

# Rešenja zadatka sa trećeg kolokvijuma iz Operativnih sistema 2, januar 2022.

## 1. (10 poena)

```
class DiskScheduler {
public:
    DiskScheduler () : head(0), cursor (0) {}

    Req* get () const { return cursor; }
    void put (Req* r);
    void remove ();

private:
    Req *head, *cursor;
};

inline void DiskScheduler::remove () {
    if (!cursor) return;
    if (cursor == head) {
        head = cursor = cursor->next;
        return;
    }
    for (Req* p=head; p; p=p->next)
        if (p->next == cursor) {
            p->next = cursor = cursor->next;
            if (!cursor) cursor = head;
            return;
        }
}

inline void DiskScheduler::put (Req* r) {
    if (!head) {
        r->next = 0;
        head = cursor = r;
        return;
    }
    if (r->block<head->block) {
        r->next = head;
        head = r;
        return;
    }
    for (Req* p=head; p; p=p->next)
        if (!p->next || r->block<p->next->block) {
            r->next = p->next;
            p->next = r;
            break;
        }
}
```

## 2. (10 poena)

```
#!/bin/bash
```

```
if [ $# -ne 1 -o ! -r $1 ]; then
    echo "Error: First parameter must be a readable file"
    exit 1
fi

duplicates=$(cat $1 | tr { } | cut -d} -f2 | sort | uniq -d)
count=$(echo -n $duplicates | wc -c)

if [ $count -gt 0 ]; then
    echo "Error: there are duplicated properties"
    echo $duplicates
    exit 2
fi

cat $1 | sed 's:.*{\(.*\)}.*{\(.*\)}.*:setProperty("\1", "\2");:'
```

## 3. (10 poena)

```
FILE* open_fifo(char *name, char *mode) {
    FILE * file;
    file = fopen(name, mode);
    if (file == NULL) {
        printf("Error in fopen");
        exit(1);
    }
    return file;
}

int main()
{
    char xfifo[] = "get_x";
    char yfifo[] = "get_y";
    mkfifo(xfifo, 0666);
    mkfifo(yfifo, 0666);

    if (fork() != 0) {
        FILE *gx = open_fifo(xfifo, "w");
        FILE *gy = open_fifo(yfifo, "r");

        for (int i = 0; i < 5; i++) {
            int x = getX();
            fwrite(&x, sizeof(x), 1, gx);
            fflush(gx);

            int y;
            fread(&y, sizeof(y), 1, gy);
            useY(y);
        }
        fclose(gx);
        fclose(gy);
    } else {
        FILE *gx = open_fifo(xfifo, "r");
        FILE *gy = open_fifo(yfifo, "w");

        for (int i = 0; i < 5; i++) {
            int x;
            fread(&x, sizeof(x), 1, gx);
        }
    }
}
```

```
        int y = useXGetY(x);
        fwrite(&y, sizeof(y), 1, gy);
        fflush(gy);
    }

    fclose(gx);
    fclose(gy);
}

return 0;
}
```