Drugi kolokvijum iz Operativnih sistema 1
Avgust 2024.

1. (10 poena)

```
class Thread {
public:
   void start () { pthread_create(&myThr,nullptr,runThread,this); }
protected:
   Thread () {}
   virtual ~Thread() { pthread_join(&myThr,nullptr); }
   virtual void run ();
private:
   pthread_t myThr;
   static void* runThread (void*);
};

void* Thread::runThread (void* t) {
   ((Thread*)t)->run();
   return nullptr;
}
```

2. (10 poena)

```
bool int_occurred[ivt_size]) = {}; // Interrupt occurred indicators

interrupt void int_handler (unsigned short entry) {
  int_occurred[entry] = true;
}

void handle_interrupts () {
  for (unsigned short i=0; i<ivt_size; i++)
    if (int_occurred[i]) {
      int_occurred[i] = false;
      ivt[i]();
    }
}

int sys_call (int num, void* args) {
  int ret = scvt[num](args);
  handle_interrupts();
  Thread* oldT = Thread::running;
  Thread* newT = Scheduler::get();
  Thread::yield(oldT,newT);
  return ret;
}
```

3. (10 poena)

```
#include <pthread>
#include <fcntl.h>
#include <sys/stat.h>
#include <semaphore.h>
#include <cstddef>
using std::size_t;

extern void read_data (char* buffer, size_t size);
extern void write_data (const char* buffer, size_t size);
extern void process_data (const char* in_buf, char* out_buf, size_t size);

const size_t buffer_size = 256;
char input_buffer[buffer_size], output_buffer[buffer_size];

pthread_t reader_thr, processor_thr, writer_thr;
sem_t *rrd_ib, *rwr_ib; // Ready to read/write input buffer
sem_t *rrd_ob, *rwr_ob; // Ready to read/write output buffer

void* reader_fn (void*) {
  while (true) {
    sem_wait(rwr_ib);
    read_data(input_buffer, buffer_size);
    sem_post(rrd_ib);
  }
}

void* writer_fn (void*) {
  while (true) {
    sem_wait(rrd_ob);
    write_data(output_buffer, buffer_size);
    sem_post(rwr_ob);
  }
}

void* processor_fn (void*) {
  while (true) {
    sem_wait(rrd_ib); sem_wait(rwr_ob);
    process_data(input_buffer, output_buffer, buffer_size);
    sem_post(rwr_ib); sem_post(rrd_ob);
  }
}

int main () {
  rrd_ib = sem_open("/rrd_ib",O_CREAT,O_RDWR,0);
  rwr_ib = sem_open("/rwr_ib",O_CREAT,O_RDWR,1);
  rrd_ob = sem_open("/rrd_ob",O_CREAT,O_RDWR,0);
  rwr_ob = sem_open("/rwr_ob",O_CREAT,O_RDWR,1);

  pthread_create(&reader_thr,nullptr,&reader_fn,nullptr);
  pthread_create(&processor_thr,nullptr,&processor_fn,nullptr);
  pthread_create(&writer_thr,nullptr,&writer_fn,nullptr);
  while (true);
}
```