גיליון תשובות

מספר נבחן:_____

שאלה 1 (25 נקודות)

א.

```
int *refc_;

SmartPointer(T *ptr) {
            ptr_ = ptr;
            refc_ = new int(1);
      }

      SmartPointer(const SmartPointer<T>& rhs) {
            link(rhs);
      }

      SmartPointer<T>& operator=(const SmartPointer<T>& rhs) {
            if (&rhs != this) {
                  clean();
                  link(rhs);
            }
            return *this;
      }
void link(const SmartPointer<T>& rhs) {
            ptr_ = rhs.ptr_;
            refc_ = rhs.refc_;
            (*refc_)++;
      }

void clean() {
            if (refc_ == nullptr) return;

            if (*refc_ == 1) {
                  delete ptr_;
                  delete refc_;
                  ptr_ = nullptr;
                  refc_ = nullptr;
            }
            else {
                  (*refc_)--;
            }
      }
```

מי שלא זכר או מימש סמארט פוינטר אלא מבנה אחר הורדו -10

מי שלא השתמש ב int* refc הורדו -3
מי שלא מימש נכון בנאים אופרטור ובנאי הורס הורדו -3 -4 נק לכל אחד

ב.

```
SmartPointer(SmartPointer<T>&& rhs) {
            steal(rhs);
      }
      void steal(SmartPointer<T>& rhs) {
            ptr_ = rhs.ptr_;
            refc_ = rhs.refc_;
            rhs.ptr_ = nullptr;
            rhs.refc_ = nullptr;
      }
```

מי שלא מימש move ctor הורדו בין -3 ל-5
מי ששכח לשים NULLPTR בSTEAL או שינה את הrefc הורדו -3

שאלה 2 (30 נקודות)

1.א

```
AtomicBoolean s = new AtomicBoolean(false);

t1:
void run() {
    …
    f();
    s.set(true);
    …
}

t2:
void run() {
    …
    while(!s.cas(true, false);
    f();
    …
}
```

2.א

```
AtomicBoolean s1 = new AtomicBoolean(false);
```

```
AtomicBoolean s2 = new AtomicBoolean(true);

t1:
void run() {
   while (true) {
         while(!s1.cas(false, true));
          A
          s2.set(false);
   }
}

t2:
void run() {
   while (true) {
         while(!s2.cas(false, true));
          B
          s2.set(false);
   }
}

t3:
void run() {
   while (true) {
         while(!s2.cas(false, true));
          C
          s1.set(false);
   }
}
```

ב.1

כן / **לא** (הקיפו את התשובה הנכונה)
הסבר: התשובה כן.

תרחיש: נניח שני תרדים T1 ו T2
T1 מגיע עד לשורה id=Thread.getCurrentThreadId(), מבלי לבצע אותה, ונעצר ע"י המתזמן. T2 מגיע עד לשורה lst.add(1), מבלי לבצע אותה, ונעצר ע"י המתזמן. T1 ממשיך וגם כן מגיע עד לשורה lst.add(1) ונעצר ע"י המתזמן. כעט, יש לנו 2 תרדים שעלולים לנסות להכניס חוליות חדשות לרשימה מקושרת שלנו יחד. מכיוון שהרשימה אינה מוגנת, אחת החוליות עלולה ללכת לאיבוד.

ב.2

האם במימוש של בן יתכן מצב של **deadlock** ? **כן** / **לא** (הקיפו את התשובה הנכונה)

האם במימוש של בן יתכן מצב של **livelock** ? **כן** / **לא** (הקיפו את התשובה הנכונה)

הסבר: לא יכול להיווצר מצב של חבק (deadlock) לפי ההגדרה, כי אין תרדים שנכנסים למצב נעילה Blocked.
כן עלול להיווצר מצב של livelock.
תרחיש: נניח שני תרדים T1 ו T2. נניח כי T1 רץ עד ...=id כולל השורה הזו, ונעצר ע"י המתזמן. T2 רץ עד break , כולל. כלומר , T2 לא ירוץ יותר לעולם. T1 חוזר לרוץ ומבצע T2 . isBusy=true ממשיך לרוץ ונכשל בבדיקה (...==if(id , כי id הינו של T2.
כעת, אף תרד לא יכול להכנס יותר , ו T1 יבצע את (while(true הפנימי לנצח.

זה livelock כי T1 כן מקבל זמן CPU ומריץ שורות קוד, אך הוא תקוע ולא מתקדם משם.

מי שכתב שהתשובה היא deadlock אבל הביא תרחיש נכון, קיבל את כל הנקודות.

שאלה 3 (30 נקודות)

א.

```
public class GetProductPrices implements Command<Stores> {

    String product;

    public GetProductPrices (string product) {
        this.product = product;
    }

    @Override
    public Serializable execute(Stores stores) {
         Map<String,Integer> store2price = new HashMap<String,Integer>();
         for (String stroe : stores.getStores()) {
            Integer price = getProductsOfStore(store).get(product);
            if (price != null)
               store2price.put(store,price);
         }
         return store2price;
    }

}
```

```
public class AddStoreProducts implements Command<Stores> {

    String store;
    Map<String,Integer> product2price;

    public addStoreProducts(String store, ,Map<String,Integer> product2price) {
       this.store = store;
       this.product2price = product2price;
    }

    @Override
    public Serializable execute(Stores stores) {
         for (Entry<String,Integer> productPrice : product2Price)
            stores.addProductToStore(store, entry.getKey(), entry.getValue());
         return null;
    }
}
```

ב.

```
public class PriceServer {

    public static void main(String[] args) {

        Stores stores = new StoresImpl(); //one shared object

        new Reactor(
          7777, //port
          10,
          () => new RemoteCommandInvocationProtocol<Stores>(stores),
          () => new ObjectEncoderDecoder()
        ).serve();
    }
}
```

ג. 11 [ת'רד התקשורת (הת'רד הראשי), עשרת הת'רדים שנותנים עבודה ב executor]

ד. במתודת ה **main** במחלקה **PriceServer** יש להחליף את **new Reactor** ב **new ThreadPerClient** (ולהוריד את הפרמטר 10)

ה. לשרת התחברו 1000 לקוחות. מודל הריאקטור יצליח לשרת את כולם, כי מספר הת'רדים בו קבוע. במודל ה ThreadPerClient השרת יקרוס מן הסתם מחוסר יכולת להריץ 1000 ת'רדים.

שאלה 4 (15 נקודות)

א. [4 נק']

```
def UpdateScoreForMovie(self, movie_id, additional_score):
      C = self._conn.cursor()
      C.execute("""UPDATE Movies SET score =  (score*num_scoreres+(?)) /
      (num_scorers+1), num_scorers = num_scorers+1 WHERE id = (?)""",
      [additional_score, movie_id])
```

הערות: היו כאלו ששמו את נוסחת העדכון כחלק מהמשתנים של פייתון. חשוב להבין שבפייתון הוא לא מכיר את ערך השדות score או num_scorers שנמצאים בטבלה. זהו DAO, ולכן שדות כמו self.score גם אינו מוגדר פה.

ב. [4 נק']

```
def GetMostPopularActionMovies(self, number_of_movies, minimum_selections):
       self._conn.cursor().execute("""SELECT (*) FROM Movies WHERE isAction=True
       AND num_scorers >= (?) ORDER BY score DESC LIMIT (?)  """,
     [minimum_selections, number_of_movies])
```

הערות: שימו לב - בהנחייה הופיע הסינטקס

SELECT column-list FROM table_name [WHERE condition]
[ORDER BY column1, column2, .. columnN] [ASC | DESC] [LIMIT number_records];

הסימון של הסוגריים המרובעות [] הוא הסימון המקובל בתיעוד תוכנה לערכים/מילות סינטקס אופציונליים לשאילתא. זהו אינו חלק מהסינטקס עצמו.

ג. [5 נק']

```
SELECT COUNT(*), AVG(Movie_watches.score) FROM Movies JOIN Movie_watches ON Movies.id
= Movie_watches.movie_id WHERE user_id = 1 AND isAction = True AND
Movie_watches.score IS NOT NULL
```

הערה: ניתן גם לשים את score בתוך COUNT ולהוריד את תנאי הNOT NULL, כיוון שאז רשומות עם ערך NULL אינן נספרות ב-COUNT.

ד. [2 נק']

לפי הנלמד בקורס השאילתא בסעיף הקודם תמומש כחלק מה-repository, כיוון שהיא כוללת שליפת מידע מיותר מטבלה אחת. ה-DAO שהוגדר בקורס כלל גישות לטבלה בודדת.