Prvi kolokvijum iz Operativnih sistema 1
Septembar 2016.

1. (10 poena)

```
void transfer () {
  // Start the controller:
  *ioCtrl = 1;

  // Perform the output transfer:
  for (int i=0; i<BUFSIZE; i++) {
      sleep(DELAY); // Sleep-wait
      *ioData = buffer[i]; // Write data
  }

  // Stop the controller:
  *ioCtrl = 0;
}
```

2. (10 poena) a)(7)

```
class Join {
public:
  Join (Thread* thread1, Thread* thread2);
  int wait ();
private:
  Thread* thread[2];
  int arrived[2];
};

Join::Join (Thread* t1, Thread* t2) {
  this->thread[0] = t1; this->thread[1] = t2;
  this->arrived[0] = this->arrived[1] = 0;
}

int Join::wait () {
  if (Thread::running!=this->thread[0] && Thread::running!=this->thread[1])
    return -1; // Exception
  lock();
  int ret;
  int myself = (Thread::running==this->thread[0])?0:1;
  int other = 1-myself;
  this->arrived[myself] = 1;
  if (this->arrived[other]) {
    ret = 1;
    Scheduler::put(this->thread[other]);
    this->arrived[0] = this->arrived[1] = 0;
  } else {
    ret = 0;
    jmp_buf old = Thread::running->context;
    Thread::running = Scheduler::get();
    jmp_buf new = Thread::running->context;
    if (setjmp(old)==0) longjmp(new,1);
  }
  unlock();
  return ret;
}
```

3. (10 poena)

```
class DataProcessor : public Thread {
public:
  DataProcessor (int offs, int sz) : offset(offs), size(sz) {}
  virtual void run ();
private:
  int offset, size;
};

void DataProcessor::run () {
  int end = this->offset+this->size;
  for (int i=this->offset; i<end; i++)
      processData(&data[i]);
}

void parallelProcessing (int n) {
   int chunk = N/n;
   int offset = 0;
   for (int i=0; i<n; i++) {
     int myChunk = chunk;
     if (i<N%n) myChunk++;
     new DataProcessor(offset,myChunk)->start();
     offset+=myChunk;
   }
}
```