אוניברסיטת בן-גוריון

מדור בחינות

מספר נבחן:_____

רשמו תשובותיכם בגיליון התשובות בלבד
תשובות מחוץ לגיליון לא יבדקו.

בהצלחה!

תאריך הבחינה: 4.3.2013
שם המורה: ד"ר אנדרי שרף
ד"ר מני אדלר
שם הקורס: תכנות מערכות
מספר הקורס: 202-1-2031
מיועד לתלמידי: מדעי המחשב, הנדסת תוכנה
שנה: תשע"ג
סמסטר: א'
מועד: ב'
משך הבחינה: שלש שעות
חומר עזר: אסור

שאלה 1 (30 נקודות)

במועד א' התוודענו לממשק המרתק BlockedBall המייצג כדור במרחב דו ממדי, הנתון בתוך מסגרת מלבנית, כך שהכדור אינו יכול לחרוג ממסגרת זו, ולסימולציה בה נע הכדור למעלה/למטה/ימינה/שמאלה ע"פ שלושה ת'רדים:

VerticalMovment: בכל צעד מזיז את הכדור למעלה עד שהוא מגיע למסגרת העליונה, אז הוא משנה את כיוונו ומתחיל לנוע למטה עד שהוא מגיע למסגרת התחתונה, וחוזר חלילה.
HorizontalMovment: בכל צעד מזיז את הכדור ימינה עד שהוא מגיע למסגרת הימנית, אז הוא משנה את כיוונו ומתחיל לנוע שמאלה עד שהוא מגיע למסגרת השמאלית, וחוזר חלילה.
Wind: בכל צעד מזיז את הכדור ימינה/שמאלה/למעלה/למטה על פי בחירה אקראית.

להלן הקוד, כפי שניתן במועד א', סעיף ג'.

```
interface BlockedBall {
        int getRectUpper(); //returns the upper line of the blocking rectangle
        int getRectBottom(); //returns the bottom do line of the blocking rectangle
        int getRectLeft(); //return the left line of the blocking rectangle
        int getRectRight(); //return the right line of the blocking rectangle

        int getBallX(); //returns the X position of the ball
        int getBallY(); //returns the Y position of the ball

        void up() throws Exception; // increases the Y position by 1
        void down() throws Exception; //decreases the Y position by 1
        void right() throws Exception; // increases the X position by 1
        void left() throws Exception; //decreases the X position by 1
}
```

```
abstract class CheckedBlockedBall implements BlockedBall {

	public boolean checkInv() {
		// returns true if the invariant holds
		...
	}

	public boolean checkPreCondUp() {
		// returns true if the pre-condition of up() method holds
		...
	}

	public boolean checkPreCondDown() {
		// returns true if the pre-condition of down() method holds
		...
	}

	public boolean checkPreCondRight() {
		// returns true if the pre-condition of right() method holds
		...
	}

	public boolean checkPreCondLeft() {
		// returns true if the pre-condition of left() method holds
		...
	}

}
```

```
class SimpleBlockedBall extends CheckedBlockedBall {

	SimpleBlockedBall(int rectUp, int rectBottom, int rectLeft, int rectRight, int ballX, int ballY) throws
Exception {
		_rectUp = rectUp; _rectBottom = rectBottom; _rectLeft = rectLeft; _rectRight = rectRight;
		_ballX = ballX; _ballY = ballY;
		if (!checkInv())
			throw new Exception("Invariant does not hold!");

	}

	public int getRectUpper() { return _rectUp; }
```

```
	public int getRectBottom() { return _rectBottom; }
	public int getRectLeft() { return _rectLeft; }
	public int getRectRight() { return _rectRight; }
	public int getBallX() { return _ballX; }
	public int getBallY() { return _ballY;  }

	public void up() throws Exception {
		if (!checkPreCondUp())
			throw new Exception("The pre-condition for the up() method does not hold!");
		_ballY++;
	}
	public void down() throws Exception {
		if (!checkPreCondDown())
			throw new Exception("The pre-condition for the down() method does not hold!");
		_ballY--;
	}

	public void right() throws Exception {
		if (!checkPreCondRight())
			throw new Exception("The pre-condition for the right() method does not hold!");
		_ballX++;
	}
	public void left() throws Exception {
		if (!checkPreCondLeft())
			throw new Exception("The pre-condition for the left() method does not hold!");
		_ballX--;
	}

	protected final int _rectUp, _rectBottom, _rectLeft, _rectRight;
	protected int _ballX, _ballY;
}
```

```
enum Orientation  {
	LEFT, RIGHT, UP, DOWN ;
}
```

```
class VerticalMovment implements Runnable {

	BlockedBall _blockedBall;
	Orientation _orientation;
```

```
	VerticalMovment(BlockedBall blockedBall, Orientation orientation) {
		_blockedBall = blockedBall;
		_orientation = orientation;
	}

	public void run() {
		while (!Thread.interrupted()) {
			int y = _blockedBall.getBallY();
			if (y == _blockedBall.getRectUpper())
				_orientation = Orientation.DOWN;
			if (y == _blockedBall.getRectBottom())
				_orientation = Orientation.UP;

			try {
				if (_orientation == Orientation.UP)
					_blockedBall.up();
				else
					_blockedBall.down();
			} catch (Exception e) {
				System.out.println(e);
			}
		}
	}
}
```

```
class HorizontalMovment implements Runnable {

	BlockedBall _blockedBall;
	Orientation _orientation;

	HorizontalMovment(BlockedBall blockedBall, Orientation orientation) {
		_blockedBall = blockedBall;
		_orientation = orientation;
	}

	public void run() {
		while (!Thread.interrupted()) {
			int x = _blockedBall.getBallX();
			if (x == _blockedBall.getRectLeft())
				_orientation = Orientation.RIGHT;
			if (x == _blockedBall.getRectRight())
				_orientation = Orientation.LEFT;
```

```
                        try {
                                if (_orientation == Orientation.LEFT)
                                        _blockedBall.left();
                                else
                                        _blockedBall.right();
                        } catch (Exception e) {
                                System.out.println(e);
                        }
                }
        }
}
```

```
public class Wind implements Runnable {

        BlockedBall _blockedBall;
        List<Orientation> _orientations;
        Random _rand;

        Wind(BlockedBall blockedBall) {
                _blockedBall = blockedBall;
                _orientations = new ArrayList<Orientation>();
                _orientations.add(Orientation.LEFT);
                _orientations.add(Orientation.RIGHT);
                _orientations.add(Orientation.UP);
                _orientations.add(Orientation.DOWN);
                _rand = new Random();
        }

        public void run() {
                while (!Thread.interrupted()) {
                        Orientation orientation = _orientations.get(_rand.nextInt(4));
                        try {
                                if (orientation == Orientation.LEFT)
                                        _blockedBall.left();
                                if (orientation == Orientation.RIGHT)
                                        _blockedBall.right();
                                if (orientation == Orientation.UP)
                                        _blockedBall.up();
                                if (orientation == Orientation.DOWN)
                                        _blockedBall.down();
                        } catch (Exception e) {
                        }
                }
```

```
        }
}
```

```
public class Simulation {
        public static void main(String[] args) throws Exception {
                BlockedBall blockedBall = new SimpleBlockedBall(10,1,1,10,5,5);
                new Thread(new HorizontalMovment(blockedBall, Orientation.LEFT)).start();
                new Thread(new VerticalMovment(blockedBall, Orientation.UP)).start();
                new Thread(new Wind(blockedBall)).start();
        }
}
```

א. במועד א' ראינו כי המערכת אינה בטוחה, כי הכדור עשוי לחרוג מהמסגרת ובכך להפר את האינווריאנטה.
נתון כי מסגרת הכדור הינה אינסופית, כך שהכדור לא חורג ממנה אף פעם, והרצת התוכנית בטוחה.
הראו כי בכל זאת הרצת התוכנית אינה נכונה. ותקנו את הקוד כך שישמור על הנכונות.
בשאלה זו נתייחס לביצוע up(), down(), left(), right() כפעולות עליהן מתבססת הגדרת הנכונות.
[10 נקודות]

ב. עדכנו את הקוד, כך שכאשר לא מתקיימים תנאי ההתחלה עבור המתודות up(), down(), left(), right(), ת'רד המבצע ממתין עד אשר תנאים אלו יתקיימו.
יש למקד את ההמתנה, כך שהת'רד לא יתעורר עקב שינויים שאינם רלבנטיים למצב עליו הוא מחכה (כמו התעוררות עקב הגבהה/הנמכה של הכדור, כאשר הת'רד מחכה לתזוזה ימינה/שמאלה).
[10 נקודות]

ג. הראו כיצד המערכת עשויה להיקלע ל deadlock עקב יישום המדיניות של סעיף ב', ופתרו בעיה זו על ידי הפסקת ההמתנה לאחר 10 שניות.
ניתן לענות על סעיף זה גם אם לא מימשתם את סעיף ב.
[10 נקודות]

חומר עזר לסעיף ג: מתודות המחלקה Object

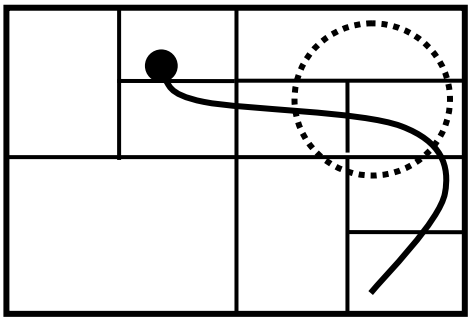
Method Summary	
protected Object	**clone**() Creates and returns a copy of this object.
boolean	**equals**(Object obj) Indicates whether some other object is "equal to" this one.
protected void	**finalize**() Called by the garbage collector on an object when garbage collection determines that there are no more references to the object.
Class<? extendsObject>	**getClass**() Returns the runtime class of an object.
int	**hashCode**() Returns a hash code value for the object.
void	**notify**()

	Wakes up a single thread that is waiting on this object's monitor.
void	**notifyAll**() Wakes up all threads that are waiting on this object's monitor.
String	**toString**() Returns a string representation of the object.
void	**wait**() Causes current thread to wait until another thread invokes the notify() method or the notifyAll() method for this object.
void	**wait**(long timeout) Causes current thread to wait until either another thread invokes the notify() method or the notifyAll() method for this object, or a specified amount of time has elapsed.
void	**wait**(long timeout, int nanos) Causes current thread to wait until another thread invokes the notify() method or the notifyAll() method for this object, or some other thread interrupts the current thread, or a certain amount of real time has elapsed.

שאלה 2 (30 נקודות)

לאור העניין שיצר מבנה הנתונים ממועד א', הוחלט על פיתוח המשך. גם כאן נעסוק כמו במועד א' במבנה חיפוש ב C++ עבור הBlockedBall משאלה 1 כאשר המבנה הינו חלוקה הירארכית של ה RectangleCell לתת-מלבנים באופן רקורסיבי.

אנו נעסוק במסלול תנועת הכדור בתוך הRectangleCell. מסלול תנועת הכדור נתון כדגימה בדידיה - אוסף של נקודות דו ממדיות (ולא תנועה רציפה). הניחו כי הכדור בעל רדיוס 0 (כלומר נקודה).



```
#include <vector>
using namespace std;

class Point2D{
public:
        void set(float x, float y){_x=x; _y=y;}
private:
        float _x;
        float _y;
};

class SceneCell{
public:
        virtual  void retrieveCells(Point2D center, float radius, vector<SceneCell*> &retrievedCells)=0;
        virtual  void buildStructure(int maxLevel, vector<Point2D> ballPath) = 0;
};

class RectangleCell : public SceneCell{
public:
        RectangleCell(float left, float right, float top, float bottom);

        virtual  void retrieveCells(Point2D center, float radius, vector<SceneCell*> &retrievedCells);
        virtual  void buildStructure(int maxLevel, std::vector<Point2D> ballPath) ;

private:
        float _left, _right, _top, _bottom; //rectangle limits
        RectangleCell *_topLeftSon, *_topRightSon, *_bottomLeftSon, *_bottomRightSon; //rectangle cell sons
};
```

א. ממשו את שתי הפונקציות העיקריות הבאות:

buildStructure המבנה החדש יאחסן בצורה יעילה את מסלול תנועת הכדור. הפונקציה מקבלת כפרמטר את מספר רמות הפיצול המקסימלי ומסלול הכדור. הפונקציה מפצלת מלבן על ידי חצית האורך והרוחב לשניים. שימו לב – אנחנו לא ניצור את כל 4 בנים אלא אך ורק את אותם הבנים הנחצים על ידי המסלול (הסתכלו בציור). יש להשתמש אך ורק ברקורסיה– אין להשתמש בלולאה!

retrieveCells שאילתא יעילה אשר מחזירה את כל ה **RectangleCell** הכי קטנים במבנה אשר המסלול עובר דרכם ונחתכים על ידי עיגול השאילתא. בציור – העיגול המקווקו הינו שאילתה אשר תחזיר את שלושת המלבנים הנחתכים על ידי העיגול.

ניתן להניח כי הבדיקה האם עיגול חוצה מלבן הינה קופסא שחורה נתונה עלידי פונקצית שרות isIntersect(**Point2D center, float radius**, RectangleCell &cell)

[14 נקודות]

ב. במבנה הנתון יש בעית תיכון (design) מהותית של המחלקות הגורמת לכך שאנחנו מבזבזים זכרון. היכן אנו מבזבזים זכרון? הצע דרך לפתור את הבזבוז (בונוס).

[4 + 2 נקודות בונוס]

```
void foo(RectangleCell r1){
   //----here-------
}

void main(){

      int maxLevel = 2;

      vector<Point2D> path;

      path.resize(3);

      path[0].set(1.7,0);

      path[1].set(1.8,1.2);

      path[2].set(0.6,1.6);

      RectangleCell rcell(0.0,0.0,2.0,2.0);

      rcell.buildStructure(maxLevel, path);

      foo(rcell);

}
```

ג. כתבו את תמונת הזיכרון המתקבלת בסימון here. אין להניח שום הנחות על המחלקות, להוסיף או לשנות אותם (מלבד הנדרש בסעיף א'). הניחו כי מסלול הכדור הנתון הוא בדיוק כמו בציור.

[12 נקודות]

שאלה 3	(10 נקודות)

על מחשב עם 24 מעבדים בוצעה שורת הפקודה הבאה:

```
>java Reactor 7895 12
```

נתון כי שני לקוחות מחוברים לשרת, וכי לקוחות אלו שלחו את ההודעות הבאות:

לקוח א: That was can-you-dig-it by Georgie Wood \n

לקוח ב: And now we'd like to do ark-the-angels-come \n

נתון כי הודעות אלו נקראו במלואן מה socket של כל לקוח ע"י ה ConnectionHandler שלו, והועברו ל Executor

במועד א' ראינו כי קיים תרחיש אפשרי בו כל הת'רדים ב Executor תקועים כי הודעת הלקוח הראשון נקראה בחלקים קטנים, כך שתור המשימות ב Executor מכיל 12 ProtocolTask זהים של אותו לקוח, דבר הגורם לתפיסת המנעול של ה ProtocolTask בשל סנכרון מתודת ה ()run. במקרה זה רק הת'רד התופש את המנעול עובד וכל שאר הת'רדים במצב blocked על אף שיש משימות בתור לביצוע עבור הלקוח השני.

```
public synchronized void run() {
    while (_tokenizer.hasMessage()) {
        T msg = _tokenizer.nextMessage();
        T response = this._protocol.processMessage(msg);
        if (response != null) {
          try {
            ByteBuffer bytes = _tokenizer.getBytesForMessage(response);
            this._handler.addOutData(bytes);
          } catch (CharacterCodingException e) { e.printStackTrace(); }
        }
    }
}
```

כדי לפתור בעיה זו, דאגנו לכך שרק מופע אחד של ProtocolTask לכל לקוח יהיה בתור המשימות של ה Executor, על ידי בדיקה האם הוא נמצא בתור המשימות לפני שמוסיפים אותו ל Executor במתודת ה ()read של ה ConnectionHandler.

```
public class ConnectionHandler<T> {
   …

   public void read() {
      if (_protocol.shouldClose())
         return;

      SocketAddress address = _sChannel.socket().getRemoteSocketAddress();
```

```
    logger.info("Reading from " + address);

    ByteBuffer buf = ByteBuffer.allocate(BUFFER_SIZE);
    int numBytesRead = 0;
    try {
        numBytesRead = _sChannel.read(buf);
    } catch (IOException e) {
        numBytesRead = -1;
    }
   if (numBytesRead == -1) {
        logger.info("client on " + address + " has disconnected");
        closeConnection();
        _protocol.connectionTerminated();
        return;
    }
     buf.flip();
    _task.addBytes(buf);
    if (!execute.getQueue().contains(_task))
       _data.getExecutor().execute(_task);
 }
```

תארו תרחיש בו עדיין יתכן שרק ת'רד אחד מתוך 12 הת'רדים ב Executor נמצא במצב ready, ואילו כל השאר נמצאים במצב blocked, למרות שתור המשימות ב Executor אינו ריק, ואינו מכיל גם מופע כפול של ProtocolTask של אותו לקוח. [10 נקודות]

שאלה 4 (20 נקודות)

בתרגיל התכנות הרביעי, מימשתם שרת עבור פרוטוקול IRC, המאפשר שליחה וקבלה של הודעות בין קבוצות חברים.

להלן תיאור של הפרוטוקול כפי שניתן בתרגול.

```
Command: NICK
Parameter: < nickname >
NICK message is used to give user a nickname or change the previous one. If nickname already exist for another
client the server issues an ERR NICKNAMEINUSE to the client and drop the NICK command.
Numeric Replies: (see section 4.6 for numeric replies details)
ERR NONICKNAMEGIVEN ERR NICKNAMEINUSE
RPL NICKACCEPTED
Example:
```

NICK moshe23 ; Introducing new nick \moshe23".

Command: USER
Parameter: < username >
The USER message is used at the beginning of connection (immediately after the NICK command) to specify the username. A username may be more than one word.
Numeric Replies:
ERR NEEDMOREPARAMS ERR ALREADYREGISTRED
RPL USERACCEPTED
Examples:
USER Israel Israeli ; registering user Israel Israeli

Command: QUIT
Parameters: [< Quitmessage >]
A client session is ended with a quit message. The server must close the connection to a client which sends a QUIT message and inform the channel the user has left. If a \Quit Message" is given, this will be sent instead of the default message which is \< nickname >" has left the channel". If, for some reason, a client connection is closed without the client issuing a QUIT command (e.g. client dies), the server is required to _ll in the quit message with some sort of message reecting the nature of the event which caused it to happen.
Numeric Replies: None.
Examples:
QUIT Gone to have lunch

Command: JOIN
Parameters: < channel >
The JOIN command is used by client to start listening to a speci_c channel. If a JOIN is successful, the user is then sent the list of users who are on the channel, which must include the user joining. (equal to the reply received from 'NAMES < channel >' described later).
NOTE: JOIN a channel that you are already in should be ignored by the server.
Numeric Replies:
ERR NEEDMOREPARAMS RPL NAMREPLY
RPL ENDOFNAMES
Examples:
JOIN #foobar ; join channel #foobar.

Command: PART
Parameters: < channel >
The PART message causes the client sending the message to be removed from the channel
6 listed in the parameter string. If a client tries to leave a channel that its not connected to an ERR NOSUCHCHANNEL should be issues.
Numeric Replies:
ERR NEEDMOREPARAMS ERR NOSUCHCHANNEL
RPL PARTSUCCESS

Command: NAMES
Parameters: [< channel >]
By using the NAMES command, a user can list all nicknames on the server. If the <channel> parameter is given only nicknames on that channel are returned.
Numeric Replies:
RPL NAMREPLY RPL ENDOFNAMES
ERR NOSUCHCHANNEL
Examples:
NAMES #twilight zone ; list users on #twilight zone channel
NAMES ; list all users

Command: LIST
Parameters: None.
The list message is used to list all the open channels on the server.
Numeric Replies:
RPL LISTSTART RPL LISTEND
RPL LIST

Command: KICK
Parameters: < nickname >
The KICK command can be used to forcibly remove a user from a channel. It 'kicks 7 them out' of the channel (forced PART). Only a channel operator may kick another user out of a channel.
Numeric Replies:
ERR NEEDMOREPARAMS RPL USERKICKED
ERR CHANOPRIVSNEEDED
Examples:
KICK moshe23 ; KICK moshe23 from the channel

א. הגדירו את הממשק IRC הממש java.rmi.Remote וכן את הפונקציונאליות של פרוטוקול זה [10 נקודות]

ב. נניח כי קיימת מחלקה IRCImpl המממשת את הממשק IRC.
הגדירו שרת IRC על בסיס ממשק זה, וכתבו את ה command lines הדרושים להרצתו [10 נקודות]

שאלה 5 (10 נקודות)

במועד א' הגדרנו מודל נתונים עבור פרויקט גנזים, המכיל פרטים שונים על כתבי היד שנמצאו בגניזה הקהירית:

- מספר קטלוגי של כתב היד
- הנושא בו עוסק כתב היד
- קישור לסריקה של כתב היד
- פרטי המחברים של כתב היד, אם הם ידועים: שם, תאריך לידה, מקום מגורים

להלן תיאור מודל הנתונים, כפי שניתן בפתרון של מועד א.

```
CREATE TABLE doc (
   id INTEGER PRIMARY KEY,
   subject VARCHAR(30),
   link VARCHAR(30),
   authorId INTEGER);

ALTER TABLE doc
   ADD FOREIGN KEY (authorId)
   REFERENCES author(id);

CREATE TABLE author (
   id INTEGER PRIMARY KEY,
   name VARCHAR(30),
   birthdate VARCHAR(30),
   address VARCHAR(30));
```

א. במודל הקיים לכל כתב יד יש נושא אחד. עם התקדמות הפרויקט זוהה הצורך לאפיין על ידי מספר נושאים את כל אחד מכתבי היד. בנוסף, יתכנו כתבי יד ללא כל נושא. עדכנו את מודל הנתונים בהתאם [4 נקודות]

ב. כתבו שאילתת SQL המחזירה עבור כל נושא את מקומות המגורים של מחברי כתבי היד הקשורים לנושא זה, ממוינים על פי הנושא, ועל פי המספר הקטלוגי של כתב היד (כמפתח מיון משני). [6 נקודות]