Rešenja prvog kolokvijuma iz Operativnih sistema 2 Septembar 2015.

1. (10 poena)

```
class Scheduler {
public:
  Scheduler ();
  PCB* get ();
  void put (PCB*, int wasBlocked);
private:
  PCB* head[2];
  PCB* tail[2];
  int active;
};

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

void Scheduler::put (PCB* pcb, int wasBlocked) {
  if (pcb==0) return; // Exception!
  int p;
  if (wasBlocked) p = active;
  else p = 1-active;
  // Put pcb in the corresponding queue:
  pcb->next = 0;
  if (tail[p]==0)
    tail[p] = head[p] = pcb;
  else
    tail[p] = tail[p]->next = pcb;
}

PCB* Scheduler::get () {
  if (head[active]==0) active = 1-active;
  PCB* ret = head[active];
  if (ret==0) return 0; // Exception, should never happen!
  head[active] = head[active]->next;
  if (head[active]==0)
    tail[active]=0;
  ret->next = 0;
  return ret;
}
```

2. (10 poena)

```
monitor FlipFlop;
  export flip, flop;

  var i : integer,
      cond : condition;

  procedure flip ();
  begin
    while i<N do cond.wait;
    i:=0;
    (* do flip *)
  end;

  procedure flop ();
  begin
    (* do flop *)
    if i<N then
      i:=i+1;
    cond.signal;
  end;

begin
  i:=N;
end; (* monitor *)
```

3. (10 poena)

```
public class President {
    private static int nextId = 0;
    int price = 0, newPrice = 0;
    int winner = 0, finished = 0, users = 0, round = 0;
    private boolean end = false;
    public synchronized int getId() { return President.nextId++; }
    public synchronized void addNewUser() {users++;}
    public synchronized void addNewPrice(int p, int id) {
        if (newPrice < p) {
            newPrice = p;
            winner = id;
        }
        finished++;
        if (finished == users) {
            if (newPrice == price) end = true;
            else {
                price = newPrice;
                finished = 0;
            }
            round++;
            notifyAll();
        }
    }
    public synchronized void removeUser() {
        users--;
        if (users == 0) System.exit(0);
        else notifyAll();
    }
    public synchronized boolean waitRound(int nextRound)
                                throws InterruptedException {
        while (nextRound != round && !end) wait();
        return end;
    }
```

```
    public static void main(String[] args) {
        President auction = new President();
        try {
            ServerSocket sock = new ServerSocket(1033);
            while (true) {
                Socket clientSocket = sock.accept();
                new ServerThread(clientSocket, auction).start();
            }
        } catch (Exception e) { System.err.println(e); }
    }
}
public class ServerThread extends Thread {
    private Socket client;
    private President president;
    private int id , nextRound;
    public ServerThread(Socket c, President p)
    { client = c; president = p; }
    private int getPrice() {
        synchronized (president) {
            nextRound = president.round + 1;
            return president.price;
        }
    }
    public void run() {
        try {
            president.addNewUser();
            id = president.getId();
            BufferedReader in = new BufferedReader(
                        new InputStreamReader(client.getInputStream()));
            PrintWriter out = new
                    PrintWriter(client.getOutputStream(),true);
            String request = in.readLine();
            if (request.equals("userRequest")) {
                out.println(getPrice());
            } else {
                client.close();
                return;
            }
            while (true) {
                int value;
                request = in.readLine();
                System.out.println(id + " received: " + request);
                if (request.equals("no")) value = 0;
                else value = Integer.parseInt(request);
                president.addNewPrice(value, id);
                if (president.waitRound(nextRound)) break;
                out.println(getPrice());
            }
            if (president.winner == id) out.println("winner");
            else out.println("end");
            client.close();
        } catch (Exception e) {
            System.err.println(e);
        } finally { president.removeUser(); }
    }
}
```