Rešenja trećeg kolokvijuma iz Operativnih sistema 2, januar 2017.

1. (10 poena)

```
DiskRequest* DiskScheduler::get () {
  if (!edfHead || !scanHead) return 0;
  DiskRequest* req = 0;
  // Select the request:
  if (edfHead->deadline<=0)
    req = edfHead;
  else
    req = scanHead;
  // Remove it from the EDF queue:
  if (req->edfPrev) req->edfPrev->edfNext = req->edfNext;
  else edfHead = req->edfNext;
  if (req->edfNext) req->edfNext->edfPrev = req->edfPrev;
  req->edfPrev = req->edfNext = 0;
  // Remove it from the SCAN queue:
  if (req->scanNext!=req)
    scanHead = req->scanNext;
  else
    scanHead = 0;
  req->scanPrev->scanNext = req->scanNext;
  req->scanNext->scanPrev = req->scanPrev;
  req->scanNext = req->scanPrev = 0;
  return req;
}
```

2. (10 poena)

```
#!/bin/bash

if [ $# -ne 3 ]; then
    echo "Nedovoljan broj argumenata"
    exit 1
fi

sablon=$1
dir=$2
pravo=$3

old_IFS=$IFS
IFS=$'\n'
for i in $(find "$dir" -name '*'); do
    ime=$(echo $i | sed 's:.*/\([^/]*\)$:\1:')
    if echo $ime | grep $sablon; then
      if [ -f $i ]; then
          chmod a+$pravo $i
      fi
    fi
done
IFS=$old_IFS
```

3. (10 poena)

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

#define INPUT 0
#define OUTPUT 1
#define LEFT 0
#define RIGHT 1
#define NUM 6
#define ITERATIONS (1000)

int main() {
        int file_descriptors[2][NUM][2];
        for (int j = 0; j < 2; j++) {
                for (int i = 0; i < NUM; i++) {
                        pipe(file_descriptors[j][i]);
                }
        }

        int id = 0;
        for (int i = 1; i < NUM;i++) {
                id = i;
                if (fork() > 0) {
                        break;
                } else {
                        id = 0;
                }
        }

        double value = id + 1;
        double ratio = 0.5;
        int left = (NUM + id - 1) % NUM;
        int right = (id + 1) % NUM;
        for (int i = 0; i < ITERATIONS; i++) {
                double share = value * ratio;
                write(file_descriptors[LEFT][id][OUTPUT] , &share, sizeof(double));
                share = value * (1 - ratio);
                write(file_descriptors[RIGHT][id][OUTPUT], &share, sizeof(double));
                double left_share, right_share;
                read(file_descriptors[LEFT][left][INPUT],
                     &left_share, sizeof(double));
                read(file_descriptors[RIGHT][right][INPUT],
                     &right_share, sizeof(double));
                value = left_share + right_share;
                ratio = left_share / value;
        }
        printf("id=%d share=%.2f\n", id, value);

        for (int j = 0; j < 2; j++) {
                for (int i = 0; i < NUM; i++) {
                        close(file_descriptors[j][i][INPUT]);
                        close(file_descriptors[j][i][OUTPUT]);
                }
        }
        return 0;
```