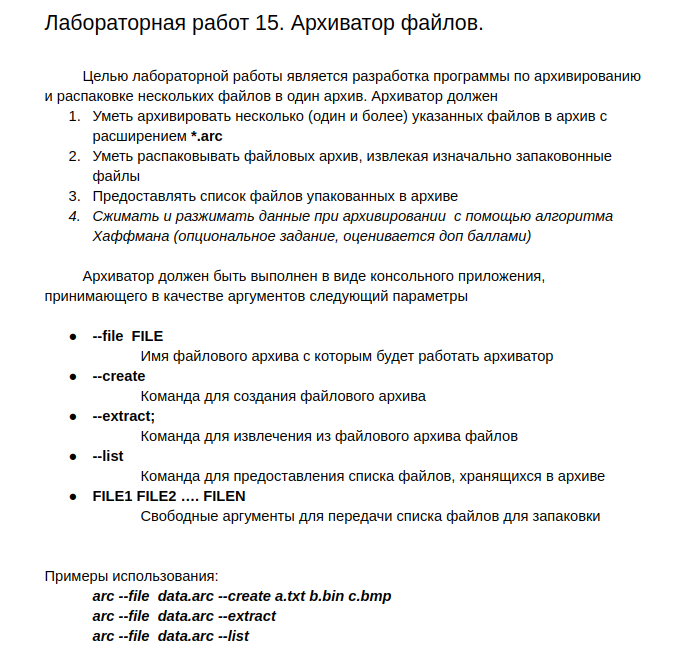
**Лабараторная работа 15**

Выполнил студент группы M3105

Клишевич Вадим Александрович

main.c

#include **"settings.h"**  
#include **"arch.h"**  
  
**int** main(**int** argc, **char** \*\*argv)  
{  
 Settings \*settings = setSettings(argc, argv);  
  
 **if** (settings->create) {  
 create(settings);  
 } **else if** (settings->extract) {  
 extract(settings);  
 } **else if** (settings->list) {  
 list(settings);  
 }  
}

settings.h

*//*  
*// Created by vadim on 3.12.19.*  
*//*  
  
#ifndef **LABA15\_SETTINGS\_H**  
#define **LABA15\_SETTINGS\_H**  
  
#pragma pack(push, 1)  
**typedef struct** tagSettings {  
 **char** \*archName;  
 **char** create;  
 **char** extract;  
 **char** list;  
 **int** filecount;  
 **char** \*fileNames[1000];  
} Settings;  
#pragma pack(pop)  
  
Settings \*setSettings(**int** argc, **char** \*\*argv);  
  
#endif *//LABA15\_SETTINGS\_H*

settings.c

*//*  
*// Created by vadim on 3.12.19.*  
*//*  
  
#include **<string.h>**  
#include **<stdlib.h>**  
#include **"settings.h"**  
  
Settings \*setSettings(**int** argc, **char** \*\*argv) {  
 Settings \*settings = malloc(**sizeof**(Settings));  
 **for** (**int** i = 1; i < argc; i++) {  
 **if** (strcmp(argv[i], **"--file"**) == 0) {  
 settings->archName = malloc(**sizeof**(**char**) \* strlen(argv[++i]));  
 strcpy(settings->archName, argv[i]);  
 } **else if** (strcmp(argv[i], **"--create"**) == 0) {  
 settings->create = 1;  
 } **else if** (strcmp(argv[i], **"--extract"**) == 0) {  
 settings->extract = 1;  
 } **else if** (strcmp(argv[i], **"--list"**) == 0) {  
 settings->list = 1;  
 } **else** {  
 settings->fileNames[settings->filecount] = malloc(**sizeof**(**char**) \* strlen(argv[i]));  
 strcpy(settings->fileNames[settings->filecount++], argv[i]);  
 }  
 }  
 **return** settings;  
}

files.h

*//*  
*// Created by vadim on 3.12.19.*  
*//*  
  
#ifndef **LABA15\_FILES\_H**  
#define **LABA15\_FILES\_H**  
  
#pragma pack(push, 1)  
**typedef struct** tagFile {  
 **int** nameSize;  
 **char** \*name;  
 **int** dataSize;  
 **char** \*data;  
} File;  
#pragma pack(pop)  
  
File \*readFile(**char** \*name);  
**void** writeFile(File \*file);  
  
#endif *//LABA15\_FILES\_H*

files.c

*//*  
*// Created by vadim on 3.12.19.*  
*//*  
  
#include **"files.h"**  
#include **<stdio.h>**  
#include **<stdlib.h>**  
#include **<string.h>**  
  
File \*readFile(**char** \*name) {  
 File \*file = malloc(**sizeof**(File));  
 file->nameSize = strlen(name) + 1;  
 file->name = malloc(**sizeof**(**char**) \* file->nameSize);  
 strcpy(file->name, name);  
 FILE \*input = fopen(name, **"r"**);  
 fseek(input, 0, **SEEK\_END**);  
 file->dataSize = ftell(input);  
 fseek(input, 0, **SEEK\_SET**);  
 file->data = malloc(**sizeof**(**char**) \* file->dataSize);  
 fread(file->data, file->dataSize, 1, input);  
 fclose(input);  
 **return** file;  
}  
  
**void** writeFile(File \*file) {  
 FILE \*output = fopen(file->name, **"w"**);  
 fwrite(file->data, file->dataSize, 1, output);  
 fclose(output);  
}

arch.h

*//*  
*// Created by vadim on 3.12.19.*  
*//*  
  
#ifndef **LABA15\_ARCH\_H**  
#define **LABA15\_ARCH\_H**  
  
#include **"files.h"**  
  
#include **"settings.h"**  
  
#pragma pack(push, 1)  
**typedef struct** tagArch {  
 **int** fileCount;  
 File \*\*files;  
} Arch;  
#pragma pack(pop)  
  
Arch \*readArch(Settings \*settings);  
**void** create(Settings \*settings);  
**void** extract(Settings \*settings);  
**void** list(Settings \*settings);  
  
#endif *//LABA15\_ARCH\_H*

arch.c

*//*  
*// Created by vadim on 3.12.19.*  
*//*  
  
#include **"arch.h"**  
#include **"settings.h"**  
#include **<stdio.h>**  
#include **<stdlib.h>**  
  
Arch \*readArch(Settings \*settings) {  
 FILE \*input = fopen(settings->archName, **"r"**);  
 Arch \*arch = malloc(**sizeof**(Arch));  
 fread(&arch->fileCount, **sizeof**(arch->fileCount), 1, input);  
 arch->files = malloc(**sizeof**(File \*) \* arch->fileCount);  
 **for** (**int** i = 0; i < arch->fileCount; i++) {  
 arch->files[i] = malloc(**sizeof**(File));  
 fread(&arch->files[i]->nameSize, **sizeof**(arch->files[i]->nameSize), 1, input);  
 arch->files[i]->name = malloc(**sizeof**(**char**) \* arch->files[i]->nameSize);  
 fread(arch->files[i]->name, arch->files[i]->nameSize, 1, input);  
 fread(&arch->files[i]->dataSize, **sizeof**(arch->files[i]->dataSize), 1, input);  
 arch->files[i]->data = malloc(**sizeof**(**char**) \* arch->files[i]->dataSize);  
 fread(arch->files[i]->data, arch->files[i]->dataSize, 1, input);  
 }  
 **return** arch;  
}  
  
**void** create(Settings \*settings) {  
 Arch \*arch = malloc(**sizeof**(Arch));  
 arch->fileCount = settings->filecount;  
 arch->files = malloc(**sizeof**(File \*) \* arch->fileCount);  
 **for** (**int** i = 0; i < arch->fileCount; i++) {  
 arch->files[i] = malloc(**sizeof**(File));  
 arch->files[i] = readFile(settings->fileNames[i]);  
 }  
  
 FILE \*output = fopen(settings->archName, **"w"**);  
 fwrite(&arch->fileCount, **sizeof**(arch->fileCount), 1, output);  
 **for** (**int** i = 0; i < arch->fileCount; i++) {  
 fwrite(&arch->files[i]->nameSize, **sizeof**(arch->files[i]->nameSize), 1, output);  
 fwrite(arch->files[i]->name, arch->files[i]->nameSize, 1, output);  
 fwrite(&arch->files[i]->dataSize, **sizeof**(arch->files[i]->dataSize), 1, output);  
 fwrite(arch->files[i]->data, arch->files[i]->dataSize, 1, output);  
 }  
 fclose(output);  
}  
  
**void** extract(Settings \*settings) {  
 Arch \*arch = readArch(settings);  
 **for** (**int** i = 0; i < arch->fileCount; i++) {  
 writeFile(arch->files[i]);  
 }  
}  
  
**void** list(Settings \*settings) {  
 Arch \*arch = readArch(settings);  
 **for** (**int** i = 0; i < arch->fileCount; i++) {  
 printf(**"%s "**, arch->files[i]->name);  
 }  
 printf(**"\n"**);  
}