Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Сибирский государственный университет телекоммуникаций и
информатики»
(СибГУТИ)

Кафедра вычислительных систем

ОТЧЕТ
по практической работе 5

по дисциплине «**Программирование**»

Выполнил:
студент гр. ИВ-122
«24» мая 2022 г. _____ Клепче Г.В..

Проверил:
старший преподаватель
Кафедры ВС
«__» февраля 2022 г. _____ Фульман В.О.

Оценка «_____»

Новосибирск 2022

ОГЛАВЛЕНИЕ

ЗАДАНИЕ

Реализовать четыре основные функции для работы с путями(input(), check(), process(), output()). Реализовать необходимые функции для работы со строками (slen(), stok(), sspn(), scmp(), scpy()).

Входными данными для всех подпрограмм является строка, содержащая пути к файлам, разделенные указанным видом разделителей (может быть пробел, двоеточие, «+»). На вход могут также поступать дополнительные данные, определяемые вариантом задания.

Вариант 1

Файлы, расположенные в каталоге `dir1` были перемещены в каталог `dir2`. Для заданных с клавиатуры `dir1` и `dir2` обновить список входных файловых путей. `dir1` и `dir2` должны относиться к одной и той же ОС.

Вход:

```
delim: +
paths: C:\Windows\system32+C:\User\test+C:\Windows\explorer.exe+D:\Windows\Distrib
dir1: C:\Windows
dir2: E:\WindowsXP
```

ВЫПОЛНЕНИЕ РАБОТЫ

Создаём файлы main.c, strings.c, strings.h.

В strings.c – функции для работы ос строками и путями 2) main.c : инициализируем переменную типа char “delim” – разделитель путей, char* “path” – сюда будут записываться входные данные с клавиатуры, char* “result” – тут будут лежать преобразованные пути, dir – домашний каталог, username – имя пользователя.

Напишем функцию main.c:

```c
int main()
{
    char *path = malloc(sizeof(char) * 500);
    char *dir1 = malloc(sizeof(char) * 100);
    char *dir2 = malloc(sizeof(char) * 100);
    char delim;
    char *result = malloc(sizeof(char) * 1000);
    input(&delim, path, dir1, dir2);
    result = process(delim, path, dir1, dir2);
    output(result);
    free(path);
    free(dir1);
    free(dir2);
    free(result);
    return 0;
}
```

С помощью функции input() мы считываем с клавиатуры данные

Перменной result присваиваем результат работы функции process(), в которой мы будем разделять строку на подстроки с помощью фукции stok().

ПРИЛОЖЕНИЕ

string.h

```
1  #pragma once
2  #include <stdio.h>
3
4  int slen(char *string);
5
6  char* scpy(char* str1, char* str2);
7
8  int sspen(char chr, char *chars);
9
10 char *stch(const char *s, const char c);
11
12 int stok(char token, char* str, char** strs);
13
14 int scmp(char* str1, char* str2);
15
16 int check(char *string);
17
18 char *path_process(char *path, char *dir2,
   char *dir1);
```

string.c

```c
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include "strings.h"

int slen(char *string)
{
   char *buffer = string;
   while (*buffer != '\0')
      ++buffer;
   return buffer - string;
}

char *scpy(char *str1, char *str2)
{
   size_t len_str1_str2 = slen(str1) + slen(str2);
   char *str = malloc(sizeof(char) * len_str1_str2 * 2);

   char *x = str;

   while (*str1){
      *x = *str1;
      str1++;
      x++;
   }
   while (*str2){
      *x = *str2;
      str2++;
      x++;
   }

   return str;
}

int sspen(char chr, char *chars)
{
   for (int i = 0; i < slen(chars); i++)
      if (chr == chars[i]){
         return 1;
      }

   return 0;
}

int check(char *string)
{
   char banned[] = "\n:?*<>|\"\0";

   for (int i = 0; i < slen(string); i++){
```

```
50          if (sspen(string[i], banned))
51              return 1;
52      }
53      return 0;
54  }
55
56  int stok(char delim, char *str, char **strs)
57  {
58      int j = 0;
59      int amount = 0;
60      for (int i = 0; *str != '\0'; i++)
61      {
62          while (str[j] != delim && str[j] != '\0'){
63              j++;
64          }
65          if (j == 0){
66              str++;
67              i--;
68              continue;
69          }
70          strs[i] = malloc(sizeof(char) * (j*2));
71          for (int k = 0; k < j; k++){
72              strs[i][k] = str[k];
73          }
74          str += j + 1;
75
76          j = 0;
77          amount = i;
78      }
79
80      return amount + 1;
81  }
82
83  int scmp(char *str1, char *str2)
84  {
85      while (slen(str1) != 0 || slen(str2) != 0)
86      {
87          if (*str1 == *str2){
88              str1++;
89              str2++;
90              continue;
91          }
92          else{
93              if (*str1 > *str2){
94                  return 1;
95              }
96              else{
97                  return -1;
98              }
```

```
 99          }
100       }
101       return 0;
102    }
103
104    char *path_process(char *path, char *dir2, char *dir1)
105    {
106       char **strs = malloc(sizeof(char *) * 360);
107       char *chec;
108       int length = stok('\\', path, strs);
109       int i = 1;
110
111       while (i < length)
112       {
113          if (check(strs[i]))
114             return "";
115          i++;
116       }
117
118       chec = strs[0];
119       if (chec[0] == path[0]){
120          char *checkdir1 = strs[0];
121          strcat(checkdir1, "\\");
122          strcat(checkdir1, strs[1]);
123
124          int flag = 3;
125          for(i = 0;i < slen(checkdir1); i++)
126             if (dir1[i] == checkdir1[i])
127                flag = 1;
128             else {
129                flag = -1;
130                break;
131             }
132          i = 2;
133          if (flag == 1){
134             char *finale = malloc(sizeof(char) * 360);
135             finale = scpy(dir2, strs[0] - 1);
136
137             while (i < length){
138                finale = scpy(finale, "\\");
139                finale = scpy(finale, strs[i]);
140                i++;
141             }
142
143             while (i < slen(finale)){
144                if (finale[i] == '\n')
145                   for (int j = i; j < slen(finale); ++j)
146                      finale[j] = finale[j+1];
147                i++;
```

```
148             }
149             return finale;
150         }
151     }
152     return path;
153 }
```

main.c:

```c
#include <stdlib.h>
#include <stdio.h>
#include "strings.h"

void input(char *delim, char *path, char *dir1, char *dir2)
{


//C:\Windows\system32+C:\User\test+C:\Windows\explorer.exe+D:\Windows\Distrib
   //C:\Windows
   //E:\WindowsXP
   //+
   //new paths:
E:\WindowsXP\system32+C:\User\test+E:\WindowsXP\explorer.exe+D:\Windows\Distr
ib
   printf("paths: ");
   fgets(path, 500, stdin);
   printf("dir1: ");
   fgets(dir1, 500, stdin);
   printf("dir2: ");
   fgets(dir2, 500, stdin);
   printf("delim: ");
   scanf("%c", delim);
}

void output(char *string)
{
   printf("new paths: %s", string);
   printf("\n");
}

char *process( char delim, char *path, char *dir1, char *dir2)
{
   char *true_delim;
   true_delim = malloc(sizeof(char));
   true_delim[0] = delim;
   char **strs = malloc(sizeof(char *) * 1000);
   int length = stok(delim, path, strs);
   int i = 1;
   char *result = malloc(sizeof(char) * 1000);
   //Формируем результат
   result = scpy(result, path_process(strs[0], dir2, dir1));
   while (i < length){
      //if (scmp(path_process(strs[i], dir2, dir1), "")){
         result = scpy(result, true_delim);
         result = scpy(result, path_process(strs[i], dir2, dir1));
      //}
      i++;
   }
   //result = scpy(result, path_process(strs[length+1], dir2, dir1));
   //result = scpy(result, strs[length+1]);
   return result;
}
```

```
50  int main()
51  {
52      char *path = malloc(sizeof(char) * 500);
53      char *dir1 = malloc(sizeof(char) * 100);
54      char *dir2 = malloc(sizeof(char) * 100);
55      char delim;
56      char *result = malloc(sizeof(char) * 1000);
57      input(&delim, path, dir1, dir2);
58      result = process(delim, path, dir1, dir2);
59      output(result);
60      free(path);
61      free(dir1);
62      free(dir2);
63      free(result);
64      return 0;
65  }
66
67
```