Rešenja prvog kolokvijuma iz Operativnih sistema 2
Septembar 2012.

1. (10 poena)

```
const int MAXPRI = …;
extern PCB* idle;

class Scheduler {
public:
  Scheduler ();
  PCB* get ();
  void put (PCB*);
private:
  PCB* head[MAXPRI+1];
  PCB* tail[MAXPRI+1];
};

Scheduler::Scheduler () {
  for (int i=0; i<MAXPRI+1; i++)
    head[i]=tail[i]=0;
}

PCB* Scheduler::get () {
  for (int i=0; i<=MAXPRI; i++)
    if (head[i]) {
      PCB* ret = head[i];
      head[i] = head[i]->next;
      if (head[i]==0) tail[i]=0;
      ret->next = 0;
      return ret;
    }
  return idle;
}

void Scheduler::put (PCB* pcb) {
  if (pcb==0) return; // Exception!
  int p = pcb->priority;
  if (p<0 || p>MAXPRI) return; // Exception!
  pcb->next = 0;
  if (tail[p]==0)
    tail[p] = head[p] = pcb;
  else
    tail[p] = tail[p]->next = pcb;
}
```

2. (10 poena)

```
type Data = ...;

monitor server;
  export write, read;

  var d : Data,
      readyToRead, readyToWrite : boolean,
      waitToRead, waitToWrite : condition;

  procedure write (data : Data);
  begin
    while (not readyToWrite) do waitToWrite.wait();
    readyToWrite:=false;
    d:=data;
    readyToRead:=true;
    waitToRead.signal();
  end;

  procedure read (var data : Data);
  begin
    while (not readyToRead) do waitToRead.wait();
    readyToRead:=false;
    data:=d;
    readyToWrite:=true;
    waitToWrite.signal();
  end;

begin
  readyToRead:=false;
  readyToWrite:=true;
end; (* server *)
```

3. (10 poena)

```
import java.io.*;
import java.net.*;
import java.util.*;

public class Server {
 static final int N = ...;
 static int count = N;
 static LinkedList<Socket> blockedList = new LinkedList<Socket>();

 public static void main(String[] args) {
  try {
   ServerSocket sock = new ServerSocket(1033);

   while (true) {
    Socket clientSocket = sock.accept();
    BufferedReader in = new BufferedReader(new
InputStreamReader(clientSocket.getInputStream()));
    String request = in.readLine();

    if (request.equals("GetToken")) {
     if(count>0){
      sendMsgToClient(clientSocket, "Continue");
      count--;
     }
     else blockedList.addLast(clientSocket);
    }else if (request.equals("ReturnToken")) {
     if(blockedList.isEmpty()) count++;
```

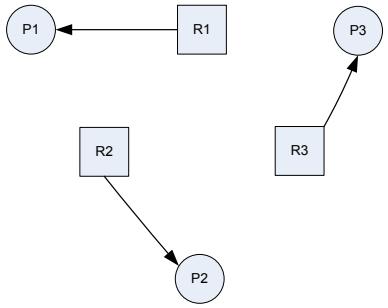
```
     else  sendMsgToClient(blockedList.poll(), "Continue");
    }
   }
  } catch (Exception e) {
   System.err.println(e);
  }
 }
 static void sendMsgToClient(Socket clientSocket,String msg) throws
UnknownHostException, IOException {
  PrintWriter newOut = new
PrintWriter(clientSocket.getOutputStream(),true);
  newOut.println(msg);
  clientSocket.close();
 }
}

public class Client {
 public static void main(String[] args) {
  try {
   while (true) {
    Socket srvSocket = new Socket("localhost", 1033);
    sendMsq(srvSocket,"GetToken");
    BufferedReader in = new BufferedReader(new
InputStreamReader(srvSocket.getInputStream()));
    System.out.println(in.readLine());
    srvSocket.close();
    //do Something...
    Thread.sleep(1000);

    srvSocket = new Socket("localhost", 1033);
    sendMsq(srvSocket,"ReturnToken");
    srvSocket.close();
   }
  } catch (Exception e) {
   System.err.println(e);
  }
 }
 private static void sendMsq(Socket srvSocket, String msg) throws
UnknownHostException, IOException {
  PrintWriter out = new PrintWriter(srvSocket.getOutputStream(), true);
  out.println(msg);
  }
}
```

4. (10 poena)

a)(5)



b)(5) Na primer: P1.request(R2), P2.request(R3), P3.request(R1)