שפת C++ – תרגיל 2

תכנות מונחה עצמים, ירושה, פולימורפיזם, תכנון, STL

<u>תאריך הגשה:</u> יום חמישי 11.1.17 עד שעה 23:55

*** הגשה מאוחרת (בהפחתת 10 נקודות): יום רביעי 18.1.17 עד שעה 23:55**

<u>תאריך ההגשה של הבוחן:</u> יום חמישי 11.1.17 עד שעה 23:55

הנחיות

הנחיות כלליות לקורס:

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 כלשהו 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 להדר (to compile בלעז) עם הדגלים Wall -Wextra- ולוודא שהתכנה מתהדרת ללא אזהרות, **תכנה שמתהדרת עם אזהרות תגרור הורדה בציון התרגיל.** למשל, בכדי ליצור תכנה מקובץ מקור בשם ex1.cpp יש להריץ את הפקודה:

```
g++ -std=c++17 -Wextra -Wall ex1.cpp -o ex1
```

6. עליכם לוודא שהתרגילים שלכם תקינים ועומדים בכל דרישות ההידור והריצה במחשבי בית הספר מבוססי מעבדי bit-64 (מחשבי האקווריום, 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 - 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 (* 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. **הקפידו מאוד** 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**).

הנחיות ספציפיות לתרגיל זה:

1. מומלץ מאוד לקרוא ולהבין כל התרגיל לפני שאתם מתחילים לתכנן את מימוש התרגיל. עיצוב הקוד (code design) שלכם הוא חלק מהדברים שייבדקו בתרגיל.
2. בתרגיל זה אתם רשאים להוסיף קבצים נוספים.
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. תרגיל זה יבוצע בזוגות. בנוסף עליכם לעבוד עם חשבון git שנספק לכם.
5. הסבר על חשבון הגיט ועבודה איתו יועברו בשיעור ונמצאים במודל.

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.

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

שח-מט

בתרגיל זה תידרשו לממש מערכת למשחק שחמט (בין שני שחקנים אנושיים). מי שלא מכיר את חוקי המשחק, מוזמן לקרוא קודם את עמוד מספר 7, או להסתכל פה או פה ולהכיר את מבנה הלוח, הכלים, והחוקים.

חוקי המשחק:

את משחק השחמט, משחקים על לוח משבצות בגודל 8X8, את צורת הלוח והסידור ההתחלתי – ראו בדוגמה בעמודים בסוף הבא.

תנועות הכלים

- צריח - קדימה, אחורה או לצדדים.
- פרש – שתי משבצות לאחד הכיוונים ומשבצת אחת לכיוון מאונך לו.
- רץ – אך ורק באלכסון.
- מלך – משבצת אחת לאיזה כיוון שירצה (קדימה, אחורה, הצידה או באלכסון).
- מלכה – קדימה, אחורה, הצידה או באלכסון – בלי הגבלה על מספר הצעדים.
- חייל – קדימה בלבד משבצת אחת בכל מהלך (אפשרות לשניים בהתחלה, אם אין כלי בינו למשבצת אליה הוא רוצה להגיע) ואכילה בצעד אלכסוני בודד קדימה בלבד. (קדימה – לכיוון ההפוך בו השחקן מתחיל, במספור עולה לחייל הלבן ומספור יורד לחייל השחור)
- כל הכלים (למעט החייל) אוכלים באותו כיוון שהם נעים.
- הצריח, הרץ והמלכה יכולים לנוע כמה משבצות שירצו, כל עוד הם לא מדלגים על כלים אחרים.
- משתמש לא יכול לסיים מהלך כשהמשתמש השני יכול לאכול את המלך שלו.

מהלכים מיוחדים

- הצרחה (הזזת המלך שתי משבצות לכיוון הצריח, והזזת הצריח שבכיוון אליו זז המלך למשבצת שבין המיקום הקודם והחדש של המלך, צעד זה ניתן לביצוע רק אם המלך והצריח לא זזו עדיין במהלך המשחק, אין כלי שמפריד ביניהם, והיריב לא מאיים על אף אחת מהמשבצות: לא זו שהמלך יצא ממנה, לא על היעד של המלך, ולא על המשבצת אליה נלקח הצריח)
- מהלך זה יתקבל ע"י קבלת קלט באופן הבא:
  ס-ס עבור הצרחה קטנה , ס-ס-ס עבור הצרחה גדולה.
- הפיכת חייל למלכה כשהוא מגיע לסוף הלוח.
- קיימים במשחק מהלכים נוספים אותם אינכם חייבים לממש (מימוש שלהם יזכה בנקודות בונוס, ראו הערה בעמוד הקודם)
- לגבי הסידור ההתחלתי והצורה של הלוח, ראו דוגמה בעמוד הבא.

מהלך התכנית

1. שני השחקנים יתבקשו להכניס את שמותיהם:
   1.1 להדפיס "Enter white player name:\n", לקרוא את שם השחקן הראשון (הלבן כמובן).
   1.2 "Enter black player name:\n" ולקרוא את שמו של השני.
2. מכאן בכל שלב התכנית:
   2.1 ~~תנקה את המסך~~ (הדפיסו "\33[2J"", הסבר נוסף על תווי בקרה בהמשך).
   2.2 תצייר את הלוח - ~~לא לנקות את המסך בין ציור לציור.~~

2.3. אם השחקן שעכשיו תורו נמצא תחת שח על התכנית להזהיר אותו ("Check!\n", בלבן על רקע אדום)

2.4. תבקש מהשחקן שזה תורו לבצע מהלך
(הדפיסו "<player name>: Please enter your move:\n")
מהלך מורכב מהמיקום של הכלי אותו המשתמש רוצה להזיז, ומהיעד שאליו הוא רוצה לקחת את הכלי (במבנה אות מספר אות מספר, בלי שום רווח באמצע, למשל כדי להזיז פרש שנמצא במשבצת D3 למשבצת C5, המשתמש יכניס D3C5)

3. עם סיום המשחק (מט, נניח שהמשחק לא יכול להסתיים בפט) על התכנית לצייר את הלוח, להדפיס: "<player name> won!\n" ולצאת.

אופן ציור הלוח

- המעטפת:
  - בכל צד של הלוח יש לצייר את הקואורדינטות המתאימות לו:
    - סדרת האותיות A-H בשורה העליונה והתחתונה (כשA משמאל)
    - סדרת המספרים 1-8 מימין ומשמאל (כש1 למטה).
  - יש להוסיף שורה ריקה בין האותיות ללוח, ורווח בין המספרים ללוח.
  - אין צורך לנקות את המסך בין ציור של הלוח לציור של הלוח.

- המשבצות:
  - יש לתת לכל משבצת צבע רקע:
    - תכלת (המשבצת 1H וכל משבצת אחרת שאפשר להגיע ל1H ממנה בתנועות אלכסוניות בלבד) או ירוק (המשבצת 1A וכל משבצת אחרת שאפשר להגיע ל1A ממנה בתנועות אלכסוניות בלבד, בסופו של דבר כל הלוח צריך להיות צבוע).
    - משבצות ריקות יש לצייר באמצעות רווח.

הכלים עצמם:

הכלים מוצגים ע"י תווי יוניקוד מיוחדים בהתאם לטבלה[1]:

מספר ביוניקוד	כלי
265F	חייל
265C	צריח
265E	סוס
265D	רץ
265A	מלך
265B	מלכה

כדי להדפיס את התו, יש לעשות לו "הברחה" (escaping), ע"י הוספת \u ואז המספר, למשל כדי לצייר חייל לבן על רקע אדום יש להריץ את השורה:

```
std::cout << "\33[37;41m\u2659\33[0m";
```

משבצות ריקות יש להדפיס באמצעות רווח.

אופן הצביעה:

כדי לשלוט על צבע הטקסט או צבע הרקע שלו, יש להדפיס את תו הבקרה 33 (escape בasciia), אחריו פתיחת סוגריים מרובעים, ואז את קודי הצבעים, מופרדים בנקודה-פסיק, אחרי קודי הצבעים יש להדפיס את האות m.

רשימת קודי צבעים (אלו שרלוונטיים לתרגיל):

- 0 – איפוס, חזרה לצבע רגיל
- 37 – טקסט לבן
- 30 – טקסט שחור
- 42 – רקע ירוק
- 46 – רקע תכלת
- 41 – רקע אדום

למשל: כדי להדפיס 'illegal move\n' בצבע לבן על רקע אדום ולהחזיר לצבעים הרגילים (בשביל ההדפסות הבאות וירידת השורה) יש להריץ את השורה (פקודות הבקרה מסומנות בכחול)

```
std::cout<<"\33[37;41millegal move\33[0m"<<std::endl;
```

למען הסדר נדרוש הדפסה של כל פקודות הבקרה (כולל החזרה לצבע הרגיל) בכל פעם שיש שינוי בצבע או בצבע הרקע של ההדפסה. למשל שירידות שורה תודפסנה בצבע הרגיל.

דוגמאות נוספות:

- כדי להציג רווח ירוק יש לשלוח להדפסה את המחרוזת "\33[0;42m \33[0m"
- כדי לצייר מלכה שחורה על רקע תכלת יש לשלוח להדפסה "\33[30;46m\u265B\33[0m"
- כדי להדפיס את שניהם אחד אחרי השני ואז ירידת שורה יש להדפיס "\33[0;42m \33[30m\33[0;46m\u265B\33[0m\n")

---

[1] תווים אלו הם רשמית של השחור, אולם התווים של הלבן נראים פחות טוב (לדעתנו) אז ביקשנו ממכם להשתמש באלו ולצבוע אותם בצבע הרלוונטי.

הערות:

1. כמה הנחות מקלות:
   1.1. אין צורך לזהות את מצבי התיקו השונים.
   1.2. אין צורך לזהות הכאה כדרך הילוכו - הסבר כאן.
   1.3. נניח כי מט יכול להימנע רק בהזזה של המלך (ולא ע"י כלי אחר שיאכל את הכלי המאיים או יחצוץ בינו למלך)
   1.4. עם זאת נקודות בונוס תינתנה למי שיממש אחד מהמהלכים הנ"ל (נא לציין זאת ואת דרך המימוש בקובץ README, אם אתם עושים זאת, מי שרוצה לזהות מצב תיקו – מספיק לזהות אחד מהם).
   1.5. למעט מקרים אלו יש לממש את כלל המהלכים במשחק (ראו פירוט בעמוד הבא)

2. בכל מקרה של ~~קלט לא תקין~~ הניחו כי הקלט תקין או בקשת מהלך לא חוקי (תנועה לא תקינה עבור סוג הכלי, מהלך שחושף את המלך או משאיר אותו חשוף, ניסיון לשחק עם כלי של היריב, לאכול כלי שלך, לצאת מהלוח או להעביר צריח, רץ, מלכה או חייל דרך כלים) על התכנית להדפיס "illegal move\n" בלבן על רקע אדום ולחזור על בקשת המהלך (כולל להזהיר את השחקן אם הוא בשח, להציג שוב את הלוח ולבקש ממנו לשחק)

3. השתדלו לתכנן את הקוד שלכם בצורה טובה, ולנצל את היכולות המונחות עצמים (oop) של השפה: כימוס (encapsulation), רב-צורתיות (polymorphism), הורשה, אופרטורים וכו'..

4. ישנן דרכים אחרות להגיע לאותה צורה של פלט על המסך (למשל: ע"י הזזה של הסמן במקום לנקות את המסך), בכל מקרה יש לתת הפלט המופיע כאן.

5. הגישו את הקוד בקובץ בשם chess.cpp (ובקבצים נוספים, בהתאם לצורך, כדי לא ליצור בלבול דרשנו הפעם שקובץ הקוד וקובץ ההרצה יתחילו באות קטנה)

דוגמה למשחק אפשרי:

הקלט והפלט של הדוגמה הנ"ל מופיעים בקובץ tar שבאתר, כדי להציג את הרצף אין ניקוי מסך בין הדפסות הלוח שבתמונה: (התמונות משמאל לימין)

<u>הערה חשובה:</u> בהדפסת הלוח בדקו האם ההדפסה שלכם תואמת לפתרון בית הספר - ההדפסה בClion וההדפסה בshell שונות (כתמונת ראי)

```
Enter white player name:
Moshe
Enter black player name:
John
  ABCDEFGH

8 ♜♞♝♛♚♝♞♜ 8
7 ♟♟♟♟♟♟♟♟ 7
6         6
5         5
4         4
3         3
2 ♙♙♙♙♙♙♙♙ 2
1 ♖♘♗♕♔♗♘♖ 1

  ABCDEFGH

Moshe: Please enter your move:
E2E4
  ABCDEFGH

8 ♜♞♝♛♚♝♞♜ 8
7 ♟♟♟♟♟♟♟♟ 7
6         6
5         5
4       ♙ 4
3         3
2 ♙♙♙♙   ♙♙♙ 2
1 ♖♘♗♕♔♗♘♖ 1

  ABCDEFGH

John: Please enter your move:
B8C6
  ABCDEFGH

8 ♜   ♝♛♚♝♞♜ 8
7 ♟♟♟♟♟♟♟♟ 7
6   ♞     6
5         5
4       ♙ 4
3         3
2 ♙♙♙♙   ♙♙♙ 2
1 ♖♘♗♕♔♗♘♖ 1

  ABCDEFGH
```

```
Moshe: Please enter your move:
D1H5
  ABCDEFGH

8 ♜   ♝♛♚♝♞♜ 8
7 ♟♟♟♟♟♟♟♟ 7
6   ♞     6
5         ♕ 5
4       ♙ 4
3         3
2 ♙♙♙♙   ♙♙♙ 2
1 ♖♘♗   ♔♗♘♖ 1

  ABCDEFGH

John: Please enter your move:
C6B4
  ABCDEFGH

8 ♜   ♝♛♚♝♞♜ 8
7 ♟♟♟♟♟♟♟♟ 7
6         6
5         ♕ 5
4 ♞     ♙   4
3         3
2 ♙♙♙♙   ♙♙♙ 2
1 ♖♘♗   ♔♗♘♖ 1

  ABCDEFGH

Moshe: Please enter your move:
F1C4
  ABCDEFGH

8 ♜   ♝♛♚♝♞♜ 8
7 ♟♟♟♟♟♟♟♟ 7
6         6
5         ♕ 5
4 ♞♗     ♙   4
3         3
2 ♙♙♙♙   ♙♙♙ 2
1 ♖♘♗   ♔   ♘♖ 1

  ABCDEFGH
```

```
John: Please enter your move:
B4C2
  ABCDEFGH

8 ♜ ♝♛♚♝♞♜ 8
7 ♟♟♟♟♟♟♟♟ 7
6         6
5        ♕ 5
4   ♗ ♟   4
3         3
2 ♟♟♞♟ ♟♟♟ 2
1 ♜♞♝ ♔ ♞♜ 1

  ABCDEFGH

Check!
Moshe: Please enter your move:
E1D1
  ABCDEFGH

8 ♜ ♝♛♚♝♞♜ 8
7 ♟♟♟♟♟♟♟♟ 7
6         6
5        ♕ 5
4   ♗ ♟   4
3         3
2 ♟♟♞♟ ♟♟♟ 2
1 ♜♞♝♔   ♞♜ 1

  ABCDEFGH

John: Please enter your move:
C2A1
  ABCDEFGH

8 ♜ ♝♛♚♝♞♜ 8
7 ♟♟♟♟♟♟♟♟ 7
6         6
5        ♕ 5
4   ♗ ♟   4
3         3
2 ♟♟ ♟ ♟♟♟ 2
1 ♞♞♝♔   ♞♜ 1

  ABCDEFGH
```

```
Moshe: Please enter your move:
H5F6
illegal move
  ABCDEFGH

8 ♜ ♝♛♚♝♞♜ 8
7 ♟♟♟♟♟♟♟♟ 7
6         6
5        ♕ 5
4   ♗ ♟   4
3         3
2 ♟♟ ♟ ♟♟♟ 2
1 ♞♞♝♔   ♞♜ 1

  ABCDEFGH

Moshe: Please enter your move:
H5F7
  ABCDEFGH

8 ♜ ♝♛♚♝♞♜ 8
7 ♟♟♟♟♟♕♟♟ 7
6         6
5         5
4   ♗ ♟   4
3         3
2 ♟♟ ♟ ♟♟♟ 2
1 ♞♞♝♔   ♞♜ 1

  ABCDEFGH

Moshe won!
```

makefile

על קובץ המakefile שלכם לתמוך בפקודות הבאות שמייצרות את קובץ ההרצה chess.

- make chess
- make all

וגם בפקודה:

- make clean

למחיקת קבצי ההרצה, הספריות וקבצים נוספים שיוצרו במהלך הבנייה (קבצי o. למשל)

- make tar

ליצירת קובץ הtar להגשה.

חומר עזר:

- קבצי הבדיקה ניתן למצוא ב:

~labcpp/www/ex2/ex2_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 לפלטים של תלמידים אחרים, או עם הפלט שנוצר כשאתם נותנים את הקלט הזה לקובץ הריצה של פתרון בית הספר.

הגשה

1. עליכם להגיש קובץ בשם ex2.tar המכיל:
   1.1. את הקובץ chess.cpp
   1.2. יתר הקבצים המכילים את הקוד שלכם
   1.3. קובץ Makefile על פי הפירוט בסעיף למעלה
   1.4. קובץ README המכיל ב2 שורות הראשונות שלו את הusernames שלכם בלבד (שורה 1 = username1,שורה 2 = username2) -בשורה 3 על הקובץ להכיל קישור לgit reposetory שלכם (ראו פירוט למטה). אם מימשתם את אחד מהבונוסים צרפו תיאור קטן בשורה הרביעית והלאה.
   1.5. extension.pdf - רק במקרה שההגשה היא הגשה באיחור.
2. ניתן ליצור קובץ tar כנדרש ע"י הפקודה:

tar cvf ex2.tar <list of files to submit>

3. שימו לב - ההגשה היינה רק מאחד המשתמשים! תרגיל שיוגש מ2 המשתמשים ייקנס בניקוד.
4. לפני ההגשה, פתחו את הקובץ ex2.tar בתיקיה נפרדת וודאו שהקבצים מתהדרים ללא שגיאות וללא אזהרות.
5. מומלץ מאוד גם להריץ בדיקות אוטומטיות ובדיקות שכתבתם על הקוד אותו אתם עומדים להגיש.
6. 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, בדיקה זו מהווה רק חלק מבדיקות ה- codingStyle)

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

```
~labcpp/www/ex2/presubmit_ex2
```

Github

1. עליכם לעבוד עם github באופן רציף. כלומר - אנחנו מצפים לראות עדכונים בreposetory שלכם על מנת לוודא שאכן אתם עומדים בדרישות המשימה.
2. שימוש רציף - לא העלתם פשוט את כל הקבצים ביום האחרון לgit ושלחתם למודל אלא עבודה נורמטיבית על ex וככלל העלאת כל השינויים שאתם עובדים עליהם.
3. בסוף התרגיל יהיה עליכם לשתף את הreposetory עם חשבון הקורס (שאותו נפרסם בפורום התרגיל בהמשך) על מנת שנוכל לראות את העבודה שלכם
4. קיימת מצגת במודל המסבירה את מהלך העבודה עם git

בהצלחה!