עבודת הגשה 3

- יש להגיש את העבודה עד 2020-06-21 דרך Moodle בלבד.
  הגשה בבודדים בלבד! (אין הגשה בזוגות, שלשות וכדומה).
- במקרים של הריון, מילואים, אשפוז יש לספק אישור מתאים על כך, לפני מועד ההגשה.
- אין להעתיק: העתקה תגרום לפסילת העבודה של המקור והמעתיקים והעברה לוועדת משמעת.
- לשאלות, תפנו למייל כשבפנייה חובה לציין שם פרטי ושם משפחה והסבר מפורט של השאלה.
  svetazam+sce@gmail.com
- יש לתעד כל מחלקה וכל פונקציה בעזרת Javadoc.
- המטלה מתבססת על מטלת הגשה 2. ניתן לעדכן ולהרחיב את המחלקות הקיימות במידת הצורך.

הקדמה:

מטרת העבודה היא הוספת מקביליות לאולימפיאדה שלנו, בעזרת תהליכונים (threads) ב-JAVA.

תהליכון החיה:

יש ליצור מחלקה חדשה בשם AnimalThread הממשת את הממשק Runnable. השדות יהיו:

- participant : Animal // החיה אותה אנו מקדמים
- neededDistance: double // המרחק הנדרש לעבור מהתחלה לסיום
- startFlag: Boolean
- finishFlag: Boolean

שימו לב, שמדובר בטיפוס Boolean עם אות גדולה. דגל זה יהיה משותף בין מספר אוביקטים, ולכן אנו צריכים מצביע לאובייקט, כלומר אנחנו צריכים object שלם ולא primitive.

כמובן שיש לממש את הפונקציה run של הממשק. פונקציה זו מבצעת מספר חלקים, ומומלץ מאוד לוודא שהם מבוצעים לפי הדרישות המתאימות של wait/notify!

1. מתבצעת המתנה על עוד startFlag נמצא על מצב false, כלומר המתנה להתחלה (למשל המתנה להודעת ההזנקה). הבדיקה צריכה להיות בבלוק של synchronized מעל startFlag.
2. יש לבצע מעקב האם התהליכון שלנו נעצר: בעזרת Thread.Interrupted() או בעזרת דגל משותף isStopped או כל אפשרות תקנית אחרת – לבחירתכם.
3. בבלוק synchronized מעל participant יש לבצע הנעה של החיה קדימה (זימון לפונקציה מתאימה). אם הגענו למרחק הנדרש, נסמן ונודיע בעזרת finishFlag (הצבה + notify) ונתחיל תהליך יציאה מהתהליכון.
4. נבצע שינה למשך זמן מסוים (מספר שהוא משתנה גלובלי הניתן לשינוי על ידי המשתמש).

התחרויות השונות:

יש ליצור package חדש המכיל את התחרויות השונות ומימושים נוספים נדרשים:

מחלקת Scores:

מחלקה יחסית פשוטה שתפקידה להציג את התוצאות והזמנים. השדות הם:

- scores: Map<String, Date> // תור אליו נוסיף את המחרוזת והזמן

פונקציות שהוא מכיל הן:

- add(String name): void // מוסיף בצורה בטוחה לתור את המחרוזת עם הזמן הנוכחי
- getAll(): Map<String, Date> // מחזירה את scores

את השדה scores יש לבנות בעזרת Collections.synchronizedMap(new HashMap())

מחלקת Referee:

תפקיד מחלקה זו, הממשת את Runnable, היא להמתין לחיה מסוימת שתגיע אליה, וברגע שהחיה מגיעה יש להוסיף את השם שלה על ידי זימון ל-add מעל השדה scores.

השדות שהוא מקבל מהבנאי הם name המייצג את שם הקבוצה אותם הוא ממתין, ו-scores, מצביע למשאב משותף של התחרות מטיפוס Scores.

תהליכון התחרות:

יש ליצור מחלקה חדשה בשם TournamentThread הממשת את הממשק Runnable. שדות החובה הם:

- scores: Scores // מחזיק את התוצאות הסופיות של כל קבוצה
- startSignal: Boolean // דגל מיוחד המתחיל את כל החיות
- groups: int // מספר הקבוצות המתחרות

תפקיד תהליכון זה הוא להתחיל (להציב + notifyAll) מעל הדגל startSignal.

הוא מממש כמובן את הפונקציה run, המתחילה את כל התחרות ותפקידו לעדכן בכל כמה זמן את הנתונים המוצגים למשתמש בחלון המתאים למידע על תחרות זו (לכל קבוצה שסיימה להציג את הזמן, להשאיר ריק לכל קבוצה אחרת). כאשר הוא מגיע לסוף (כולם מילאו)
השתמשו ב-scores.getAll().

במחלקה זו בוודאות תצטרכו להוסיף שדות נוספות בשביל עבודה מול הממשק.

תחרות:

מחלקה אבסטרקטית בשם **Tournament**, שתפקידה בבנאי לקרוא לפונקציה האבסטרקטית setup המקבלת מערך דו ממדי של חיות (הרשימה מחולקת למערך של קבוצות כאשר כל קבוצה מהווה מערך של חיות) ועוד מידע הנדרש להקמת התחרות.

המחלקה מכילה שדה יחיד שהוא protected מטיפוס TournamentThread.

יש לנו פונקציה בודדת המחזירה את השדה הבודד.

תחרות שליחים:

יש ליצור מחלקה חדשה בשם CourierTournament המייצגת תחרות שליחים, היורשת מ-**Tournament**.
החלק הכבר במחלקה זו הוא יצירת כל התלות המתאימה בין החיות. בפונקציה setup נבצע:

1. נבנה בהתחלה דגל מיוחד startFlag מטיפוס Boolean ואת scores מטיפוס Scores.
2. לכל קבוצה יש לבצע (נסמן ב-n את גודל הקבוצה):
   2.1. ליצור n דגלים מסוג Boolean (שימו לב שאות גדולה) ולאתחלם ל-false.
   2.2. לכל חיה i בקבוצה יש ליצור AnimalThread חדש, המקבל את הדגל ה-$[i-1]$ בתור startFlag, והדגל ה-$[i]$ בתור finishFlag. חשבו בהתאם את neededDistance (נניח על ידי חלוקה ב-n – תפעילו דמיון).
   2.3. לחיה במקום $[0]$ כמובן אין דגל התחלה מתאים, אנחנו נקשר שם את startFlag.
   2.4. ניצור Referee חדש המקבל את הדגל האחרון משלב 2 ואת scores.
3. כעט ניצור ונשמור בשדה המיוחד הבודד את startFlag, scores ואת מספר הקבוצות.

שימו לב שכשאתם יוצרים כל משתנה ממחלקה המממשת את Runnable, יש להציב ישר ב-Thread ולבצע start.
אם הגדרתם נכון, הממתנה תתבצע אוטומטית.

תחרות רגילה:

יש ליצור מחלקה חדשה בשם RegularTournament המייצגת תחרות רגילה, כלומר כל חיה מתחרה לבדה ללא תלות, היורשת מ-**Tournament**.

כאן ההקמה ב-setup יותר פשוטה, והיא כוללת:

1. נבנה בהתחלה דגל מיוחד startFlag מטיפוס Boolean ואת scores מטיפוס Scores.
2. לכל קבוצה (יש בה רק חיה אחת) נבצע:
   2.1. ליצור דגל סיום מטיפוס Boolean.
   2.2. ליצור AnimalThread חדש לחיה הבודדת, המקבלת את startFlag ודגל הסיום.
   2.3. ניצור Referee חדש המקבל את דגל הסיום ואת scores.
3. כעט ניצור ונשמור בשדה המיוחד הבודד את startFlag, scores ואת מספר הקבוצות.

שימו לב שכשאתם יוצרים כל משתנה ממחלקה המממשת את Runnable, יש להציב ישר ב-Thread ולבצע start.
אם הגדרתם נכון, הממתנה תתבצע אוטומטית.

חלונות GUI והתממשקות מולם:

בחלק זה עליכם לתכנן והחליט על הצורות השונות אותם תקימו את הממשק הגרפי ואיך אתם מתחילים את התחרויות השונות.

בעיות עיקריות עליכם לפתור בחלק זה:

1. חלוקת החיות לקבוצות (לתחרות רגילה ולתחרות שליחים) - יש להוסיף מספר חלונות וכפתורים המתאימים ליציאה – מתואר בהמשך.
2. הוספת חלונות מידע (ולקשרם כך שיגיעו ל-TournamentThread כדי שהוא יוכל לעדכן את הנתונים) שיפתחו בלחיצה על כפתור מתאים.
3. בגלל כל המקביליות שתוארה בעמודים הקודמים, מאוד מסובך לממש כמו שצריך את ציור החיות על המסלולים. ולכן נבצע פתרון הרבה יותר פשוט, כך שעל מנת לבצע את הציור המתאים, כל tick (כדאי Thread נוסף) יש לעבור על כל חיה, בבלוק synchronized על החיה עצמה נקבל את מיקומה הנוכחי, ואז נצייר אותה בהתאם.

חלון ליצירת תחרות חדשה:

בלחיצה על הכפתור Add Competition, אמור להיפתח חלון חדש. מתואר כאן רעיון כללי מה אמור להיות בחלון, אבל כל עוד אתם ממשים משהו המאפשר את אותה פונקציונליות זה יתקבל.

בעזרת Radio Buttons הוא מאפשר למשתמש לבחור האם הוא מעוניין בתחרות שליחים או בתחרות רגילה.

מתחת לבחירה מופיע מעין טבלה ובה כפתורי הוספה או פרטי חיות משתתפות. על עמודה מייצגת קבוצת חיות וכל שורה מייצגת את החיות השונות בקבוצות. כלומר יש לנו לכל עמודה כפתור להוספת חיות (אפשר להגביל את מספר החיות בקבוצה, לבחירתכם) לקבוצה, וקיימת עמודה שבה בראשה יש כפתור הוספה היוצר קבוצה חדשה עם חיה בודדת.
מומלץ מאוד להגדיר setEnabled(false) כאשר אתם לא מאפשרים שינוי מסוים (למשל אם קיימת קבוצה עם יותר מחיה אחת – זה כבר לא יכול להיות תחרות רגילה) – זה גורר לממשק להיראות יותר מקצועי.

בלחיצה על כפתור הוספה נפתח חלון מתאים המאפשר לבחור איזו חיה לבחור, להגדיר ערכים מתאימים, להגדיר שם וכו'.