Tic-Tac-Toe Tournament

תרגיל 2 בקורס מבוא לתכנות מונחה עצמים

תאריך הגשה: 16.11 23:55

בתרגיל זה נממש טורניר של משחקי איקס עיגול.

בגרסה שלנו הלוח יהיה בגודל $n \times n$, וסיבוב יחיד של משחק נגמר כאשר יש שחקן מנצח או כאשר לא נותרו משבצות ריקות בלוח.

שחקן מוגדר כמנצח אם הוא השיג על הלוח רצף של k משבצות מסומנות בסימן שלו (X או O). הרצף יכול להיות מאונך, מאוזן או אלכסוני.

כאשר n ו k הם משתנים עם ערכים שיבחר המשתמש, ועם הערכים הדיפולטיביים 4 ו-3 בהתאמה.

נשים לב, שבשביל שיהיה משמעות למשחק, נדרוש $2 \leq k \leq n$.

בתרגיל יהיו שני סוגי שחקנים, שחקן אנושי שישחק את תורו דרך ה-$System.in$; ושחקן אוטומטי, שיפעל באופן אוטומטי לפי אסטרטגיה מוגדרת.

מהלך המשחק

1. בתחילת המשחק יבחר המשתמש שני סוגי שחקנים, השחקנים האלה יכולים להיות מאותו סוג - שניהם אוטומטיים/אנושיים, או מסוגים שונים - אחד אוטומטי ואחד אנושי.
2. השחקנים שנבחרו יערכו טורניר המורכב ממספר סיבובים שיגדיר המשתמש.
   בכל סיבוב יחליפו השחקנים תפקיד כך שישחקו לסירוגין בסימנים איקס ועיגול, ולוח התוצאות יציג כמה סיבובים ניצח שחקן 1, כמה סיבובים ניצח שחקן 2 וכמה סיבובים הסתיימו בתיקו.
3. כמו כן, יבחר המשתמש משתנה שלישי שמגדיר האם הלוח ירונדר על המסך או לא. האפשרות לא להציג את הלוח שימושית במיוחד כשרוצים להריץ טורניר של מספר סיבובים בין שחקנים אוטומטיים.

שורת ההרצה

משתמש שרוצה להריץ את המשחק, יקליד את הפקודה הבאה:

Java Tournament [round count] [size] [win_streak] [render target: console/none]
[player: human/whatever/clever/genius]×2

לדוגמא, הפקודה הבאה תריץ טורניר של $10,000$ סיבובים על לוח בגודל 4×4, ורצף ניצחון של 3, בין השחקנים $Mr.\ Whatever$ ו $Mr.\ Clever$ (ר׳ פירוט למטה לגבי סוגי השחקנים), ולא תדפיס את הלוח על המסך:

Java Tournament 10000 4 3 none whatever clever

כדי לשחק שלושה משחקים בין שחקן אוטומטי למשתמש, הפקודה היא:

Java Tournament 3 4 3 console whatever human

- במקרה שבו תבחרו בערך none עבור רינדור הלוח ותגדירו גם שחקן אנושי, תקבלו בדיוק את מה שביקשתם: הלוח לא יודפס על המסך אבל כשיגיע תורו של המשתמש האנושי הוא עדיין יתבקש לבחור משבצת על לוח המשחק.
- במידה ויש שגיאת הקלדה ("$Typo$") באחד משמות השחקנים, תודפס על המסך הודעה **אינפורמטיבית** על כך בניסוח קבוע, וריצת התוכנית תסתיים ללא הרצת טורניר.

  לדוגמא, עבור הפקודה Java Tournament 3 4 3 console whtever human

  יודפס:

Choose a player, and start again

The players: [human, clever, whatever, genius]

  שימו לב להקפיד על הניסוח המדויק.
- קלט שבשם השחקן מכיל אותיות גדולות ($Capitals$) נחשב תקין.

  כלומר, הקלטים $Clever$, $CLEVER$ ו $clever$ שקולים ותקינים.
- ניתן להניח שגודל הלוח ורצף הניצחון הם ספרות בטווח 2-9, אך צריך לבדוק שערכו עומד בהוראות המשחק.
- תניחו ששורת ההרצה מכילה את כל 6 הארגומנטים שנדרשים להרצת טורניר.
- עבור שאר הארגומנטים, ניתן להניח קלט תקין.

<u>נציג את ה API של המשחק המורכב משני ממשקים, שלוש מחלקות משחק, שתי מחלקות רינדור, ארבע מחלקות שחקן, שתי מחלקות מפעל ומ $Enum$:</u>

- ממשקים: $Player$, $Renderer$
- מחלקות:
  1. $ConsoleRenderer$ - המימוש שלה נמצא ב Moodle.
  2. $VoidRenderer$
  3. $Board$
  4. $Game$
  5. $HumanPlayer$
  6. $CleverPlayer$
  7. $WhateverPlayer$
  8. $GeniusPlayer$

9. $Tournament$
10. $PlayerFactory$
11. $RendererFactory$

★ $enum\ Mark$

- ❖ כל המתודות שנציג כחלק מה API הם בעלות מגדיר נראות פומבי ($public$).
- ❖ הרגישו חופשיים להגדיר כמה פונקציות עזר פרטיות ושדות פרטיים שתצטרכו.

הממשק $Player$ **:**

ממשק שמייצג לוגיקה או אסטרטגיה מסוימות לביצוע תור במשחק בהינתן לוח מסוים.

מתודה	משמעות
$void\ playTurn(Board\ board,\ Mark\ mark)$	מבצעת את האסטרטגיה של השחקן.

<u>המחלקות שמממשות את</u> $Player$

ארבעת המחלקות $CleverPlayer,\ WhateverPlayer,\ GeniusPlayer$ ו $HumanPlayer$ יממשו את $Player$.

תפקידי המחלקה $HumanPlayer$ **:**

המחלקה מייצגת שחקן אנושי. האחריות היחידה של מחלקה זו היא בקשת קלט מהמשתמש. שימו לב שמחלקה זו לא תלויה במשחק מסוים; מופע אחד של שחקן יכול לקחת חלק במשחק אחד או בכמה, ועם סימונים שונים. כתוצאה מכך, הבנאי של $HumanPlayer$ לא דורש שום קלט, היות ולא נדרש אף מידע לגבי המשחק או הלוח כדי לאתחל את השחקן.

רשימת המתודות של המחלקה:

מתודה	משמעות
$HumanPlayer()$	בנאי, מגדיר שחקן חדש
$void\ playTurn(Board\ board,\ Mark\ mark)$	מבקשת קואורדינטות מהמשתמש וממקמת את הסימן במידה והקואורדינטות "טובות" - תקינות ולא תפוסות; במקרה שלא, היא

	תבקש קלט נוסף. הודעת הדפסה: $"Player" + mark + ", type\ coordinates:"$ $"Invalid\ coordinates,\ type\ again:"$ הגדרת קואורדינטות חוקיות: טווח שורות ועמודות: $[0, Board.Size - 1]$ דוגמא לקלט חוקי: "31"≡ שורה רביעית, עמודה שניה. הערה: ניתן להניח שהקלט מתקבל בצורה תקינה (מספרים טבעיים), אך לא ניתן להניח שהערכים תקינים.

הגדרת השחקנים האוטומטיים

עבור כל אחד מהשחקנים, עליכם לבנות אסטרטגיה שתתנהג באופן הבא:

- המהלכים של $Mr.\ Whatever$ אקראיים.
- $Mrs.\ Clever$ תנצח את $Mr.\ Whatever$ ברוב המוחלט של המשחקים.
- $Ms.\ Genius$ תשחק טוב יותר מ $Mrs.\ Clever$.

המחלקה $WhateverPlayer$

בחירה של משבצת אקראית על הלוח (אם אתם לא זוכרים איך בוחרים מספר אקראי, חזרו לשיעור 1.2 בקמפוס).

המחלקה $CleverPlayer$

לבחירתכם: אסטרטגיה שחכמה יותר מהשחקן $WhateverPlayer$ ומנצחת אותו ברוב הפעמים.

בידקו את הצלחתו על ידי הרצת טורניר בין $10,000$ משחקים בין השחקן הנוכחי לבין $WhateverPlayer$**, על לוח ורצף ניצחון בגדלים הדיפולטיביים.** על המימוש שלכם ל $CleverPlayer$ לנצח לפחות ב 55% מהמשחקים.

המחלקה $GeniusPlayer$

אסטרטגיה שמנצחת את השחקן החכם ברוב הפעמים.

גם כאן, עליכם לוודא את הצלחתו באופן הבא, בטורניר בין $10,000$ משחקים, **על לוח ורצף ניצחון בגדלים הדיפולטיביים**:

1. בין השחקן הנוכחי לבין $WhateverPlayer$**.** על המימוש שלכם ל $GeniusPlayer$ לנצח לפחות ב 55% מהמשחקים.
2. בין השחקן הנוכחי לבין $CleverPlayer$**.** על המימוש שלכם ל $GeniusPlayer$ לנצח לפחות ב 55% מהמשחקים.

הממשק $Renderer$ **ומחלקות הרינדור**

ממשק שמייצג צורה להצגת מהלך של משחקי איקס עיגול על המסך.

המחלקה $ConsoleRenderer$ **:**

מייצגת צורה של הצגה ויזואלית של מהלך משחק על הלוח על ידי הדפסות לטרמינל.

ה API של המחלקה מכיל שתי מתודות:

1. בנאי דיפולטיבי
2. מתודה עם החתימה: $void\ renderBoard(Board\ board)$, שמקבלת לוח ומציגה אותו על המסך.

המימוש של מחלקה זו נמצא באתר הקורס (moodle), בקישור הבא. כלומר אין לשנות או לערוך אותה.

המחלקה $VoidRenderer$**:**

מימוש ריק של $Renderer$ שלמעשה לא מציג כלום על המסך, אך זה עדיין צורה של רינדור.

המחלקה $Board$ **:**

המחלקה אחראית על מצב הלוח: גודל הלוח, סימון משבצות ושמירת כל מה שסומן על הלוח.

מספור המשבצות על הלוח מתחיל ב-0.

רשימת המתודות של המחלקה:

מתודה	משמעות
$Board()$	בנאי דיפולטיבי, מגדיר לוח ריק חדש בגודל דיפולטיבי.
$Board(int\ size)$	בנאי נוסף, מגדיר לוח ריק חדש בגודל $size$.
$int\ getSize()$	מחזיר את גודל הלוח
$Mark[][]\ getBoard()$	מחזיר את המערך הדו-מימדי שמייצג את הלוח
$boolean\ putMark(Mark\ mark,\ int\ row,\ int\ col)$	תנסה לסמן את המשבצת $(row,\ col)$ ב $mark$. מחזירה אמת אם"ם סימנה את המשבצת בהצלחה (לפי כללי המשחק הסטנדרטיים).

$Mark\ \ getMark(int\ row,\ int\ col)$	תחזיר את הסימון שיש במשבצת הנתונה. במקרה של קואורדינטות לא חוקיות תחזיר $Mark.BLANK$

תפקידי המחלקה $Game$ **:**

מופע של המחלקה מייצג משחק יחיד. עליו לדעת מתי המשחק נגמר, מי היה המנצח והאם הוא הסתיים בתיקו.

מתודה	משמעות
$Game(Player\ playerX, Player\ playerO,\ Renderer\ renderer)$	בנאי, מגדיר משחק חדש, עם ערכים דיפולטיביים.
$Game(Player\ playerX, Player\ playerO,\ int\ size,\ int\ winStreak,\ Renderer\ renderer)$	בנאי נוסף, מגדיר לוח בגודל $size$, ורצף ניצחון באורך $winStreak$. במידה שהוכנס ערך לא חוקי, רצף הניצחון יוגדר להיות שווה לגודל של הלוח.
$int\ getWinStreak()$	מחזיר את אורך רצף הניצחון
$Mark\ run()$	מריצה מהלך של משחק - מתחילתו ועד סופו, ומחזירה את המנצח. המשחק מסתיים כאשר לאחד מהשחקנים יש רצף ניצחון או כאשר לא נותרו משבצות ריקות בלוח. במקרה שבו המשחק נגמר בתיקו יוחזר $Mark.BLANK$

סימון X **ו-**O **על הלוח**

$Enum$ בשם $Mark$ ייצג את הסימונים על הלוח, והוא יוגדר כך:

$$enum\ Mark\{BLANK,\ X,\ O\}$$

והוא ישב בקובץ $Mark.java$.

המחלקה $Tournament$

תפקיד המחלקה $Tournament$ הוא להריץ טורניר של מספר משחקים.

ה API **של** $Tournament$

המחלקה מבצעת סדרה של משחקי איקס עיגול (סיבובים) בין שחקנים מסוימים בממשק רינדור מסוים, כאשר: בסיבוב הראשונה, השחקן הראשון משחק איקס והשני עיגול, ובסוף כל סיבוב הם מחליפים סימנים. באופן הזה, בסבבים עם אינדקס זוגי השחקן הראשון איקס, ואילו באינקדסים האי-זוגיים זה הפוך.

בסיום כל טורניר, מודפסת על המסך התוצאה העדכנית, באופן הבא:

######### Results #########

Player 1, [player_type] won: _ rounds

Player 2, [player_type] won: _ rounds

Ties: _

לדוגמא:

######### Results #########

Player 1, clever won: 687 rounds

Player 2, whatever won: 271 rounds

Ties: 42

כדי לבצע את תפקידה, המחלקה זקוקה רק לשיטה אחת, מלבד לבנאי.

מתודה	משמעות
$Tournament(int\ rounds,\ Renderer\ renderer,\ Player[]\ players)$	בנאי
$void\ playTournament(int\ size,\ int\ winStreak,\ String[]\ playerNames)$	נקראת ע"י main, ושם קורה כל הלוגיקה של הסיבובים וקריאות למשחק. שמות השחקנים הם הסוגים שלהם, והם נמצאים בארגומרטים של שורת ההרצה $args[4],\ args[5]$
$main$	$The\ main\ method$

מפעלים

המחלקה $PlayerFactory$ אחראית על יצירת השחקנים ובכתיבתה אנו אוכפים את "עקרון האחריות הבודדת". בדומה, המפעל $RenderFactory$ יהיה אחראי על יצירת ממשק הרינדור המתאים. זאת דרך פשוטה ואלגנטית

להשאיר לעצמינו את האפשרות להוסיף בעתיד אפשרויות רינדור נוספות, כשכל מה שצריך לשנות הוא המפעל ולא שום חלק אחר בקוד.

המפעל $PlayerFactory$ אחראי למפות את המחרוזת משורת הפקודה לאובייקט שחקן ממשי. אין סיבה להשתמש ביותר משיטה אחת בנוסף לבנאי, טיפוס הקלט של שיטה זו יהיה $String$ וטיפוס הפלט שלה יהיה $Player$. לשיטה זו נקרא $buildPlayer$. באופן דומה נטפל ב $RendererFactory$ רק שהוא יצפה לקבל את אחת מהמחרוזות "$console$" or "$none$", ושם המתודה יהיה $buildRenderer$.

להלן חתימות המתודות של המפעלים:

- $public\ Renderer\ buildRenderer(String\ type,\ int\ size)$
- $public\ Player\ buildPlayer(String\ type)$

הנחיות והערות:

- אנו מצפים שתבחרו שמות אינפורמטיביים עבור המשתנים, מתודות, קבועים והודעות ההדפסה.
- אסור להוסיף פונקציות ושדות **פומביות** שלא הוזכרו בתאור המחלקות לעיל. ניתן להגדיר כמה פונקציות עזר פרטיות ושדות פרטיים שתצטרכו.
- כל הקוד בתרגיל צריך להיות מתועד היטב (מחלקות, מתודות ושדות, פומביים ופרטיים).
- כדאי להתחיל במימוש המחלקות הנצרכות להרצת משחק יחיד, לאחר מכן אנו ממליצים להריץ משחק יחיד בין שני שחקנים אנושיים וכך לבדוק את עצמכם תוך כדי עבודה - מתכנת טוב הוא מי שתופס את הבאגים מוקדם ולא מי שמתכנת בלי באגים.

עליכם להגיש קובץ Jar בשם $ex2.jar$ שמכיל את הקבצים הבאים:

1. $Game.java$
2. $Board.java$
3. $Tournament.java$
4. $Player.java$
5. $Renderer.java$
6. $PlayerFactory.java$
7. $RendererFactory.java$
8. $HumanPlayer.java$
9. $WhateverPlayer.java$
10. $CleverPlayer.java$
11. $GeniusPlayer.java$

12. $VoidRenderer.java$
13. $ConsoleRenderer.java$
14. $Mark.java$
15. $README$

קובץ ה $README$ -

- בשורה הראשונה בקובץ יופיע שם המשתמש CSE שלכם ובשורה השניה מספר תעודת הזהות.
- השורה השלישית ריקה.
- עליכם לפרט בקובץ על מהי האסטרטגיה שמימשתם עבור כל אחד מהשחקנים האוטומטיים.
- ענו על השאלה:

מה היתרון בעיצוב התוכנה באופן שבו כל אחת ממחלקות השחקנים מממשת ממשק משותף?

- הריצו טורניר בין $10,000$ משחקים בין $CleverPlayer$ לבין $WhateverPlayer$. וכתבו כמה פעמים כל אחד מהם ניצח. (בחרו לוח בגודל 4 לפחות, ורצף ניצחון 3 לפחות).
- הריצו טורניר בין $10,000$ משחקים בין $CleverPlayer$ לבין $GeniusPlayer$. וכתבו כמה פעמים כל אחד מהם ניצח. (בחרו לוח בגודל 4 לפחות, ורצף ניצחון 3 לפחות).
- הריצו טורניר בן $10,000$ משחקים בין שני שחקני $WhateverPlayer$ וכתבו כמה פעמים כל אחד מהם ניצח.

בהצלחה!!