שפת C++ – תרגיל 2

תכנות מונחה עצמים, ירושה, פולימורפיזם, תכנון, STL

תאריך הגשה של התרגיל והבוחן: **יום ראשון 10.09.17 עד שעה 23:55** [1]

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. 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 כלשהוא 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++17 -pthread -g 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++17 -pthread -g2 ex1.cpp -o ex1

8. עליכם לוודא שהתרגילים שלכם תקינים ועומדים בכל דרישות הקימפול והריצה במחשבי בית הספר מבוססי מעבדי bit-64 (מחשבי האקווריום, 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.

[1] ניתן להגיש באיחור של עד 24 שעות עם קנס של 10 נקודות.

2. הנחיות חשובות לכלל התרגילים בקורס ++C

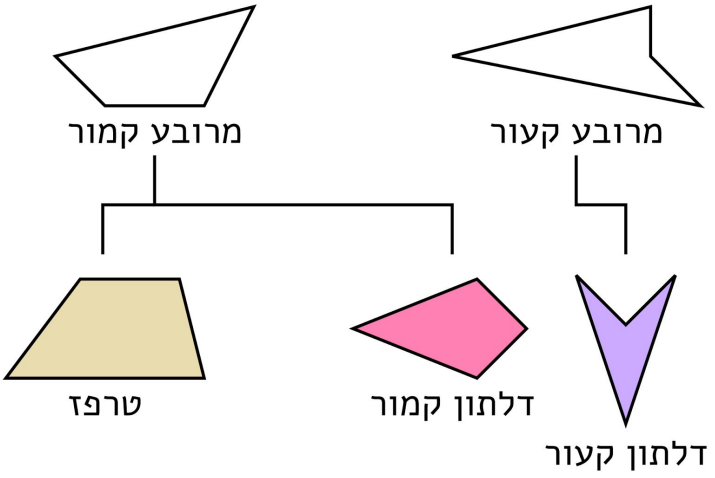
1. הקפידו להשתמש בפונקציות ואובייקטים של ++C (למשל new, delete, cout) על פני פונקציות של C (למשל malloc, free, printf).
בפרט השתמשו במחלקה string (ב-std::string) ולא במחרוזת של C (char *).
2. יש להשתמש בספריות סטנדרטיות של ++C ולא של C אלא אם כן הדבר הכרחי (וגם אז עליכם להוסיף הערה המסבירה את הסיבות 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 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).
8. שימו לב שהאלגוריתמים שלכם צריכים להיות יעילים.
9. אתם רשאים (ולעתים אף נדרשים) להגדיר פונקציות נוספות לשימושכם הפנימי.

3. הנחיות ספציפיות לתרגיל זה:

1. בתרגיל זה אתם רשאים להוסיף קבצים נוספים.
2. בתרגיל זה אתם נדרשים להשתמש ב STL, (לדוגמא vector-ishared_ptr עשויים לעזור).
3. 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').
4. מומלץ מאוד לקרוא ולהבין כל התרגיל לפני שאתם מתחילים לתכנן את מימוש התרגיל. ה-design שלכם יבדק.
5. בתרגיל זה אתם יכולים להניח כי תוכן קבצי הקלט תקינים, וכי הם יהיו בדיוק בפורמט שתואר בתרגיל.

4. חיתוך צורות וחישוב שטח - Shapes:

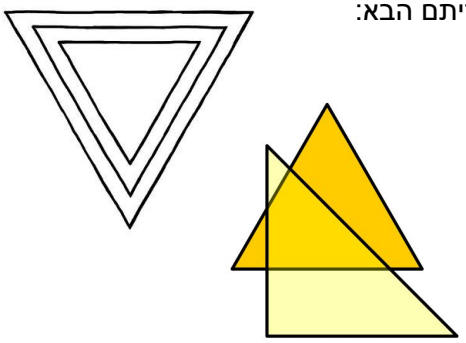
1. צורה קמורה - צורה במישור הדו-ממדי נקראת קמורה אם כל קטע של קו ישר המחבר שתיים מנקודותיה שייך כולו לצורה.
למשל:



2. בתרגיל זה בהינתן קבוצה של צורות קמורות במישור הדו-ממדי, עליכם לבדוק שאף צורה לא חותכת את רעותה ולחשב את השטח הכולל שלהן.
3. לצורך הפשטות, התרגיל יתמקד בשתי צורות:

a. משולשים.
b. טרפזים - שצלעותיהם המקבילות, מקבילות לציר ה-X.

עם זאת, עליכם לתכנן ולממש את התרגיל בצורה שתאפשר בקלות להוסיף צורות נוספות בהמשך.



4. לבדיקה האם שתי צורות קמורות נחתכות עליכם לממש את האלגוריתם הבא:
יהיו S1,S2 שתי צורות קמורות:
a. בדוק האם צורה S1 מוכלת ממש בצורה S2, או להיפך.
=> במידה ואחת מהן מוכלת ממש בשנייה החזר "נחתך".
b. לכל זוג צלעות:
=> במידה והצלעות e1,e2 נחתכות החזר "נחתך".
c. החזר "לא נחתך".

בתרגיל אנחנו מתעלמים ממקרה קצה בו אחד מהקודקודים של מצולע אחד נמצא על צלע/קודקוד של מצולע אחר.

5. להלן מספר משפטים שיקלו עליכם לממש את האלגוריתם:
a. צורה S מוכלת ממש בתוך צורה 'S ⇔ כל קודקוד $p \in S$ נמצא בתוך צורה 'S.
b. הנקודה p נמצאת בתוך צורה קמורה'S ⇔ כשעוברים על כל צלעות S לפי סדר (הצלע הבאה מתחילה בנקודה בה הסתיימה הצלע הקודמת) הנקודה p נמצאת תמיד באותו הצד של כל הצלעות.
c. הקווים (p3,p4), (p1,p2) נחתכים ⇔ שני התנאים הבאים מתקיימים:
   - הנקודות p1,p2 נמצאות בצדדים מנוגדים של הקטע (p3,p4).
   - הנקודות p3,p4 נמצאות בצדדים מנוגדים של הקטע (p1,p2).
d. יהיו p(x1,y1), p2(x2,y2), p3(x3,y3) שלוש נקודות במרחב, ויהי k

$$k = \frac{1}{2}\det\begin{pmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{pmatrix}$$

אזי:

- הערך המוחלט של k שווה לשטח המשולש שקודקודיו p, p2, p3.
- אם k < 0 אזי הנקודה p נמצאת מימין[2] לישר (p2,p3)
- אם k > 0 אזי הנקודה p נמצאת משמאל לישר (p2,p3)

(אם k==0 אזי הנקודה נמצאת על היישר, וכאמור בתרגיל הזה אנחנו מתעלמים ממקרה קצה כזה).

[2] כלומר מי שהולך מ-p2 ל-p3 יראה את הנקודה p1 מצד ימין שלו.

6. עליכם לכתוב תוכנית בשם Shapes (בין הקבצים צריך להיות קובץ בעל פונקציית main, בשם Shapes.cpp) המקבלת שני ארגומנטים באופן הבא:

```
>Shapes <input_file_name> [<output_file_name>]
```

7. הארגומנט הראשון הוא שם של קובץ המכיל את הקלט לתוכנית.
8. הארגומנט השני (אם קיים) הוא שם של קובץ אליו יודפס פלט התוכנית.במידה והתוכנית הועלה מבלי ארגומנט זה, פלט התוכנית יהיה ל- standard output.
9. אם מספר הפרמטרים אינו נכון, או אם פתיחת הקובץ לקריאה/כתיבה נכשלה, על התוכנית להדפיס הודעת שגיאה מתאימה ל-std::cerr ולצאת עם ערך שגיאה 1-.
10. פורמט הקלט הוא:

```
<Shape description>
...
<Shape description>
```

11. Shape description - שורה המכילה תיאור צורה.
    a. טרפז מתואר באופן הבא:

```
t\t<x1>\t<y1>\t<x2>\t<y2>\t<x3>\t<y3>\t<x4>\t<y4>
```

    b. משולש מתואר באופן הבא:

```
T\t<x1>\t<y1>\t<x2>\t<y2>\t<x3>\t<y3>
```

    c. בין כל שני אברים סמוכים בשורה מופיע tab יחיד ('\t'). ובסוף שורה מופיע '\n'.
    d. אתם רשאים להניח כי הקלט יהיה בפורמט תקין.
    e. עליכם להניח כי כל 2 נקודות שמופיעות אחת ליד השניה בקלט מייצגות צלע.
    f. שימו לב שמספר הצורות לא ניתן לכם, ואסור לקרוא את הקובץ יותר מפעם אחת.
12. בדיקת חוקיות הצורות:
    a. עליכם לוודא כי הצורות חוקיות ולא מנוונות - בדקו את כל מקרי הקצה.
    b. במקרה של טרפז - עליכם לוודא גם כי הקווים(x1,y1),(x2,y2) ו-(x3,y3),(x4,y4) מקבילים לציר ה-x
    c. אם אחת מהצורות אינה חוקית, הדפיסו הודעת שגיאה אינפורמטיבית לפלט השגיאה הסטנדרטי וצאו עם ערך החזרה 1-.
13. פלט התוכנית:
    a. במקרה שכל הצורות תקינות. התוכנית שלכם תבדוק כי כל הצורות אינן נחתכות.
    b. במידה וקיימות 2 (או יותר) צורות נחתכות, התוכנית שלכם:
        - תדפיס את 2 הצורות הראשונות הנחתכות באמצעות הפונקציות המתאימות (שמופיעות בקובץ PrintOuts.h).
          הצורות יודפסו לפי הסדר בו הן הופיעו בקובץ הקלט.
          ובדיקת הזוגות תתבצע גם היא בהתאם לסדר בו בן הופיעו בקובץ הקלט.
        - תדפיס הודעה שמצאתם חיתוך צורות באמצעות הפונקציה reportDrawIntersect (שמופיעה בקובץ PrintOuts.h). .
    c. במידה וכל הצורות זרות זו לזו, התוכנית שלכם:
        - תדפיס באמצעות הפונקציה printArea את סך השטח שכל הצורות תופסות.
    d. עיינו בדוגמאות הקלט והפלט שמפורסמות בתיקיית הקורס.
14. תכנון ומבנה הנתונים:
    a. זהו תרגיל בתכנון (design) ובמימוש. יש לכם חופש לתכנן את חלוקת המחלקות, תפקידיהן והקשרים ביניהן. למעט:
    b. על כל המחלקות המייצגות צורה/נקודה/קו להכיל מחרוזת const data member שתכיל את שם הצורה.

c. כשיש כמה אפשרויות, תצטרכו לשקול יתרונות וחסרונות של כל אחת ולבצע בחירה לפי שיקול דעתכם. אבל זה גם אומר שלבודקים יהיה מקום לשיקול דעת. חופש בתכנון לא אומר שכל תכנון הוא טוב.מומלץ לתכנן את מבנה המחלקות היטב לפני שתתחילו לממש את הקוד.
d. זהו תרגיל שמטרתו לימוד ירושה ופולימורפיזם. חובה עליכם להשתמש בירושה.
e. שמות המחלקות, המשתנים והפונקציות צריכים להיות אינפורמטיביים ובעלי משמעות לפי תפקידם.
f. תעדו בבירור את הקוד שלכם.
g. חישבו היטב כיצד להשתמש ב const, אילו רכיבים (מתודות, משתנים) צריכים להיות סטטיים, אילו מחלקות מייצרות מופעים ואילו לא (ולכן צריכות להיות אבסטרקטיות)
h. חישבו אילו מתודות צריכות להיות מוגדרות כ pure virtual.
i. חישבו אילו בנאים צריכים להיות ציבוריים ואילו צריכים להיות חבויים.
j. הוסיפו בקובץ README תיאור קצר של תכנון התכנית שלכם (אילו מחלקות ישנן, מה מבנה הירושה. מה הקשרים בין המחלקות), והסבר קצר על החלטות שביצעתם, למה בחרתם לתכנן כך.

15. הדרכה והנחיות כלליות:

a. קריאת הקבצים הוא לא החלק החשוב של התרגיל ואנחנו לא רוצים שתשקיעו יותר מדי זמן בזה. החלק החשוב הוא תכנון (design) התכנית (חלוקה למחלקות, תכנון ירושה) ושימוש בירושה ובפולימורפיזם. כדאי שקודם תשקיעו בתכנות ובבניית המחלקות, ורק אז תפנו לחלק של קריאת הנתונים מהקבצים (רק אז תדעו מה לעשות עם הנתונים שתקראו – אילו אובייקטים לייצר וכדו').
b. לשם מימוש האלגוריתם, תצטרכו לאפשר מעבר על הצלעות, מומלץ לעשות זאת באמצעות איטרטור.
c. **אין לשנות את הקבצים PrintOuts.cpp, PrintOuts.h, Defs.h** (אולם אתם רשאים לייצר קובץ כותר נוסף ויכלול דברים נוספים) .
d. כל ההדפסות צריכות להתבצע בדיוק של 2 ספרות אחרי הנקודה (ניתן להשתמש ב-std::setprecision לשם כך).
e. כל הקואורדינטות תוצגנה באמצעות מספרים ממשיים מסוג CordType (שמוגדר בקובץ Defs.h) (כך, ניתן לשפר את רמת הדיוק באמצעות שינוי במקום אחד בלבד.)
f. ההשוואה בין מספרים צריכה להתבצע בדיוק של EPSILON (שמוגדר בקובץ Defs.h).
g. ניתן לבצע את בדיקת החוקיות של הצורה/קו בבנאי.
h. אתם רשאים להניח כי תוכן קובץ הקלט (אם קיים) תואם לפורמט שצוין לעיל. ומכיל לפחות צורה אחת.
i. אתם רשאים(אך אין חובה) להניח כי אורך שורה בקובץ הקלט =< 255 תווים.
j. אתם רשאים להניח כי במקרה ששם קובץ הקלט מכיל תיקיה, אזי היא קיימת ויש לכם הרשאות מתאימות בשבילה.
k. עליכם לוודא שהתוכנית הופעלה עם הארגומנטים המתאימים ושפתיחת הקבצים הצליחה.
l. אתם רשאים להניח כי כל 3 נקודות בקובץ הקלט, השייכות לשתי צורות שונות אינן יושבות על ישר אחד. ובפרט:
   - אין 2 נקודות (מצורות שונות) בעלות קואורדינטות זהות.
   - אף קודקוד של צורה S לא נמצא על צלע של צורה 'S'.

5. בונוס (3 נקודות):

1. כל סטודנט שימצא שגיאה חדשה בפתרון בית הספר יקבל בונוס של 3 נקודות לציון התרגיל.
2. בכדי לקבל את הבונוס עליכם לפתוח דיון בפורום התרגיל שיכלול את:
   a. קובץ קלט שגורם לשגיאה.
   b. קובץ פלט שמכיל את פלט פתרון בית הספר שנוצר מריצתו עם קובץ הקלט.
   c. הסבר מפורט מהי השגיאה.

6. חומר עזר:

1. את פתרון הבית ספר ניתן למצוא ב:

~plabcpp/www/ex2/schoolSol.tar

2. את הקבצים המסופקים לצורך התרגיל ניתן למצוא ב:

~plabcpp/www/ex2/ex2_files.tar

3. קבצי בדיקה לדוגמא ניתן למצוא ב:

~plabcpp/www/ex2/tests_examples.tar

4. הפניית ערוץ (redirection):
   ניתן לבצע redirection של הקלט/פלט באמצעות המתודות ()rdbuf של cout/cin.
   לדוגמא, קטע הקוד הבא מבצע redirection ל-standard input מקובץ קלט:

```
#include <iostream>

#include <fstream>

int main()

{

 static std::ifstream s_inF("input1.in");  // (assuming that the file exists and we succeed opening it).

 std::cin.rdbuf(s_inF.rdbuf());

 // From this point on, each call with std::cin will return the  next input from the file input1.in
```

שימו לב שבסוף התוכנית, או הקטע בו משתמשים ב redirection, יש להפנות חזרה את cin/cout לקלט והפלט הסטנדרטי.

7. הגשה:

1. עליכם להגיש קובץ tar בשם ex2.tar המכיל את כל הקבצים הנדרשים לקימפול התוכנית שלכם ואת הקבצים הבאים:
   - README
   - קובץ Makefile התומך לפחות בפקודות הבאות:
     - make tar - יצירת קובץ tar בשם ex2.tar, המכיל רק את הקבצים שצריך להגיש בתרגיל.
     - make clean - ניקוי כל הקבצים שנוצרו באמצעות פקודות ה-makefile.
     - make - קימפול[3] ויצירת התוכנית Shapes.
   - extension.pdf - רק במקרה שההגשה היא הגשה מאושרת באיחור (מכיל את האישורים הרלוונטים להארכה).

[3] כלומר עם הדגל NDEBUG.

- שימו לב! - אל 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 ex2.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:

~plabcpp/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/ex3/presubmit_ex2

בהצלחה!