Rešenja prvog kolokvijuma iz Operativnih sistema 2 novembar 2021.

1. (10 poena)

```
class GlobalScheduler {
public:
  GlobalScheduler () {}

  void register (Scheduler* s, unsigned priClass) { scheds[priClass] = s; }
  PCB* get ();
  void put (PCB*, bool wasBlocked);

private:
  Scheduler* scheds[N_SCHED_CLASS] = {};
  static const unsigned maxPri[N_SCHED_CLASS];
};

void GlobalScheduler::put (PCB* pcb, bool wasBlocked) {
  if (pcb==0) return; // Exception!
  for (unsigned priClass = 0; priClass < N_SCHED_CLASS; priClass++)
    if (pcb->pri <= maxPri[priClass]) {
      if (!scheds[priClass]) // Exception;
      if (wasBlocked &&
         ((priClass>0) ?
            (pcb->pri>maxPri[priClass-1]+1):(pcb->pri>0)) pcb->pri--;
      if (!wasBlocked && pcb->pri<maxPri[priClass]) pcb->pri++;
      scheds[priClass]->put(pcb,wasBlocked);
      return;
    }
  // Exception;
}

PCB* Scheduler::get () {
  for (int i=0; i<N_SCHED_CLASS; i++) {
    if (!scheds[i]) continue;
    PCB* p = scheds[i]->get();
    if (p) return p;
  }
  return 0;
}
```

2. (10 poena)

a)(5)

```
class Condition {
public:
  Condition (Semaphore* mutex) : mux(mutex), cond(0) {}
  void wait () {  Semaphore::signalWait(mux,&cond); mux->wait(); }
  void signal () { if (cond.value()<0) cond.signal(); }
private:
  Semaphore *mux, cond;
};
```

b)(5)

```
void BoundedBuffer::put (Data* d) {
  Mutex dummy(&mutex);
  if (numberInBuffer==BUFFER_SIZE) slotAvailable.wait();
  numberInBuffer++;
  ...
  itemAvailable.signal();
}

Data* BoundedBuffer::get () {
  Mutex dummy(&mutex);
  if (numberInBuffer==0) itemAvailable.wait();
  numberInBuffer--;
  Data* d = ...;
  ...
  slotAvailable.signal();
  return d;
}
```

3. (10 poena)

```
public class Server {
    private Map<String, Data> allData = new HashMap<String, Data>();

    public void run() {
        ServerSocket serverSocket = null;

        try {
            serverSocket = new ServerSocket(5555);

            while (true) {
                Socket clientSocket = serverSocket.accept();
                Service service = new Service(clientSocket);

                Thread t = new RequestHandler(this, service);
                t.start();

            }
        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    public void update(String name, String data) {
                Data dd;
                synchronized(this) {
                        if (!allData.containsKey(name)) {
                                allData.put(name, new Data());
                        }
                        dd = allData.get(name);
                }

                synchronized(dd) {
                        dd.update(data);
                }
    }

    public synchronized void draw(String name) {
                Data dd;
                synchronized(this) {
                        dd = allData.get(name);
                }
```

```
                synchronized(dd) {
                        dd.draw();
                }
    }

    public static void main(String[] args) {
        new Server().run();
    }
}

public class RequestHandler extends Thread {
    private final Server server;
    private final Service service;

    public RequestHandler(Server server, Service service) {
        this.server = server;
        this.service = service;
    }

    public void run() {
        while (true) {
            try {
                String msg = service.receiveMsg();
                if (msg == null) {
                    break;
                }

                execute(msg);

                service.sendMsg("Ok");
            } catch (IOException e) {
                e.printStackTrace();
                break;
            }

        }

        service.close();
    }

    private void execute(String msg) {
        // #name#op#data#
        String[] data = msg.split("#");
        String name = data[1];
        String op = data[2];
        String dd = data[3];
        int ret;
        if ("update".equals(op)) {
           server.update(name, dd);
        } else {
            server.draw(name);
        }

    }
}
Klasa Service data je na vežbama.
```