אוניברסיטת בן-גוריון

מדור בחינות

מספר נבחן:_____

רשום/רשמי תשובותיך בגיליון התשובות בלבד
תשובות מחוץ לגיליון לא יבדקו.

בהצלחה!

תאריך הבחינה: 4.2.2005
שם המורה: ד"ר מיכאל אלחדד
מני אדלר
שם הקורס: תכנות מערכות
מספר הקורס: 202-1-2031
מיועד לתלמידי: מדעי המחשב, הנדסת תוכנה
שנה: תשס"ה
סמסטר: א'
מועד: א'
משך הבחינה: שלש שעות
חומר עזר: אסור

שאלה 1 (30 נקודות)

במשחק המיתולוגי Fire (Game & Watch 1980) נדרשים צמד כבאים אמיצים, בעזרת אלונקה, לקלוט בשלום ניצולים מבית בוער. מיקום הכבאים ואלונקתם הנאמנה ניתן לשינוי בעזרת הכפתורים left ו right.

במימוש המשחק לתוכנית מחשב ב Java, הוחלט להגדיר ארבעה ת'רדים:

- Thrower מטיל אנשים מהבנין הבוער.
- UserInterface מזהה לחיצות על המקשים left ו right.
- Left עבור כל הקשה על הכפתור left (המזוהה ע"י הת'רד UserInterface), מזיז את האלונקה והכבאים מקום אחד שמאלה.
- Right עבור כל הקשה על הכפתור right (המזוהה ע"י הת'רד UserInterface), מזיז את האלונקה והכבאים מקום אחד ימינה.

בשאלה זו נתמקד במחלקה Firemen הממשת את זוג הכבאים והאלונקה, ובשני הת'רדים ה"מכירים" אותה – Left ו Right.

המצב הפנימי של המחלקה Firemen מורכב מהשדה _x מטיפוס int המציין את מיקום הכבאים והאלונקה בציר מספרים מ 1 עד 10, כאשר 1 מציין את הקצה השמאלי של המסך ו 10 את הקצה הימני. בנוסף מוגדרות המתודות moveLeft() ו moveRight() המשנות את ערכו של _x מקום אחד שמאלה או ימינה, בהתאם.

```
class Firemen
{
   public static final int MIN_X = 1;
   public static final int MAX_X = 10;
   private int x_;

   // @INV:  MIN_X < =  x_  < = MAX_X
   Firemen() { x_ = 5; }

   public void repaint()
   {
      // code that paints the firemen and the stretcher at position x_
   }

   public void moveLeft()
   {
      if (x_ > MIN_X)
         x_--;
      repaint();
   }

   public void moveRight()
   {
      if (x_ < MAX_X)
         x_++;
      repaint();
   }
}
```

ה'רד Left מפעיל את המתודה moveLeft() בלבד, וה'רד Right מפעיל את המתודה moveRight() בלבד. במידה ובוצעו כמה הקשות על הכפתורים left/right הן יזוהו על ידי ה'רד UserInterface, וה'רדים Left/Right יבצעו בהתאם מספר פעמים moveLeft()/moveRight() על Firemen.

א. האם המחלקה Firemen בטוחה תחת החישוב המקבילי של ה'רדים Left ו Right מבחינת קיום התכונה הנשמרת של ערך השדה _x בתחום [MIN_X,MAX_X]? הסבר/י. (6 נקודות)

ב. נגדיר את קלט התוכנית כסדרה של לחיצות על המקשים left ו right, לדוגמא: (left,left,right), ואת פלט התוכנית, כסדרה המתארת את מיקום האלונקה, לדוגמא: (5,4,3,4).
האם המחלקה Firemen בטוחה מבחינת התאמת כל פלט אפשרי של הרצה במודל חישוב מקבילי לפלט הרצה במודל חישוב סידרתי בסדר כל שהוא של הקלט? (6 נקודות)

ג. נסח/י תנאי התחלה וסיום (@precond, @postcond) למתודות moveLeft() ו moveRight(), וציין/ני האם תנאים אלו מתקיימים בהכרח. (6 נקודות)

ד. בגרסה המתקדמת של המשחק, הוחלט להוסיף שני כפתורים up ו down, וכן את ה'רדים Up ו Down המגביהים ומנמיכים את האלונקה על ידי הפעלת המתודות moveUp() ו moveDown(). התכונה הנשמרת הורחבה, והיא כוללת את האילוצים:
   - גובה הקרקע מוגדר להיות 1, ניתן להגביהה את האלונקה עד לגובה 3 =_y.
   - בתחום [4,6] של _x האלונקה חייבת להיות על הקרקע.

בנוסף לכך, נקבע כי בדומה להתעלמות מהקשות על left כאשר 1 = _x ומהקשות right כאשר 10=_x, יש להתעלם מהקשות על up כאשר 3 = _y ומהקשות על down כאשר 1 = _y. אולם במקרה של הקשה על left/right/up/down הסותרת את התנאי של המצאות על הקרקע בתחום [4,6] של _x, יש להמתין ולבצע את הפעולה כאשר היא תתאפשר. לדוגמא, אם מיקום האלונקה הוא (3,2) ובוצע right ואח"כ down, על האלונקה לעבור בסופו של דבר למקום (4,1) מבלי לעבור במקום הלא חוקי (4,2). וכן אם מיקום האלונקה

הוא (4,1) ובוצע up ואח"כ left, על האלונקה לעבור בסופו של דבר למקום (3,2) מבלי לעבור במקום הלא חוקי (4,2).

עדכן/י את המחלקה Firemen כך שתתמוך בדרישות החדשות, תוך שמירה על הבטיחות הן מבחינת התכונה הנשמרת, הן מבחינת התאמת פלט ההרצה המקבילית לפלט כל שהוא של הרצה סדרתית , והן מבחינת קיום תנאי ההתחלה והסיום של השיטות השונות. במידה ותנאי ההתחלה לא מתקיימים עבור פעולה מסוימת, יש לדאוג שהת'רד המבצע לא יתפוס זמן CPU (12 נקודות)

שאלה 2 (30 נקודות)

התבוננו במחלקה הבאה, המממשת מבנה נתונים של קבוצת מספרים שלמים. בקבוצה כידוע, אין שני אברים בעלי אותו ערך. גודל הקבוצה חסום, כך שהיא יכולה להכיל עד m אברים. ערך כל אבר בקבוצה מוגבל לתחום המספרים [1,n]

```
class intset {

private:
   int maxsize_;      // How many elements the set can store
   int maxval_;       // All elements values must be in [1,maxval_]
   vector<int> elements_;
public:
   intset(int m, int n); //at most m elements with values in range [1,n]
   ~intset();
   bool member(int t);            // is t a member?
   void insert(int t);            // add t to set
};
```

א. הגדירו את האינווריאנטה inv@ של המחלקה, וממשו מתודה bool inv() הבודקת את קיום האינווריאנטה. (5 נקודות)
הניחו כי קיימת פונקציה bool hasRepetitions(const std::vector<int>& v) המקבלת ווקטור של מספרים שלמים ומחזירה ערך true אם מספר כל שהוא מופיע יותר מפעם אחת.

ב. הגדירו את תנאי ההתחלה precond@ עבור ה constructor, וממשו אותו. (5 נקודות)
במידה ותנאי ההתחלה לא מתקיימים יש לזרוק exception, השתמשו בתחביר הבא:

```
if (<condition>)
     throw exception("explain why the exception is thrown");
```

<u>הערה</u>: הדרישה עבור קבוצה עם גודל מקסימלי maxsize כי תיתכן אפשרות שהקבוצה תכיל maxsize אברים.

ג. ממשו את ה destructor. (2 נקודות)

ד. הגדירו תנאי התחלה וסיום (@precond, @postcond) למתודות insert ו member, וממשו את המתודות. (6 נקודות)
במידה ותנאי ההתחלה או הסיום לא מתקיימים יש לזרוק exception, השתמשו בתחביר הבא:

```
if (<condition>)
     throw exception("explain why the exception is thrown");
```

ה. הוסיפו const בכל מקום שמתבקש בהגדרת המחלקה. (3 נקודות)

ו. ציינו את הבעיות בקוד להלן (9 נקודות)

התייחסו לבעיות מסוג:

- בעיות קומפילציה
- זריקת exceptions
- "נזילות" זיכרון (memory leaks)
- גישה לזיכרון משוחרר ו/או שאינו מוקצה (null pointers)

במידה ושורת קוד גורמת לזריקת exception הניחו כי תהליך ההרצה ממשיך ועובר לבצע את שורת הקוד הבאה.

```
{
   intset s0(3, 5);
   intset* s1 = new intset(2, 5);
   intset s2(10, 5);
   intset s3;
   s0.insert(1);
   s0.insert(1);
   s0.insert(2);
   s0.insert(3);
   s0.insert(5);
   s0.insert(6);
   *s1 = s0;
   {
      intset s4(1, 5);
      s4.insert(3);
      s0 = s4;
   }
   s0.member(1);
}
```

שאלה 3 (12 נקודות)

נתונה מחלקה `IncNumber` המממשת את הממשק `Increasable` באופן הבא:

```
public interface Increasable
  extends java.io.Serializable
{
    public void increase();
}

class IncNumber
  implements Increasable
{
  protected long num_;
  protected long offset_;

  public IncNumber(long n, long o) {
      num_ = n;
      offset_ = o;
  }

  public synchronized void increase() {
      num_ += offset_;
  }

  public long getNum()  { return num_; }
}
```

MarshalOutputStream היא מחלקה ב Java היורשת את המחלקה OutputStream. בשאלה זו נשתמש בשיטה אחת שלה:

```
void writeObject(Object obj)
```
השיטה כותבת ל OutputStream את ה serialization של האובייקט obj, בהתאם לסוגו, על פי חוקי RMI שנלמדו בכיתה.

באותו אופן, MarshalInputStream היא מחלקה ב Java היורשת את המחלקה InputStream. בשאלה זו נשתמש בשיטה אחת שלה:

```
Object readObject()
```
השיטה מבצעת deserialization לאוביקט המגיע ב InputStream, בהתאם לסוגו, על פי חוקי RMI שנלמדו בכיתה, ומחזירה אותו.

להלן תיאורם של client ושל server:

```
import sun.rmi.server.MarshalOutputStream;
import sun.rmi.server.MarshalInputStream;
import java.net.Socket;

public class client {
    public static void main(String[] args) throws Exception {
        Socket s = null;
        IncNumber element = new IncNumber(2,7);
        Socket server = new Socket(args[0], 2000);
        MarshalOutputStream out =
           new MarshalOutputStream(server.getOutputStream());
        out.writeObject(element);

        MarshalInputStream in =
            new MarshalInputStream(server.getInputStream());
        Boolean reply = (Boolean) in.readObject();

        System.out.println(element.getNum());

        in.close();
        out.close();
        server.close();
    }
}
```

```
import sun.rmi.server.MarshalOutputStream;
import sun.rmi.server.MarshalInputStream;
import java.net.Socket;
import java.net.ServerSocket;

public class server {
    public static void main(String[] args) throws Exception {
        ServerSocket serverSocket = new ServerSocket(2000);
        while (true) {
            Socket client = serverSocket.accept();
            MarshalInputStream in =
                new MarshalInputStream(client.getInputStream());
            Increasable element = (Increasable) in.readObject();

            for (int i=0;  i<3;  i++)
```

```
                element.increase();

            MarshalOutputStream out =
                new MarshalOutputStream(client.getOutputStream());
            out.writeObject(new Boolean(true));

            out.close();
            in.close();
            client.close();
        }
    }
}
```

א. מה מדפיסה התוכנית client? נמק/י (4 נקודות)

ב. שנה/י את הגדרת הממשק `Increasable` והמחלקה `IncNumber` המממשת אותו, מבלי לשנות את הקוד של client ו server כך שהערך המודפס ב client יהיה מעודכן לשינויים ב server, והסבר/י את הסיבה לכך (8 נקודות)

שאלה 4 (18 נקודות)

להלן המחלקה MessageProcessorTask של תבנית ה Reactor שנלמדה בכיתה:

```
class MessageProcessorTask implements Task {
    protected String _message;
    protected SocketChannel _channel;

    public MessageProcessorTask(String message, SocketChannel channel) {
        _message = message;
        _channel = channel;
    }

    public void executeTask() throws TaskFailedException {
        String response = "Got your message!;";
        try {
          _channel.write(ByteBuffer.wrap(response.getBytes()));
        } catch (IOException io) {
          throw new TaskFailedException(
            "I/O exception while processing the message: " + _message, io);
        }
    }
}
```

השיטה write(ByteBuffer bytes) במחלקה SocketChannel מחזירה מספר שלם המציין כמה bytes נכתבו בפועל ל OutputStream (להזכירכם, מדובר ב non blocking I/O).

א. מה הבעיה בקוד של executeTask() המוגדר למעלה? (3 נקודות)
ב. תקנו את הפרוצדורה executeTask() כך שתימנע הבעיה בסעיף א (5 נקודות).
ג. עדכנו את הקוד של תבנית ה Reactor כך שתימנע הבעיה של סעיף א, מבלי לעכב את הת'רד בניסיונות חוזרים ונשנים לכתוב ל SocketChannel (10 נקודות)

חומר עזר:

1. מתודות ושדות נבחרות/ים של המחלקה SelectionKey

static int	**OP_ACCEPT** Operation-set bit for socket-accept operations.
static int	**OP_CONNECT** Operation-set bit for socket-connect operations.
static int	**OP_READ** Operation-set bit for read operations.
static int	**OP_WRITE** Operation-set bit for write operations

שילוב של מספר ביטים ניתן ע"י פעולת or לוגית, לדוגמא: OP_CONNECT | OP_ACCEPT

boolean	**isAcceptable**() Tests whether this key's channel is ready to accept a new socket connection.
boolean	**isConnectable**() Tests whether this key's channel has either finished, or failed to finish, its socket-connection operation.
boolean	**isReadable**() Tests whether this key's channel is ready for reading.
boolean	**isWritable**() Tests whether this key's channel is ready for writing.

2. קטעי קוד נבחרים של תבנית ה Reactor

```
public class Reactor extends Thread {

...

 public void run() {
   try {
     _pool = new ThreadPool(_poolSize);
     _pool.startPool();

     ServerSocketChannel ssChannel = ServerSocketChannel.open();
     ssChannel.configureBlocking(false);
     ssChannel.socket().bind(new InetSocketAddress(_port));

     Selector selector = Selector.open();
     ssChannel.register(selector,SelectionKey.OP_ACCEPT,
                        new ConnectionAcceptor(selector,ssChannel,_pool));
     while (_shouldRun) {
       selector.select();
       Iterator it = selector.selectedKeys().iterator();
       while (it.hasNext()) {
         SelectionKey selKey = (SelectionKey)it.next();
         it.remove();
         if (selKey.isValid() && selKey.isAcceptable()) {
           ConnectionAcceptor connectionAcceptor =
                                (ConnectionAcceptor)selKey.attachment();
           connectionAcceptor.accept();
```

```
          }
          if (selKey.isValid() && selKey.isReadable()) {
            ConnectionHandler connectionHandler =
                                    (ConnectionHandler)selKey.attachment();
             connectionHandler.read();
          }
        }
      }
    } catch (IOException e) {
       e.printStackTrace(System.err);
       stopReactor();
    }
    stopReactor();
  }
...
```

```
public class ConnectionAcceptor {
  protected Selector _selector;
  protected ServerSocketChannel _ssChannel;
  protected ThreadPool _pool;

  public ConnectionAcceptor(Selector selector,
                            ServerSocketChannel ssChannel, ThreadPool pool)
  {
    _selector = selector;
    _ssChannel = ssChannel;
    _pool = pool;
  }

  public void accept() throws IOException {
    SocketChannel sChannel = _ssChannel.accept();
    if (sChannel != null) {
      sChannel.configureBlocking(false);
      sChannel.register(_selector, SelectionKey.OP_READ,
                        new ConnectionHandler(sChannel, _pool));
    }
  }
}
```

```
 public class ConnectionHandler {
    public static final int BUFFER_SIZE = 256;
    public static final char MESSAGE_END = ';';
    protected SocketChannel _sChannel;
    protected String _incomingData;
    protected ThreadPool _pool;

    public ConnectionHandler(SocketChannel sChannel, ThreadPool pool) {
        _sChannel = sChannel;
        _pool = pool;
        _incomingData = "";
    }

    public void read() throws IOException {
      ByteBuffer buf = ByteBuffer.allocate(BUFFER_SIZE);
      while (true) {
          buf.clear();
          int numBytesRead = _sChannel.read(buf);
          if (numBytesRead == -1) {
             _sChannel.close();
             break;
          }
```

```
            if (numBytesRead > 0) {
               buf.flip();
               String str = new String(buf.array(), 0, numBytesRead);
               _incomingData = _incomingData + str;
            }
            if (numBytesRead < BUFFER_SIZE)
                break;
         }
         while (true) {
             int pos = _incomingData.indexOf(MESSAGE_END);
             if (pos==-1)
                 break;
             String message = _incomingData.substring(0, pos);
             _incomingData = ((pos ==_incomingData.length()-1) ?
                              "" : _incomingData.substring(pos+1));
             _pool.addTask(new MessageProcessorTask(message, _sChannel));
         }
     }
}
```

שאלה 5 (10 נקודות)

בישיבה האחרונה של האקדמיה ללשון עברית הוחלט לאחסן את המילון העברי בבסיס נתונים.
המילון מורכב ממילים וניתוחים.

- מילה מוגדרת על ידי:
  - מחרוזת המילה בעברית
  - מספר ההופעות של המילה במקורות
- ניתוח מוגדר על ידי:
  - מספר סידורי
  - חֵלֶק דִיבֶּר (שם,פועל,תואר)
  - מין (זכר,נקבה)
  - מספר (יחיד,רבים)
- למילה עשויים להיות כמה ניתוחים אפשריים, וכן ניתוח אחד עשוי להתאים לכמה מילים.

לדוגמא, המילה 'מספרים':
מחרוזת המילה בעברית: 'מספרים'
שכיחות: 15370
ניתוחים: 3, שם, זכר, יחיד [עבור הקריאה 'מִסְפָּרַיִם']
4, שם, זכר, רבים [עבור הקריאה 'מִסְפָּרִים']
7, פועל, זכר, רבים [עבור הקריאה 'מְסַפְּרִים']

ניתוח מספר 3 מתאים גם למילה 'כיסא', בעוד שניתוח מספר 7 מתאים גם למילה 'ילמדו'.

א. הגדר/י מודל נתונים לאחסון המילים וניתוחיהם באופן שלם ומינימאלי. ההגדרה תתבסס על טבלאות, שדות, מפתחות ראשיים וזרים - כפי שנלמד בכיתה. (7 נקודות)

ב. כתוב/בי שאילתת SQL המציגה את רשימת המילים המופיעות יותר מ 10,000 פעמים במקורות, ממוינים בסדר יורד על פי מספר הופעותיהם. (3 נקודות)