Drugi kolokvijum iz Operativnih sistema 1
Maj 2023.

1. (10 poena)

```
#include <signal.h>
#include <stdlib.h>
#include <stdio.h>
#include <unistd.h>

#define handle_error(msg) \
  do { printf("Error: %s\n",msg); exit(-1); } while(0)

typedef void (*SIGHANDLER) (int);

volatile int complete = 0;

void handler (int) { complete = 1; }

int main () {
  SIGHANDLER oldh = signal(SIGTERM,handler);
  if (oldh==SIG_ERR) handle_error("Cannot set the signal handler.");

  pid_t pid = fork();

  if (pid<0) handle_error("Cannot create the child process.");

  else if (pid==0)
    while (1) {
      while (!complete);
      printf("Are your sure you want to exit? (y/n)");
      int c = getchar();
      if (c=='Y' || c=='y') exit(0);
      complete = 0;
    };

  else {
    signal(SIGTERM,oldh);
    kill(pid,SIGTERM);
  }
}
```

2. (10 poena)

```
void initContext (PCB* pcb, void* usrStk, void* krnlStk, void* startAddr) {
  uint32* kst = (uint32*)krnlStk;
  *(kst-0) = (uint32)startAddr;
  *(kst-1) = (uint32)usrStk;
  pcb->sp = kst-34;
}
```

3. (10 poena)

```
class Mutex {
public:
  Mutex () : holder(0) {}

  void wait ();
  void signal ();

private:
  Queue blocked;
  Thread* holder;
};

void Mutex::wait () {
  lock();
  if (Thread::running == holder) { unlock(); throw ERR_MUTEX; }
  if (holder==0)
    Scheduler::put(holder = Thread::running);
  else {
    blocked.put(Thread::running);
  }
  yield();
  unlock();
}

void Mutex::signal () {
  lock();
  if (Thread::running != holder) { unlock(); throw ERR_MUTEX; }
  Thread* t = blocked.get();
  holder = t;
  if (t) Scheduler::put(t);
  unlock();
}
```