Prvi kolokvijum iz Operativnih sistema 1
Septembar 2014.

1. (10 poena)

```
static const unsigned int timeout = 50; // 50 ms
static int completed = 0;
static REG* ptr = 0;
static unsigned int count = 0;

void transfer (REG* buffer, unsigned int cnt) {
  // Initialize transfer
  completed = 0;
  ptr = buffer;
  count = cnt;

  // Start transfer:
  *ioCtrl = 1; // Input request
  *timer = timeout; // Start timer

  while (!completed); // Busy wait for transfer completion
}

interrupt void timerInterrupt () {
  *ptr++ = *ioData; // Read data
  if (--count) {
    *ioCtrl = 1; // New input request
    *timer = timeout; // Restart timer
  } else  // Completed
    completed = 1;
}
```

2. (10 poena)

a)(5) VA: Segment(8):Page(16):Offset(8); PA: Frame(20):Offset(8).

b)(5) FF00DDh

3. (10 poena) a)(7)

```
dispatch:  ; Save the current context
           LOAD  Rx,Rp       ; Rx:=# of current processor
           SHL   Rx,2        ; Rx:=Rx*4
           LOAD  Rx,#runningProcesses[Rx]; Rx:=&running process' PCB
           STORE #offs_r0[Rx],R0   ; store R0
           STORE #offs_r1[Rx],R1   ; store R1
           …
           STORE #offs_psw[Rx],PSW ; store PSW
           STORE #offs_sp[Rx],SP   ; store SP
           ; Call scheduler
           CALL  schedule
           ; Restore the next context
           LOAD  Rx,Rp       ; Rx:=# of current processor
           SHL   Rx,2        ; Rx:=Rx*4
           LOAD  Rx,#runningProcesses[Rx]; Rx:=&next process' PCB
           LOAD  R0,#offs_r0[Rx]   ; restore R0
           LOAD  R1,#offs_r1[Rx]   ; restore R1
           …
           LOAD  PSW,#offs_psw[Rx] ; restore PSW
           LOAD  SP,#offs_sp[Rx]   ; restore SP
           RTS                     ; return from subroutine
```

b)(3) Nije potrebno, jer svaki procesor jedini pristupa samo svom odgovarajućem elementu *n* niza `runningProcesses`, pa nema potencijalnog konflikta između procesora (nema deljenih podataka u memoriji kojima pristupaju različiti procesori u ovoj operaciji).

4. (10 poena)

```
class Search : public Thread {
public:
  Search (int a[], int i, int j, int x) : array(a), ii(i), jj(j), xx(x)  {}
protected:
  virtual void run () { find(ii,jj); }
  void find (int i, int j);
private:
  int *array, ii, jj, xx;
};

void Search::find (int i, int j) {
  if (array==0 || i<0 || j<0 || j<i) return;
  if (i==j) {
    if(array[i]==x)
      printf("Found at %d!\n",i);
    else
      printf("Not found at %d!\n",i);
    return;
  }
  int k = (i+j)/2;
  Thread* t = new Search(array,k+1,j,x);
  t->start();
  find(i,k);
}
```