Prvi kolokvijum iz Operativnih sistema 1
Odsek za softversko inženjerstvo
Mart 2014.

1. (10 poena)

```
void transfer () {
  if (ioHead==0) return; // Exception
  *dmaAddress = ioHead->buffer;
  *dmaCount = ioHead->size;
  *dmaCtrl = 1; // Start I/O
}

interrupt void dmaInterrupt () {
  if (ioHead==0) return; // Exception
  if (*dmaStatus&2) // Error in I/O
    ioHead->status = -1;
  else
    ioHead->status = 0;
  ioHead = ioHead->next; // Remove the request from the list
  if (ioHead==0) { // No more requests
    *dmaCtrl = 0;
    return;
  }
  // Start a new transfer
  *dmaAddress = ioHead->buffer;
  *dmaCount = ioHead->size;
  *dmaCtrl = 1; // Start I/O
}
```

2. (10 poena)

Virtual address (hex)	Mapping result (hex)
12FA	2502FA
C0FF	D67FF
1675	X
B014	P
CDAB	X

3. (10 poena)

```
yield:      ; Save current context
            push  r0
            load  r0,#cur[sp]
            store r1,#offsR1[r0] ; save r1
            pop   r1  ; save r0 through r1
            store r1,#offsR0[r0]
            store r2,#offsR2[r0] ; save other regs
            store r3,#offsR3[r0]
            ...
            store r31,#offsR31[r0]
            store psw,#offsPSW[r0] ; save psw
            store sp,#offsSP[r0] ; save sp

            ; Restore new context
            load  r0,#nxt[sp]
            load  sp,#offsSP[r0] ; restore sp
            load  psw,#offsPSW[r0] ; restore psw
            load  r31,#offsR31[r0] ; restore r31
            ...  ; restore other regs
            load  r1,#offsR1[r0]
            load  r0,#offsR0[r0] ; restore r0
            ; Return
            ret
```

4. (10 poena)

```
void main (int argc, char* const argv[]){
  if (argc<2) return; // Exception!

  // Get the argument value:
  int myArg = 0;
  sscanf(argv[1],"%d",&myArg);
  if (myArg<=0) return;

  // Prepare the arguments for the child:
  char childArg[10];  // The value of the second argument
  sprintf(childArg,"%d",myArg-1);
  char* childArgs[3];
  childArgs[0] = argv[0];
  childArgs[1] = childArg;
  childArgs[2] = NULL;

  // Create a child:
  int status = fork();
  if (status<0) exit(); // Exception!
  if (status==0) // Child's context
    execvp(argv[0],childArgs);
  else {  // Parent's context
    printf("%s\n",myArg);
    wait(NULL);
  }
}
```