גיליון תשובות

מספר נבחן:_____

Q1	Q2
Q3	Q4
Q5	Q6
Q7	TOTAL

שאלה 1 (20 נקודות)

סעיף א (10 נקודות)

```
public synchronized boolean test() {
      int tmpPrev = prev;
      int tmpCurr = curr;
      int tmp;

      while (tmpPrev >= 0 && tmpCurr >=0) {
         if (tmpPrev == 0 && tmpCurr == 1)
            return true;

         tmp = tmpCurr;
         tmpCurr = tmpPrev;
         tmpPrev -= tmp;
      }



      return false;
}
```

סעיף ב (10 נקודות)

```
class Fib {

      private int prev;
      private int curr;

      Fib() {
            prev = 0;
            curr = 1;
      }

      Fib(int p,int c) {
            prev = p;
            curr = c;
      }

      public int synchronized getValue() {
            return prev;
      }
```

```
      public Fib next() {
            return new Fib(curr,prev+cur);
      }
}
```

שאלה 2 (10 נקודות)

```
public Class Summer {

   private int sum;   // Automatically initialized to zero.

   public synchronized void up() {
      sum++;
      if (sum == 1)
         notifyAll();
      System.out.print(sum);
   }
   public synchronized void down() {
      while (sum == 0)
          wait();
      sum--;
      System.out.print(sum);
   }
}
```

שאלה 3 (10 נקודות)

סעיף א (5 נקודות)

במחלקה A משתנה מטיפוס B הצריך להיות מאותחל בשלב איתחול השדות עם default constructor של מחלקה B שאינו קיים.

סעיף ב (5 נקודות)

אפשרות א:

```
class A {
public:
    A(): b(0){}
private:
    B b;
};
```

אפשרות ב:

```
class B {
public:
    B(int i = 0){ m_i = i;}
private:
    int m_i;
};
```

שאלה 4 (20 נקודות)

סעיף א (5 נקודות)

```
Test
a1.sum = 1
a1.sum = 3
double a3 = 10
```

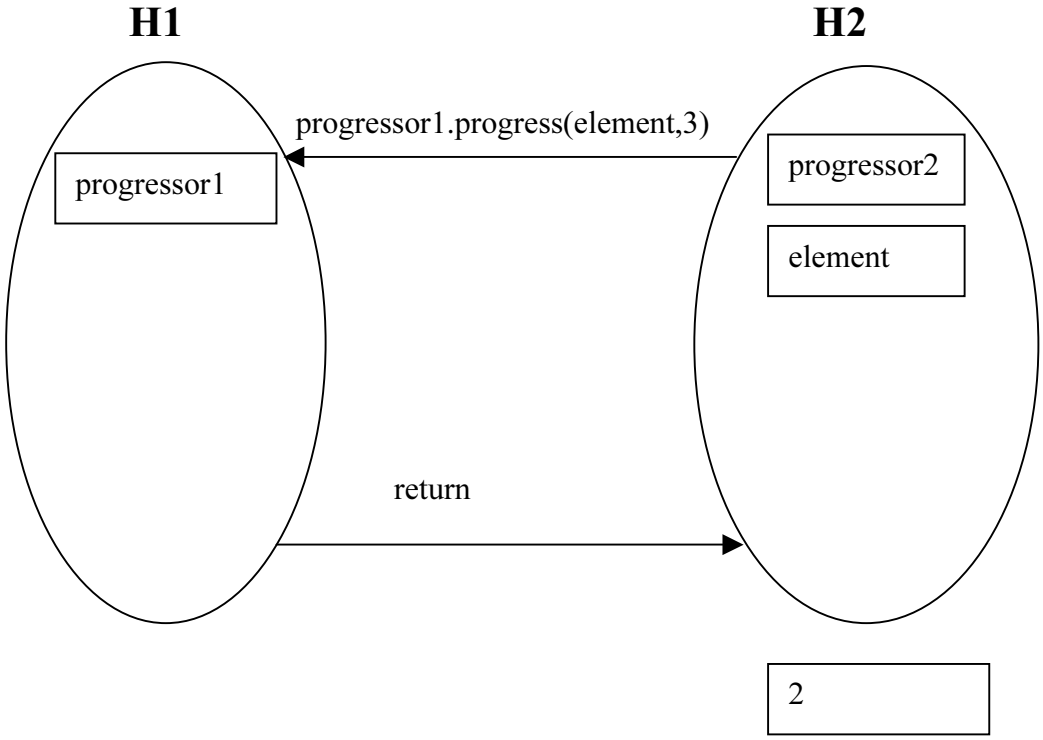
סעיף ב (15 נקודות)

```
#include <iostream>

class A {
protected:
  int x_;
private:
   A* a_;
public:
  A(int x) : x_(x), a_(0) {}

// Destructor
  virtual ~A() {}

// Define setA
   virtual setA(A* a) { a_ = a; }

// Define sum
   virtual int sum() {
      if (a_ == NULL)
         return x_;
      else
         return (x_ + a_->sum());
   }

// Define doubled
   virtual int doubled() {
      return (x_ + x_);
   }
};

class B : public A {
protected:
  int y_;

public:
// Constructor
   B(int x,int y): A(x), y_(y) {}

// Destructor
   virtual ~B() {}

// Define Doubled
   virtual int doubled() {
      return (2 * (x_ + y_));
   }
};
void main() {
  std::cout << "Test" << std::endl;
```

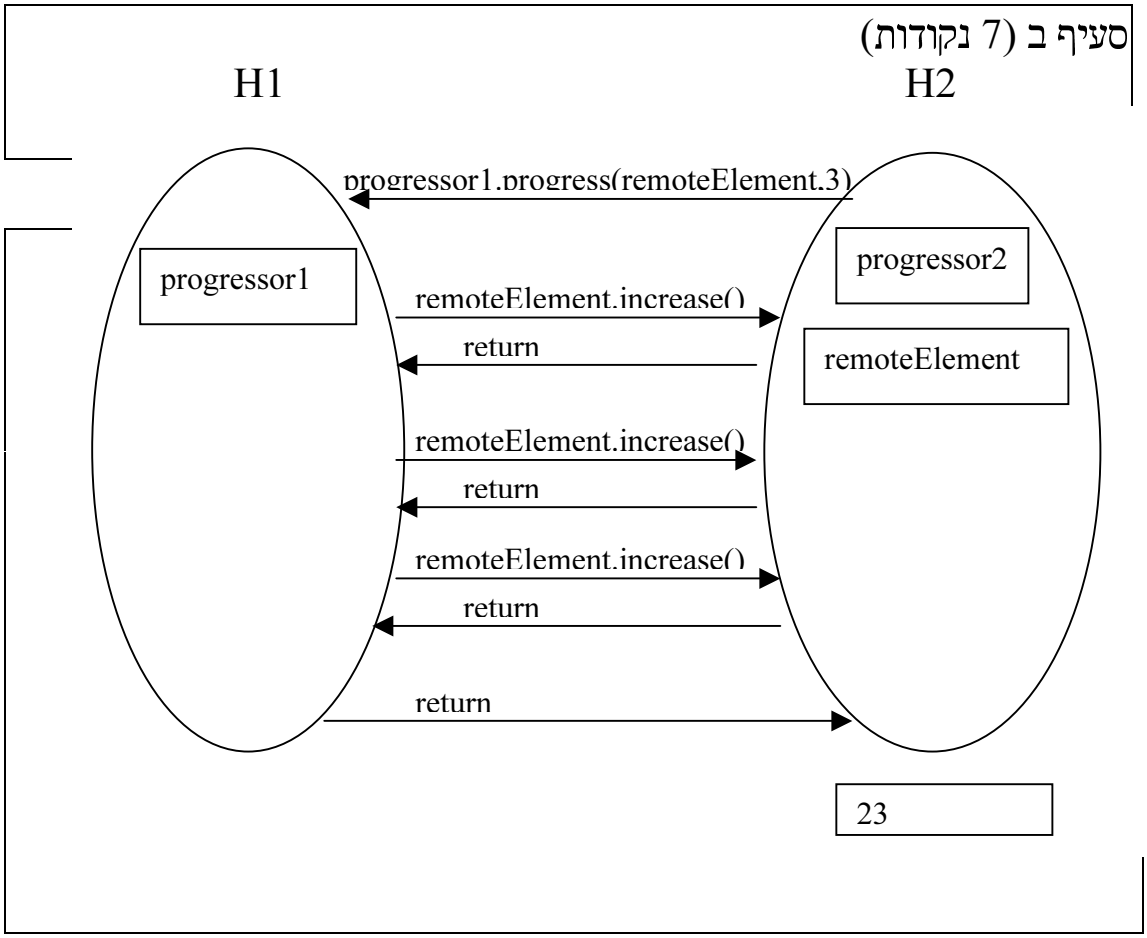
```
    A* a1 = new A(1);
    std::cout << "a1.sum = " << a1->sum() << std::endl;
    A* a2 = new A(2);
    a1->setA(a2);
    std::cout << "a1.sum = " << a1->sum() << std::endl;
    A* a3 = new B(2,3);
    std::cout << "doubled a3 = " << a3->doubled() << std::endl;

    delete a1;
    delete a2;
    delete a3;
}
```

שאלה 5 (14 נקודות)

סעיף א (7 נקודות)



סעיף ב (7 נקודות)



שאלה 6	(16 נקודות)

Time Server

Model 4: The client sends a "clock read" request (an empty UDP message) to the server. Upon the request message arrival, the server reads its clock, and sends the client a (UDP) message that includes the value of the server's clock. Message lost (requests or replays) are considered; for each server the client sets a timer for un-replayed requests. Upon timeout, client sends another request to the server.
Model 1: The client connects to the server. Upon connection establishment the server reads its clock, and writes to the client the value of the server's clock. Note that TCP may retransmit the message because of packet loss and therefore the client can read an out-of-date clock value.
We do not use the **pipelining** technique, because at any time there should be at most one request message.

Video Streaming Server

Model 4: The server sends the video stream to a multicast address. A client that wishes to receive the video stream joins the multicast group.
The **pipelining** technique is used to buffer message so that the user receives a continuous video stream despite of message decoding.

Database Server

Model 3: A DBMS is both CPU and IO bounded. Therefore, we use different threads to process different tasks (e.g., disk, network, query processing).
We use the **pipelining** technique extensively; incoming SQL commands are stored, a buffer stores outgoing results, priority queues schedule internal processing task. Note that, we use transactions (a synchronization technique) to can assure data consistency.

Web Server

Model 3: An http (the Internal World Wide Web protocol) server is IO bounded. Therefore, we use different threads to process different tasks (e.g., disk, network).
We use the **pipelining** technique extensively; incoming http commands are stored, and buffers stores outgoing HTML files.

שאלה 7 (10 נקודות)

Table: Movies Primary Key: Id
Id
Name
Director
Year
Length

Table: Schedule Primary Key: Data+Time Foreign Key: MovieId
Date
Time
MovieId

צריך למתוח קו משדה Id בטבלה Movies לשדה MovieId בטבלה Schedule (לא הצלחתי לעשות זאת ב Word...) באופן הבא:

1 n

Id MovieId

```
SELECT      Schedule.Date,
            Schedule.Time,
            Movies.Id,
            Movies.Name,
            Movies.Director,
            Movies.Year,
            Movies.Length

FROM        Movies, Schedule

WHERE Movies.Id = Schedule.MovieId   AND
      Schedule.Date = '1/1/2003'
```

ולמי שרוצה:

```
ORDER BY    Schedule.Date, Schedule.Time
```