Rešenja zadataka za treći kolokvijum iz Operativnih sistema 1 Jun 2015.

1. (10 poena)

```
int readBlock(int diskNo, BlkNo block, Byte* buffer) {
  if (diskNo<0 || diskNo>=MaxNumOfDisks) return -1; // Error
  if (disks[diskNo].isValid==0) return -2; // Error
  if (disks[diskNo].readBlock==0) return -3; // Error
  return (disks[diskNo].readBlock)(block,buffer);
}

int writeBlock(int diskNo, BlkNo block, Byte* buffer) {
  if (diskNo<0 || diskNo>=MaxNumOfDisks) return -1; // Error
  if (disks[diskNo].isValid==0) return -2; // Error
  if (disks[diskNo].writeBlock==0) return -3; // Error
  return (disks[diskNo].writeBlock)(block,buffer);
}

int registerDriver(int diskNo, DiskOperation read, DiskOperation write) {
  if (diskNo<0 || diskNo>=MaxNumOfDisks) return -1; // Error
  if (disks[diskNo].isValid) return -2; // Error
  disks[diskNo].isValid = 1;
  disks[diskNo].readBlock = read;
  disks[diskNo].writeBlock = write;
}
```

2. (10 poena)

```
int lock(FCB* f, unsigned int op) {
  if (f==0) return -1; // Exception!
  if ((op & OP_WR) && (!f->sharedLock && !f->exclLock))
    return f->exclLock = 1;
  if ((op & (OP_RD|OP_EX)) && !f->exclLock)
    return f->sharedLock = 1;
  return 0;
}
```

3. (10 poena)

a)(3)

```
void blockToBit(unsigned long blkNo, unsigned long& bt, byte& mask) {
  bt = blkNo/BITS_IN_BYTE;
  mask = 1<<(blkNo%BITS_IN_BYTE);
}

void bitToBlk(unsigned long& blkNo, unsigned long bt, byte mask) {
  blkNo = bt*BITS_IN_BYTE;
  for (; !(mask&1); mask>>=1) blkNo++;
}
```

b)(7)

```
void freeBlock (unsigned long blk) {
  if (blk>=NumOfBlocks) return;
  unsigned long bt = 0; byte msk = 0;
  blockToBit(blk,bt,msk);
  blocks[bt] &= ~msk;
}
```

```
unsigned long allocateBlock (unsigned long startingFrom) {
  unsigned long bt = 0; byte msk = 0;
  for (unsigned long blk = startingFrom; blk<NumOfBlocks; blk++) {
    blockToBit(blk,bt,msk);
    if ((blocks[bt]&msk)==0) {
      blocks[bt] |= msk;
      return blk;
    }
  }

  for (unsigned long blk =1; blk<startingFrom; blk++) {
    blockToBit(blk,bt,msk);
    if ((blocks[bt]&msk)==0) {
      blocks[bt] |= msk;
      return blk;
    }
  }
  return 0;
}
```

Nešto efikasnija implementacija:

```
unsigned long allocateBlock (unsigned long startingFrom) {
  if (startingFrom>=NumOfBlocks) return 0;
  unsigned long bt = 0; byte msk = 0;
  blockToBit(startingFrom,bt,msk);
  for (; bt<NumOfBlocks/BITS_IN_BYTE; bt++) {
    if (blocks[bt]==~0) continue; // All blocks in this byte are occupied
    unsigned long blk = 0;
    bitToBlock(blk,bt,~blocks[bt]);
    blockToBit(blk,bt,msk)
    blocks[bt] |= msk;
    return blk;
  }

  blockToBit(1,bt,msk); // blk 0 is reserved
  for (; bt<NumOfBlocks/BITS_IN_BYTE; bt++) {
    if (blocks[bt]==~0) continue; // All blocks in this byte are occupied
    unsigned long blk = 0;
    bitToBlock(blk,bt,~blocks[bt]);
    blockToBit(blk,bt,msk)
    blocks[bt] |= msk;
    return blk;
  }

  return 0;
}
```