תכנות מכוון עצמים ושפת C++ - סמ' ב' תשע"ז

תרגיל 3

בתרגיל זה עליכם לעשות שימוש בהורשה ולאפשר הרצה של משחק עם "אלגוריתם" שחקן.
הממדים של לוח המשחק לא משתנים.

[1]
שימוש בהורשה למימוש שחקן מקלדת, שחקן קובץ ושחקן "אלגוריתם" (שעליו נדבר בהמשך).
המטרה היא שקוד משותף ייכתב רק פעם אחת.

מנהל המשחק יעבוד מול "שחקן" כללי כאשר רק בתחילת המשחק חשוב לדעת איזה שחקן ליצור אבל במהלך המשחק מנהל המשחק יהיה אדיש לשאלה מול איזה סוגי שחקנים הוא עובד.

העבודה של מנהל המשחק מול "שחקן" כללי תהיה באופן הבא:
יש ליצור מחלקת בסיס אבסטרקטית בשם BoardData שתציג את הממשק הבא:

```
class BoardData {
public:
  enum {rows = 13, cols = 13};
  virtual ~BoardData() { }
  virtual char charAt(int x, int y) const = 0;
};
```

אסור להוסיף למחלקה זו פונקציות או משתנים נוספים כלשהם.
כל הקורדינטות שנמסרות בממשקים שמתוארים כאן מתחילות תמיד מ-1,1.
הפונ' charAt תחזיר את ה-char שמיוצג בקורדינטה שנשלחה, אך כאשר מדובר בכלי של שחקן יריב הפונקציה תחזיר את התו # (סולמית) שיסמן שבמשבצת ישנו כלי של שחקן יריב.

בנוסף יש ליצור מחלקת בסיס אבסטרקטית בשם AbstractPlayer שתציג את הממשק הבא:

```
class AbstractPlayer {
public:
  virtual ~AbstractPlayer() { }
  /* player: 1 for 1-2-3 player, 2 for 7-8-9 */
  virtual void setPlayer(int player) = 0;
  virtual void init(const BoardData& board) = 0;
  virtual GameMove play(const GameMove& opponentsMove) = 0;
  virtual string getName() const = 0;
};
```

אסור להוסיף למחלקה זו פונקציות או משתנים נוספים כלשהם.
מנהל המשחק יישלח לפונקציה play את המהלך האחרון שביצע היריב. בפתיחת המשחק יישלח מהלך "סרק" מוגדר מראש: מקור 0,0 ליעד 0,0.

גם החתימה של המחלקה GameMove נתונה מראש:

```
struct GameMove {
  const int from_x, from_y;
  const int to_x, to_y;
  GameMove(int x1, int y1, int x2, int y2): from_x(x1), from_y(y1), to_x(x2), to_y(y2) { }
};
```

[2]

<u>שחקן "אלגוריתם"</u>

יש לממש אלגוריתם חכם אחד או שניים - לבחירתכם - באמצעות מחלקה שתירש מ-AbstractPlayer.
המחלקה תדע להחזיר מהלכים הגיוניים במטרה לנצח במשחק תוך ניסיון לעקוב אחר מהלך המשחק.
המחלקה תדע לשחק עבור שחקן A ו-B בהתאם למה שיימסרו לה בתחילת המשחק בפונקציה setPlayer.

<u>רישום אוטומטי של האלגוריתם</u>:

בקובץ ה-cpp של המחלקה תופיע השורה הבאה כשורה של יצירת משתנה גלובלי:

```
AlgorithmRegistration algo_StudentId("StudentId", []{return new MyAlgo()});
```

את החלק של StudentId יש להחליף בת"ז של אחד המגישים.
הפרמטר השני לבנאי של AlgorithmRegistration הוא פונקציית למבדא ונדבר על זה בשיעור.
את ה-return שבלמבדא יש להחליף ב-new למחלקת האלגוריתם שלכם. יש לדאוג ששמה של מחלקת האלגוריתם יהיה Algo_StudentId וכמובן יש להחליף את החלק של StudentId במס' ת"ז.
במידה ומגישים שני אלגוריתמים יש ליצור בכל אחד את השורה הנ"ל לצורך רישום האלגוריתם ולהשתמש בת"ז של שני חברי הצוות או בת"ז אחת עם סיומת a ו-b על-מנת ליצור שמות שונים לאובייקטים הגלובליים מסוג AlgorithmRegistration.
האחריות ליצור את המחלקה AlgorithmRegistration היא עליכם (אבל אולי ניגע בה בתירגולים).

כאשר משחקים משחק עם שחקנים מבוססי אלגוריתמים המשחק יהיה בהכרח בין שני אלגוריתמים - גם אם נרשמה רק מחלקת אלגוריתם אחת (אותה המחלקה תשרת את שני השחקנים באמצעות שני אובייקטים שונים). במידה ונרשמו שתי מחלקות או יותר, הראשונה שנרשמה תייצג את השחקן A והשניה את B (אנחנו לא ממש שולטים על סדר הרישום שלהן וזה בסדר).

יתכן שבבדיקה אנחנו נריץ תחרות בין האלגוריתמים של תלמידי הכיתה ונודיע על המנצחים. אלגוריתמים שחושבים יותר מידי או עפים או מתנהגים באופן לא תקין שמעיף את המתחרה יוסרו מהתחרות ועלולים להפסיד נקודות. לא תהיה הפחתת ניקוד בגין מקום נמוך בתחרות כל עוד נכתב אלגוריתם סביר ועובד.

על התרגיל לתמוך בכל הדרישות של תרגילים 1-2 ובנוסף לאפשר הרצה של משחק אוטומטי עם אלגוריתמים מתחרים.

הבחירה במשחק עם אלגוריתמים מתחרים תהיה באמצעות פרמטר command-line חדש:

-moves a

כאשר a מסמן algorithm

במידה ונבחרה אפשרות זו - מנהל המשחק יריץ תחרות בין שני אלגוריתמים שרשמו את עצמם על-פני כל קבצי הלוחות שנמצאים ב-path או על-פני מספר מסכים רנדומליים כפי שייקבע בפרמטר command-line שיוגדר להלן.

<u>הרצת המשחק</u>

כאשר ישנו קלט של מהלכים מקובץ או ריצה של אלגוריתמים אוטומטיים לא יוצגו תפריטים.
התפריטים יעבדו כמו בתרגיל 1 - עבור משחק מקלדת בלבד.

המשחק יורץ באמצעות פרמטרים ב-command-line.
הפרמטרים יכולים להגיע בסדר כלשהו:

-board

r = random board (default)

after ‘r’ may appear space and then a number indicating number of random boards to run - relevant only for algorithm run

or f = file board

-moves

k = keyboard moves (default)

or f = file moves

or a = algorithms

-path <relative or absolute path with or without slash at the end>

path where files should be looked for (default: working directory)

-quiet

do not print board during game, only final results

this option is relevant only with ***-moves f*** or ***-moves a*** otherwise ignored

-delay <delay in ms>

delay in ms between player’s move, default is 20 ms

this option is ignored if ***-quiet*** is active

-save

save moves (and in case of random boards - also boards) to files

this option behaves the same as the “save” option in the menus and is relevant only with ***-moves a*** otherwise ignored. Default (if not set) is <u>not</u> to save

דוגמאות חדשות להרצה (מעבר לדוגמאות מתרגיל 2 שעדיין רלבנטיות):

[1]

FlagsGame.exe -board r 10 -moves a -quiet -delay 100

התוצאה:

הרצה של 10 לוחות משחק רנדומליים על אלגוריתמים אוטומטיים ללא תצוגה של המשחק במסך (יודפסו רק תוצאות סופיות של המשחקים) וללא תפריט. הפרמטר delay לא רלבנטי ומתעלמים ממנו - במצב זה אין בכלל delay.

[2]

FlagsGame.exe -path "C:/Projects/CPP/Ex3/" -board f -quiet -delay 100

התוצאה:

הרצה של לוחות משחק מה-path שסופק. המהלכים ייקלטו מהמקלדת. תהיה תצוגה של המשחק במסך (מכיון שהמהלכים נקלטים מהמקלדת). הפרמטר delay רלבנטי וייקבע ל-100 ms.
תפריט משתמש יעבוד כמו בתרגיל 1.

[3]

FlagsGame.exe -path "C:/Projects/CPP/Ex2/" -board f -moves a -delay 50

התוצאה:

הרצה של לוחות מקבצים ומהלכים מאלגוריתמים. תהיה תצוגה של המשחק במסך עם delay של 50 ms.
ללא תפריט משתמש.

[4]

FlagsGame.exe -path "C:/Projects/CPP/Ex2/" -board f -moves a -quiet -save

התוצאה:

הרצה של לוחות מקבצים ומהלכים מאלגוריתמים. ללא תצוגה של המשחק וללא תפריט. שמירה של המהלכים לקבצים, אין צורך לשמור את הלוחות מכיון שהם ממילא מגיעים מקובץ בהרצה זו.

<u>מעבר בין משחקונים</u>

המעבר בסיום כל משחקון למשחקון הבא יהיה אוטומטי עם delay באורך 50 פעמים ה-delay שבין מהלכים ולאחריו יתחיל משחקון חדש עם מסך חדש (מקובץ חדש או רנדומלי בהתאם לעניין).

כאשר המשחק רץ במוד quiet בסיום כל משחקון יודפסו למסך הפרטים הבאים בפורמט המדוייק הבא:

Game cycle: <number>

Num moves: <total number of moves of A + B including “no op” moves>

Winner: <A or B or NONE>

<u>הודעת הסיום</u>

במצב של ריצה אוטומטית של אלגוריתמים הודעת הסיום תהיה בפורמט:

Game Summary

<Algorithm name via `AbstractPlayer::getName()`> for A points: <points>

<Algorithm name via `AbstractPlayer::getName()`> for B points: <points>