

# Отчет по курсу “Операционные системы”

## Задание 2.01

Владимир Руцкий, 4057/2

21 декабря 2009 г.

### Постановка задачи

*“Напишите программу, вычисляющую суммарный размер всех файлов в указанном каталоге и его подкаталогах.”*

### Выбранный метод решения

1. Программа принимает на вход список каталогов (файлов), суммарный размер файлов в которых требуется вычислить.
2. Для вычисления суммарного размера файлов в каталоге используется рекурсивная процедура, принимающая на вход текущий обрабатываемый каталог (файл).
3. Процедура определяет тип обрабатываемого файла с помощью функции **stat()** и, если это файл, то возвращает его размер, а если это каталог, то вызывает себя от файлов в каталоге (считывание содержимого каталога осуществляется функциями **opendir()**, **readdir()** и **closedir()**).

### Исходный код

Исходный код 1: task\_2\_01.c

```
1 /* task_2_01.c
2  * Task 2.01 on Unix course.
3  * Vladimir Rutsky <altsysrq@gmail.com>
4  * 20.12.2009
5  */
6
7 #include <stddef.h>
8 #include <stdio.h>
9 #include <stdlib.h>
10 #include <string.h>
11 #include <sys/types.h>
12 #include <sys/stat.h>
13 #include <dirent.h>
```

```

14
15 static long long getFileSize( char const *fileName )
16 {
17     struct stat sb;
18
19     if (stat(fileName, &sb) == 0)
20     {
21         if (S_ISDIR(sb.st_mode))
22         {
23             /* Directory */
24             long long size = 0;
25             DIR *dp;
26
27             dp = opendir(fileName);
28             if (dp != NULL)
29             {
30                 struct dirent *ep;
31
32                 while ((ep = readdir(dp)) != NULL)
33                 {
34                     if (strcmp(ep->d_name, ".") != 0 && strcmp(ep->d_name, "..") !=
35                         0)
36                     {
37                         char *newName;
38
39                         if ((newName = malloc(sizeof(char) * (strlen(fileName) + 1 +
40                             strlen(ep->d_name) + 1))) != NULL)
41                         {
42                             long long dirSize = 0;
43                             sprintf(newName, "%s/%s", fileName, ep->d_name);
44
45                             dirSize = getFileSize(newName);
46                             size += dirSize;
47
48                             free(newName);
49                         }
50                         else
51                         {
52                             perror("malloc");
53                         }
54                     }
55                 }
56                 if (closedir(dp))
57                 {
58                     perror("closedir");
59                     fprintf(stderr, "Error: _closedir()_failed_on_'%s'.\n", fileName);
60                 }
61                 else
62                 {
63                     perror("opendir");
64                     fprintf(stderr, "Error: _opendir()_failed_on_'%s'.\n", fileName);
65                 }
66
67             /* Debug */
68             fprintf(stderr, "%lld\t%s\n", size, fileName);

```

```

68     return size;
69 }
70 else if (S_ISREG(sb.st_mode))
71 {
72     /* Regular file */
73
74     /* Debug */
75     fprintf(stderr, "%lld\t%s\n", (long long)sb.st_size, fileName);
76
77     return sb.st_size;
78 }
79 else
80 {
81     /* Not a regular file, nor a directory */
82
83     fprintf(stderr, "Not a regular file: '%s'.\n", fileName);
84 }
85 }
86 else
87 {
88     perror("stat");
89     fprintf(stderr, "Error: stat() failed on '%s'.\n", fileName);
90 }
91
92
93 return 0;
94 }
95
96 static int printFileSize( char const *fileName )
97 {
98     long long size = 0;
99
100     size = getFileSize(fileName);
101     //printf("%lld\t%s\n", (long long)size, fileName);
102
103     return 0;
104 }
105
106 int main( int argc, char const *argv[] )
107 {
108     if (argc <= 1)
109         return printFileSize(".");
110     else
111     {
112         int i;
113         for (i = 1; i < argc; ++i)
114         {
115             int result = printFileSize(argv[i]);
116             if (result != 0)
117                 return result;
118         }
119     }
120
121     return 0;
122 }

```