Rešenja prvog kolokvijuma iz Operativnih sistema 2
Decembar 2014.

1. (10 poena) HP, MP, LP, MP, HP, HP, MP, LP, MP, LP

2. (10 poena)

```
monitor TaxiDispatcher;
  export userRequest, taxiAvailable;

  var waitingUsers, availableTaxis : integer;
      waitForUser, waitForTaxi : condition;
      taxiID : integer;

  procedure userRequest (var assignedTaxiID : integer);
  begin
    if (availableTaxis>0)
      begin
        availableTaxis:=availableTaxis-1;
        waitForUser.signal;
        assignedTaxiID:=taxiID;
      end
    else
      begin
        waitingUsers:=waitingUsers+1;
        waitForTaxi.wait;
        assignedTaxiID:=taxiID;
      end;
  end;

  procedure taxiAvailable (myID : integer);
  begin
    if (waitingUsers>0)
      begin
        waitingUsers:=waitingUsers-1;
        taxiID:=myID;
        waitForTaxi.signal;
      end
    else
      begin
        availableTaxis:=availableTaxis+1;
        waitForUser.wait;
        taxiID:=myID;
      end;
  end;

begin
  waitingUsers:=0; availableTaxis:=0;
end;
```

3. (10 poena)

```
public class TaxiDispatcher {
 static LinkedList<Socket> blockedTaxi = new LinkedList<Socket>();
 static LinkedList<Socket> blockedUsers = 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("userRequest")) {
     if(!blockedTaxi.isEmpty()){
      sendMsgToClient(blockedTaxi.remove(), "Continue");
      sendMsgToClient(clientSocket, "Continue");
     }
     else blockedUsers.add(clientSocket);
    }else if (request.equals("taxiAvailable")) {
     if(!blockedUsers.isEmpty()){
      sendMsgToClient(blockedUsers.remove(), "Continue");
      sendMsgToClient(clientSocket, "Continue");
     }
     else blockedTaxi.add(clientSocket);
    }
   }
  } 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();
 }
}
```