תאריך הבחינה: 10.02.2022

שם המרצים: פרופ אנדרי שרף

ד"ר מני אדלר

ד"ר מרינה קוגן-סדצקי

שם הקורס: תכנות מערכות

מספר הקורס: 202-1-2031

שנה: 2022 סמסטר: א' מועד: א

משך הבחינה: **שעתיים וחצי**

חומר עזר: **אין**

- במבחן זה מספר שאלות, הניקוד שלהן לא בהכרח אחיד
- שימו לב: במבחן זה אין אופציה לכתוב "לא יודע/ת" , זה לא מזכה בניקוד
- השאלות הן רבי ברירה. יש לבחור רק תשובה אחת שהיא הכי נכונה
- יש **לסמן בצורה ברורה** את התשובות

בהצלחה !

1. (**7 נקודות**) נתון קטע הקוד הבא:

```
class A {
public:
   int _a;
   A(int a): _a(a) {}
   A& operator=(const A& other) { _a = other._a;  return *this; }
};

class B : public A {
public:
   int _b;
   B(int a, int b): A(a),_b(b) {}
   B(const B& other) :A(other._a) { _b = other._b; }
   B& operator=(const B& other) { _b = other._b; return *this;  }
};

int main() {
   B b1(1,2);
   B b2(3,4);
   b1 = b2;
   return 1;
}
```

מהם הערכים של השדות של האובייקט b1 לאחר ההשמה בשורה 3 (השורה המודגשת) של ()main ?
סמנו את כל התשובות הנכונות בדף התשובות:

א. a=3,b=4
ב. a=1,b=2
ג. a=1,b=4
ד. לא ניתן לדעת. ההשמה לא תתבצע כי שורת קוד זו גורמת ל (run-time error)
ה. לא ניתן לדעת. ההשמה לא תתבצע כי שורת קוד זו גורמת ל(compilation error)

2. (**6 נקודות**) נתונה המחלקה A, ונתונות שתי פונקציות f ו- g המוגדרות באופן הבא:

```
class A {
public:
   int _i=4;
};
```

```
void g(A& a) { cout << a._i; }
void f(A&& a) { g(a); }
```

א. ההגדרה של g שגויה, הפרמטר צריך היה להיות מוגדר עם שני סימנים && כמו בפונקציה f.
ב. ההגדרה של f שגויה, רק move constructor ו- move assignment operator יכולים לקבל פרמטר מהסוג הזה.
ג. הגדרה של f שגויה, לא ניתן להעביר הלאה (כלומר לפונקציה אחרת) ארגומנט מהסוג הזה.
ד. אין בעיה בהגדרות של f ושל g.

3. (**6 נקודות**) נתונה המחלקה A
(SomeClass היא מחלקה כלשהי שאופן מימושה אינו רלבנטי לשאלה)

```
class A {
private:
   SomeClass _x;
public:
   A(const SomeClass& x): _x(x) {}
};
```

לפניכם מספר דרכים להגדיר המחלקה B, היורשת את A:

(i)
```
class B: public A {
private:
   SomeClass _y;
public:
    B(const SomeClass& x, const SomeClass& y): _x(x), _y(y) {}
};
```

(ii)
```
class B: public A {
private:
   SomeClass _y;
public:
    B(const SomeClass& x, const SomeClass& y): A(x),_y(y) {}
};
```

(iii)
```
class B: public A {
private:
   SomeClass _y;
public:
```

```
    B(const SomeClass& x, const SomeClass& y): {   A(x); _y = y; }
};
```

(iv)

```
class B: public A {
private:
    SomeClass _y;
public:
    B(const SomeClass& x, const SomeClass& y): A(x) { _y = y; }
};
```

סמנו את כל התשובות הנכונות:

א. ישנן שתי אפשרויות תקינות אך לא באותה מידה של יעילות
ב. כל האפשרויות תקינות ויעילות באותה מידה
ג. כל האפשרויות תקינות אך לא באותה מידה של יעילות
ד. ישנן שתי אפשרויות תקינות ויעילות באותה מידה
ה. ישנה רק אפשרות אחת תקינה

4. (**6 נקודות**) נתון קטע הקוד הבא:

```
class A {
public:
  virtual void f() { cout << "A::f\n"; }
  void g() { cout << "A::g\n"; f(); }
  virtual ~A() {cout << "A::~\n"; }
}
class B: public A {
public:
  virtual void f() { cout << "B::f\n"; }
  void g() { cout << "B::g\n"; f(); }
  virtual ~B() {cout << "B::~\n"; }
}
int main() {
  A *obj = new B();
  obj->g();
  delete obj;
  return 1;
}
```

מה יקרה כשננסה לקמפל ולהריץ את קטע הקוד?

א. התכנית תדפיס:

A::g
B::f
B::~
A::~

ב. התכנית תדפיס:

B::g
B::f
B::~
A::~

ג. התכנית תדפיס:

B::g
B::f
A::~
B::~

ד. התכנית תדפיס:

A::g
B::f
A::~
B::~

ה. התכנית לא תתקמפל או תתקמפל אך תהיה שגיאת זמן ריצה (תלוי גרסת הקומפילר)

5. (**8 נקודות**) התוכנית Test1 בנספח מריצה שני Threads שמשימתם היא החלפת הערכים של שני אובייקטים נתונים מטיפוס Even:

ציינו אילו מהערכים הבאים אפשריים עבור e1, e2, בסיום הריצה של שני ה THREADS :

א. כל האפשרויות נכונות
ב. e1=2, e2=4
ג. e1=4, e2=2
ד. e1=4, e2=4
ה. e1=2, e2=2
ו. אף תשובה לא נכונה

6. (**10 נקודות**) התוכנית Test1 בנספח מריצה שני Threads שמשימתם היא החלפת הערכים של שני אובייקטים נתונים מטיפוס Even. ציינו האם הרצת התוכנית Test1 בנספח מקיימת תנאי נכונות ובטיחות ע"פ הנלמד בכיתה

א. ההרצה בטוחה ונכונה
ב. ההרצה בטוחה אך לא נכונה
ג. ההרצה לא בטוחה ולא נכונה
ד. ההרצה לא בטוחה אך נכונה
ה. לא ניתן לקבוע

7. (**6 נקודות**) התוכנית Test2 בנספח מריצה שני Threads המבצעים משימת החלפה של שני אובייקטים מטיפוס Even.
איזו מהטענות נכונה?

א. הקוד מקיים נכונות ולא ייתכן deadlock
ב. הקוד אינו מקיים נכונות אך לא ייתכן deadlock
ג. הקוד מקיים נכונות אך יתכן deadlock
ד. הקוד אינו מקיים נכונות ובנוסף ייתכן deadlock

8. (**6 נקודות**) התוכנית Test3 בנספח מריצה שני Threads המבצעים משימת החלפה של שני אובייקטים מטיפוס Even.
איזו מהטענות נכונ?

א. הקוד מקיים נכונות ולא ייתכן deadlock
ב. הקוד מקיים נכונות אך יתכן deadlock
ג. הקוד אינו מקיים נכונות ובנוסף ייתכן deadlock
ד. הקוד אינו מקיים נכונות אך לא ייתכן deadlock

9. (**6 נקודות**) נתון שרת בתבנית Command-Invocation המאפשר הרצת פקודות על מבנה נתונים מטיפוס NewsFeed

```
public interface NewsFeed {
    List<String> fetch(String category);
    void publish(String category, String news);
}
```

נתונים חמישה לקוחות המחוברים לשרת מסוג זה.
מצאו כמה ת'רדים ירוצו (בשרת) בכל אחת מהרצות השרת הבאות?

```
NewsFeed feed = new NewsFeedImpl();
```

```
new SingleThreadedServer(7777,
        ()->new new RemoteCommandInvocationProtocol<NewsFeed>(feed),
        ()->new ObjectEncoderDecoder()).serve();

new ThreadPerClientServer(7777,
        ()->new new RemoteCommandInvocationProtocol<NewsFeed>(feed),
        ()->new ObjectEncoderDecoder()).serve();

new FixedThreadPoolServer(7777,
        10,
        ()->new new RemoteCommandInvocationProtocol<NewsFeed>(feed),
        ()->new ObjectEncoderDecoder()).serve();
```

א. 1,6,11
ב. 2,6,11
ג. 1,5,10
ד. 2,5,10
ה. 1,6,10
ו. 2,6,10

10. (**9 נקודות**) אילו מהקביעות הבאות נכונות עבור שרת פקודות שרץ כ FixedThreadPoolServer?

```
public interface NewsFeed {
   List<String> fetch(String category);
   void publish(String category, String news);
}

NewsFeed feed = new NewsFeedImpl();
new FixedThreadPoolServer(7777,
        10,
        ()->new new RemoteCommandInvocationProtocol<NewsFeed>(feed),
        ()->new ObjectEncoderDecoder()).serve();
```

בדיון שנערך בכיתה עלו הקביעות הבאות על ידי הסטודנטים:

1. הלקוח לא צריך להכיר את מימוש הממשק NewsFeed
2. השרת תחת פרוטוקול TCP לא מבצע בהכרח את פקודות הלקוח על פי הסדר שבו הן נשלחו
3. לא ניתן לעצב מחדש את השרת לפי פרוטוקול UDP במקום TCP
4. מספר סוגי הפקודות (מימושי ה Command) שהלקוח יכול לשלוח לשרת אינו מוגבל כל עוד השרת מכיר את המימוש שלהן
5. המחלקה NewsFeedImpl חייבת להיות thread-safe

אילו מהקביעות נכונות?

א. קביעות 1, 4, 5 נכונות
ב. קביעות 3, 4 נכונות
ג. קביעות 2, 3 נכונות
ד. קביעות 1, 2, 5 נכונות
ה. קביעות 2, 3, 5 נכונות

11. (**7 נקודות**) אחד הסטודנטים בקורס הציע להחליף את ה ActorThreadPool בקוד הריאקטור שנלמד בכיתה ב ExecutorService רגיל.

```
public class Reactor {
...
private final ExecutorService pool;
...
private void handleReadWrite(SelectionKey key) {
      NonBlockingConnectionHandler handler = (NonBlockingConnectionHandler)
key.attachment();
      if (key.isReadable()) {
         Runnable task = handler.continueRead();
         if (task != null)
            pool.submit(task);
      }
      if (key.isWriteable())
         handler.continueWrite();
   }
 }
...
}
```

```
public class NonBlockingConnectionHandler {
   ...
   public Runnable continueRead() {
      ...
      return () -> {
            try {
               while (buf.hasRemaining()) {
                  T nextMessage = encdec.decodeNextByte(buf.get());
                  if (nextMessage != null) {
                     T response = protocol.process(nextMessage);
                     if (response != null) {
                        writeQueue.add(ByteBuffer.wrap(encdec.encode(response)));
                        reactor.updateInterestedOps(chan,
```

```
                    SelectionKey.OP_READ | SelectionKey.OP_WRITE);
                } // end if (response != null)
              } // end if (nextMessage != null)
            } // end while
          } finally {
            releaseBuffer(buf);
          }
      };
      ...
  }
  ...
}
```

בדיון שנערך בכיתה עלו הקביעות הבאות על ידי הסטודנטים:

1. ההודעות שהלקוח שלח לא בהכרח יתבצעו
2. ייתכן שהשרת יבצע עבור הלקוח הודעה שהוא לא שלח
3. ההודעות לא יבוצעו בהכרח על פי הסדר שבו הן נשלחו
4. השרת יבצע כתקנן את ההודעות בסדר שבו הן נשלחו אך ביעילות נמוכה יותר מקוד הריאקטור שנלמד בכיתה.
5. השרת יבצע כתקנן את ההודעות בסדר שבו הן נשלחו באותה יעילות כמו קוד הריאקטור שנלמד בכיתה

אילו מהקביעות נכונות?
א. קביעות 2, 4 נכונות
ב. קביעות 2, 3 נכונות
ג. קביעות 1, 2, 4 נכונות
ד. קביעות 1, 2, 5 נכונות
ה. קביעות 1, 2, 3 נכונות

12. (**8 נקודות**) אחד הסטודנטיות בקורס הציעה לסנכרן את הגישה ההדדית של הת'רדים ב executor לאותו לקוח באופן הבא:

```
class Reactor {
   ..
  private final ExecutorService pool;
  private Map<NonBlockingConnectionHandler, Queue<Runnable>> handler2tasks =
      new ConcurrentHashMap<NonBlockingConnectionHandler, Queue<Runnable>>();
   ..

  private void handleAccept(ServerSocketChannel serverChan, Selector selector) throws
IOException {
      SocketChannel clientChan = serverChan.accept();
      clientChan.configureBlocking(false);
      final NonBlockingConnectionHandler handler = new NonBlockingConnectionHandler(
            readerFactory.get(),
            protocolFactory.get(),
            clientChan,
            this);
      clientChan.register(selector, SelectionKey.OP_READ, handler);
      handler2tasks.put(handler, new ConcurrentLinkedQueue<Runnable>());
   }

   private void handleReadWrite(SelectionKey key) {
      NonBlockingConnectionHandler handler = (NonBlockingConnectionHandler)
key.attachment();
      if (key.isReadable()) {
         Runnable task = handler.continueRead();
         if (task != null) {
            Queue<Runnable> tasks = handler2tasks.get(handler);
            tasks.add(task);
            pool.submit(() -> {
                  handler.lock();
                  tasks.remove().run();  // remove the first item of the queue and apply run
                  handler.unlock();
            });
         }
      }
       if (key.isWriteable())
         handler.continueWrite();
   }
}
```

```
public class NonBlockingConnectionHandler {
   public AtomicBoolean key = new AtomicBoolean(true);
   public void lock() { while (!key.compareAndSet(true,false)); }
   public void unlock() { key.set(true); }
   ...
   public Runnable continueRead() {
      ...
      return () -> {
            try {
               while (buf.hasRemaining()) {
                  T nextMessage = encdec.decodeNextByte(buf.get());
                  if (nextMessage != null) {
                     T response = protocol.process(nextMessage);
                     if (response != null) {
                        writeQueue.add(ByteBuffer.wrap(encdec.encode(response)));
                        reactor.updateInterestedOps(chan,
                              SelectionKey.OP_READ | SelectionKey.OP_WRITE);
                     } // end if (response != null)
                  } // end if (nextMessage != null)
               } // end while
            } finally {
               releaseBuffer(buf);
            }
      };
      ...
  }
  ...
}
```

אילו מהקביעות הבאות נכונות?

א. השרת יבצע כתקנן את ההודעות בסדר שבו הן נשלחו אך ביעילות נמוכה יותר מקוד הריאקטור שנלמד בכיתה

ב. ההודעות שהלקוח שלח לא בהכרח יתבצעו

ג. ייתכן שהשרת יבצע עבור הלקוח הודעה שהוא לא שלח

ד. ההודעות לא יבוצעו בהכרח על פי הסדר שבו הן נשלחו

ה. השרת יבצע כתקנן את ההודעות בסדר שבו הן נשלחו באותה יעילות כמו קוד הריאקטור שנלמד בכיתה

13. (**5 נקודות**) נדרש להגדיר מודל נתונים לשמירת המידע הנדרש לשם קביעת תורים לחיסון קורונה:

אנשים: מספר זהות, גיל, מחלות רקע

מחלות רקע: סוג המחלה, רמת תיעדוף לקביעת תור

תורים: תאריך, חולה, מספר חיסון (ראשון או שני)

לשם כך הוגדרו הטבלאות הבאות:

```
CREATE TABLE Patients(Id INTEGER PRIMARY KEY,
                      Name Text,
                      Age INTEGER);
CREATE TABLE Background(Disease Text PRIMARY KEY,
                        Priority INTEGER);
CREATE TABLE PatientBackground(Patient INTEGER,
                               Disease Text,
                               PRIMARY KEY(Patient, Disease));
CREATE TABLE Queue(Patient INTEGER,
                   Day DATETIME,
                   Num INTEGER,
                   PRIMARY KEY(Patient, Num));
```

אילו מהקביעות הבאות נכונות?
א. מאחר שלא הוגדרו מפתחות זרים לא ניתן להשתמש במודל הנתונים לאחסון המידע הנדרש
ב. מאחר שלא הוגדרו מפתחות זרים לא ניתן להגדיר שדות בכלי ערך ייחודי (UNIQUE) בטבלאות
ג. מאחר שלא הוגדרו מפתחות זרים ניתן להוסיף רשומות לא תואמות לטבלאות השונות
ד. כל התשובות נכונות

14. (**5 נקודות**) נתונות הטבלאות הבאות:

Patients

17	Dinna	18
89	Danny	65
5	Ringo	30

Background

Cancer	5
Diabetes	1

Queue

89	1/1	1
17	15/1	1
89	22/1	2

PatientBackground

17	Cancer
89	Diabetes

כמה רשומות מחזירה השאילתה הבאה (INNER JOIN ו JOIN זה אותו דבר):

```
SELECT Patients.name, PatientBackground.Disease, Queue.Day, Queue.Num
FROM (Patients INNER JOIN PatientBackground ON Patients.id = PatientBackground.patient)
INNER JOIN Queue ON PatientBackground .patient = Queue.Patient
```

א. 3
ב. 2

ג. 4
ד. אף תשובה אינה נכונה

15. (**5 נקודות**) עבור הגרסה האחרונה של ה Application Persistence שמומשה בכיתה (הקוד עם פונקציית ה ORM), אילו מהקביעות הבאות נכונות:

א. הוספת טבלה בבסיס הנתונים דורשת הגדרה של מחלקת DAO חדשה עבורה
ב. הוספת שדות בטבלה בבסיס הנתונים דורשת עדכון של ה DTO המתאים
ג. שינוי טיפוס של שדה קיים בטבלה דורשת עדכון של ה DTO המתאים
ד. לא נדרש שום שינוי בקוד כאשר בסיס הנתונים משתנה

נספח: הקוד לשאלות בנושא מקביליות

```
class Even{
  //@INV get() %2 == 0
  private  int _val;

  Even(int val) throws Exception {
     set(val);
 }
 public synchronized int get() { return _val; }
 public synchronized void set(int val) throws Exception {
     if (val % 2 != 0)
        throw new Exception("Odd value!");
     _val = val;
 }
}

—-----------------------------------------------------------

public class Swapper1  implements Runnable{
   Even _e1,_e2;
   Swapper1(Even e1, Even e2) { _e1=e1; _e2=e2; }
   public void run()  {
      int tmp = _e1.get();
      _e1.set(_e2.get());
      _e2.set(tmp);
   }
}

class Test1 {
     public static void main(String[] args) throws Exception {
        Even e1 = new Even(2);
        Even e2 = new Even(4);
        new Thread(new Swapper1(e1,e2)).start();
        new Thread(new Swapper1(e1,e2)).start();
     }
}

—-------------------------------------------------------------------------------------

public class Swapper2 implements Runnable{
   Even _e1,_e2;
   Swapper2(Even e1, Even e2) { _e1=e1; _e2=e2; }
   public void run() {
```

```
      synchronized(_e1) {
         synchronized(_e2) {
            int tmp = _e1.get();
            _e1.set(_e2.get());
            _e2.set(tmp);
         }
      }
   }
}

class Test2 {
    public static void main(String[] args) throws Exception {
        Even e1 = new Even(2);
        Even e2 = new Even(4);
        new Thread(new Swapper2(e1,e2)).start();
        new Thread(new Swapper2(e2,e1)).start();
    }
}

—-----------------------------------------------------------------------------------

public class Swapper3 implements Runnable{
   Even _e1,_e2;
   final Object lock = new Object();
   Swapper3(Even e1, Even e2) { _e1=e1; _e2=e2; }
   public void run() {
       synchronized(lock) {
             tmp = _e1.get();
             _e1.set(_e2.get());
             _e2.set(tmp);
       }
    }
}

class Test3 {
    public static void main(String[] args) throws Exception {
        Even e1 = new Even(2);
        Even e2 = new Even(4);
        new Thread(new Swapper3(e1,e2)).start();
        new Thread(new Swapper3(e2,e1)).start();
    }
}
```

נספח: הקוד לשאלות בנושא DB

```
CREATE TABLE Patients(Id INTEGER PRIMARY KEY,
                      Name Text,
                      Age INTEGER);
CREATE TABLE Background(Disease Text PRIMARY KEY,
                        Priority INTEGER);
CREATE TABLE PatientBackground(Patient INTEGER,
                               Disease Text,
                               PRIMARY KEY(Patient, Disease));
CREATE TABLE Queue(Patient INTEGER,
                   Day DATETIME,
                   Num INTEGER,
                   PRIMARY KEY(Patient, Num));
```

פתרון:

1 - ג
2 - ד
3 - א
4 - א
5 - ב,ג,ד,ה
6 - ב
7 - ג
8 - ד
9 - א
10 - א
11 - ה
12 - א
13 - ג
14 - א
15 - ב