Rešenja zadataka za treći kolokvijum iz Operativnih sistema 1, septembar 2016.

1. (10 poena)

```
class IOStream {
public:
  IOStream (BlockIOHandle d) : dev(d), cursor(-1), curBlock(-1)
    { seek(0); }

  int seek (long offset);
  int getChar (char& c);
protected:
  int loadBlock(); // Helper; loads the block corresponding to the cursor
private:
  BlockIOHandle dev;
  char buffer[BlockSize];
  long curBlock, cursor;
};

int IOStream::loadBlock () {
  long blockNo = cursor/BlockSize;
  if (curBlock==blockNo) return 0;
  if (readBlock(dev,blockNo,buffer)<0)) {
    cursor = -1;
    return -1;
  }
  curBlock = blockNo;
  return 0;
}

int IOStream::seek (int offset) {
  cursor = offset;
  if (cursor<0 || cursor>=getSize(dev)) {
    cursor = -1;
    return -1;
  }
  return 0;
}

int IOStream::getchar (char& c) {
  if (cursor<0 || cursor>=getSize(dev)) {
    cursor = -1;
    return -1;
  }
  if (loadBlock()<0) return -1;
  c = buffer[(cursor++)%BlockSize];
  return 0;
}
```

2. (10 poena)

```
#include <stdio.h>

int fcopyreverse (const char* filenamefrom, const char* filenameto) {
  File* from = fopen(filenamefrom,"r");
  if (from==NULL) return -1;  // Error with opening input file
  File* to = fopen(filenameto,"w");
  if (to==NULL) return -2;  // Error with opening output file
  int ret = 0;
  ret = fseek(from,-1,SEEK_END);
  while (ret==0) {
    int c = fgetc(from);
    if (c==EOF) return -3; // Error with reading input file
    ret = fputc(c,to);
    if (ret==EOF) return -4; // Error with writing to output file
    ret = fseek(from,-2,SEEK_CUR);
  }
  ret = fclose(from);
  if (ret==EOF) return -5; // Error with closing input file
  ret = fclose(to);
  if (ret==EOF) return -6; // Error with closing output file
  return ret;
}
```

3. (10 poena)

```
void* getFileBlock (FCB* fcb, ulong lBlockNo) {
  if (fcb==0) return 0; // Exception
  ulong entry0 = lBlockNo/IndexSize;
  if (entry0>=IndexSize) return 0; // Exception
  ulong entry1 = lBlockNo%IndexSize;
  ulong index = fcb->index;
  if (index==0) return 0; // Exception
  ulong* index0 = (ulong*)getPBlock(index);
  if (index0==0) return 0; // Exception
  index = index0[entry0];
  if (index==0) return 0; // Exception
  ulong* index1 = (ulong*)getPBlock(index);
  if (index1==0) return 0; // Exception
  return getBlock(index1[entry1]);
}
```