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

#define NUM_BLOCKS 10

// Structure to represent a disk block
typedef struct DiskBlock {
    int blockNum;
    struct DiskBlock* nextBlock;
} DiskBlock;

// Structure to represent a file
typedef struct File {
    int fileNum;
    DiskBlock* firstBlock;
    DiskBlock* lastBlock;
} File;

// Structure to represent the directory
typedef struct Directory {
    File* files[NUM_BLOCKS];
} Directory;

// Function to initialize the directory
void initializeDirectory(Directory* dir) {
    for (int i = 0; i < NUM_BLOCKS; i++) {
        dir->files[i] = NULL;
    }
}

// Function to allocate blocks for a file
```

```
void allocateBlocks(Directory* dir, int fileNum, int numBlocks) {
    File* file = (File*)malloc(sizeof(File));
    file->fileNum = fileNum;
    file->firstBlock = NULL;
    file->lastBlock = NULL;

    for (int i = 0; i < numBlocks; i++) {
        DiskBlock* newBlock = (DiskBlock*)malloc(sizeof(DiskBlock));
        newBlock->blockNum = i;
        newBlock->nextBlock = NULL;

        if (file->firstBlock == NULL) {
            file->firstBlock = newBlock;
        } else {
            file->lastBlock->nextBlock = newBlock;
        }

        file->lastBlock = newBlock;
    }

    dir->files[fileNum] = file;
}

// Function to free blocks allocated to a file
void freeBlocks(Directory* dir, int fileNum) {
    if (dir->files[fileNum] == NULL) {
        printf("File %d does not exist\n", fileNum);
        return;
    }

    File* file = dir->files[fileNum];
```

```
    DiskBlock* currentBlock = file->firstBlock;
    DiskBlock* nextBlock;

    while (currentBlock != NULL) {
        nextBlock = currentBlock->nextBlock;
        free(currentBlock);
        currentBlock = nextBlock;
    }

    dir->files[fileNum] = NULL;
}

// Function to display the directory
void displayDirectory(Directory* dir) {
    printf("Directory:\n");

    for (int i = 0; i < NUM_BLOCKS; i++) {
        if (dir->files[i] != NULL) {
            printf("File %d: ", i);
            DiskBlock* currentBlock = dir->files[i]->firstBlock;
            while (currentBlock != NULL) {
                printf("%d -> ", currentBlock->blockNum);
                currentBlock = currentBlock->nextBlock;
            }
            printf("NULL\n");
        }
    }
}

int main() {
    Directory dir;
```

```
    initializeDirectory(&dir);

    allocateBlocks(&dir, 0, 4);
    allocateBlocks(&dir, 1, 3);
    displayDirectory(&dir);

    freeBlocks(&dir, 0);
    displayDirectory(&dir);

    return 0;
}
```