שסופק לכם).
  - make clean – ניקוי כל הקבצים שנוצרו באמצעות פקודות ה-Makefile (וניתן לשחזר באמצעות קריאה מחודשת לפקודות ה- make המתאימות)

- extension.pdf - רק במקרה שההגשה היא הגשה מאושרת באיחור בקישור ex3_late (מכיל את האישורים הרלוונטים להארכה).

2. לפני ההגשה, פתחו את הקובץ ex3.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:
~plabc/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 הפלט (submission.pdf) וודאו שההגשה שלכם עוברת את ה-presubmission script ללא שגיאות או אזהרות.

~plabcpp/www/ex1/presubmit_ex3

בהצלחה!