

Name : Mohammed Tousif  
Roll : CB.EN.P2CYS22008

## SECURE CODING LAB - IV SYSCALLS

1. Write your own version of printf named myprintfunction().

a. It should be able to accept various types of parameters such as char, int, double, etc.

b. Bonus : The function should be able to accept different parameter count. The first parameter says the count of parameters, followed by actual parameters

Source Code :-

```
#include <stdio.h>
#include <stdlib.h>
#include <stdarg.h>
#include <string.h>
char * _strrev (char *str)
{
    int i;
    int len = 0;
    char c;
    if (!str)
        return NULL;
    while(str[len] != '\0'){
        len++;
    }
    for(i = 0; i < (len/2); i++)
    {
        c = str[i];
        str [i] = str[len - i - 1];
        str[len - i - 1] = c;
    }
    return str;
}
char * _itoa(int i, char *strout, int base)
{
    char *str = strout;
```

```

int digit, sign = 0;
if (i < 0)
{
    sign = 1;
    i *= -1;
}
while(i)
{
    digit = i % base;
    *str = (digit > 9) ? ('A' + digit - 10) : '0' + digit;
    i = i / base; str++;
}
if(sign)
{
    *str++ = '-';
}
*str = '\0';
__strev(strout);
return strout;
}

void myprintf(int n_args, ...)
{
    va_list vl;
    int i = 0, j=0;
    char buff[100]={0}, tmp[20];
    char * str_arg;
    va_start(vl, n_args);
    const char *str = va_arg(vl, const char*);
    printf("Total count of parameters given : %d\n", n_args);
    while (str && str[i])
    {
        if(str[i] == '%')
        {
            i++;
            switch (str[i])
            {
                case 'c':
                {
                    buff[j] = (char)va_arg( vl, int );
                    j++;

```

```

break;
}
case 'd':
{
_itoa(va_arg( vl, int ), tmp, 10);
strcpy(&buff[j], tmp);
j += strlen(tmp);
break;
}
case 'x':
{
_itoa(va_arg( vl, int ), tmp, 16);
strcpy(&buff[j], tmp);
j += strlen(tmp);
break;
}
case 'o':
{
_itoa(va_arg( vl, int ), tmp, 8);
strcpy(&buff[j], tmp);
j += strlen(tmp);
break;
}
case 's':
{
str_arg = va_arg( vl, char* );
strcpy(&buff[j], str_arg);
j += strlen(str_arg);
break;
}
}
}
else
{
buff[j] =str[i];
j++;
}
i++;
}
fwrite(buff, j, 1, stdout);

```

```

        va_end(vl);
    }
    int main()
    {
        int n_args=5;
        myprintf(n_args, "%c %d %o %x %s\n",'s', 13, 950,
2022,"String Data");
        return 0;
    }

```

Output :-

```

tousif@TousifVM:~/sclab$ gcc 3lab1.c -o 3lab1
tousif@TousifVM:~/sclab$ ./3lab1
Total count of parameters given : 5
s 13 1666 7E6 String Data
tousif@TousifVM:~/sclab$ 

```

2. Write a program to read all txt files (that is files that ends with .txt) in the current directory and merge them all to one text file and return a file descriptor for the new file.

Source Code :-

```

#include <stdio.h>
#include <dirent.h>
#include <string.h>

int main(void)
{

    FILE *ip, *op;
    char ch;
    char *txt = ".txt";
    struct dirent *de;
    DIR *dir = opendir(".");
    if(dir == NULL)
    {

```

```

        printf("Can't open current directory.");
        return 0;
    }
    while((de = readdir(dir)) != NULL)
    {
        char *filename = de->d_name;
        char *ext = strrchr(filename, '.');
        if(!ext || ext == filename)
        {
            if(strcmp(ext, txt) == 0)
            {
                op = fopen("merged.txt", "a+");
                ip = fopen(filename, "r");
                while(1)
                {
                    ch = fgetc(ip);
                    if(ch == EOF)
                        break;
                    putc(ch, op);
                }
                fclose(ip);
                fclose(op);
            }
        }
    }
    closedir(dir);
    printf("Succesfully merged all .txt files data into merged.txt file.\n");
    return 0;
}

```

Output :-

```

tousif@TousifVM:~/sclab$ ls
3lab.c file1.txt file2.txt file3.txt
tousif@TousifVM:~/sclab$ cat file1.txt
this is line of file 1
tousif@TousifVM:~/sclab$ cat file2.txt
this is line of file 2
tousif@TousifVM:~/sclab$ cat file3.txt
this was a line 3
tousif@TousifVM:~/sclab$ █

```

```
tousif@TousifVM:~/sclab$ ls
3lab.c file1.txt file2.txt file3.txt
tousif@TousifVM:~/sclab$ gcc 3lab.c -o 3lab
tousif@TousifVM:~/sclab$ ./3lab
Successfully merged all .txt files data into merged.txt file.
tousif@TousifVM:~/sclab$ cat merged.txt
this is line of file 1
this is line of file 2
this was a line 3
tousif@TousifVM:~/sclab$
```

3. Write a program that will categorize all files in the current folder based on their file type. That is all .txt files in one folder called txt, all .bmp files in another folder called bmp etc. The argument to the program is a folder name.

Source code :-

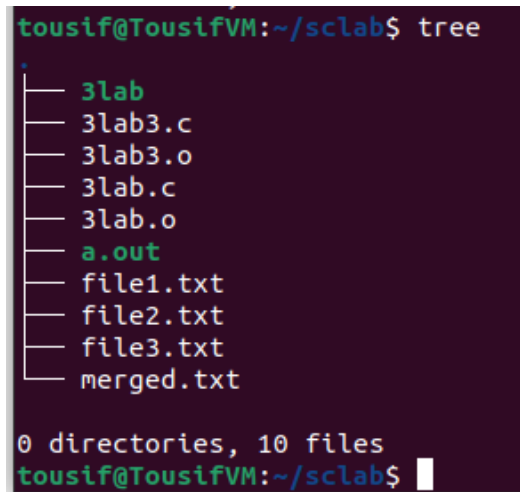
```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <dirent.h>
#include <sys/stat.h>
int main(void)
{
    DIR *crdir;
    char *p1,*p2, ext[100][100], c , filename[50], path[100];
    for(int i=0; i<100; i++)
        strcpy(ext[i], "0");
    int retn;
    struct dirent *dir;
    crdir = opendir(".");
    if (crdir)
    {
        while ((dir = readdir(crdir)) != NULL)
        {
            p1=strtok(dir->d_name,".");
            p2=strtok(NULL,".");
            if(p2!=NULL)
            {
```

```

        if(strcmp(ext[p2[0]-97], "0") == 0)
        strcmp(ext[p2[0]-97], p2);
        strcpy(filename, p1);
        strcat(filename, ".");
        strcat(filename, p2);
        mkdir(p2, 0755);
        strcpy(path, p2);
        strcat(path, "/");
        strcat(path, filename);
        FILE *fp1 = fopen(path, "w");
        FILE *fp2 = fopen(filename, "r");
        while((c = fgetc(fp2)) != EOF)
        fputc(c, fp1);
    }
}
closedir(crdir);
}
return 0;
}

```

Output :-



```

tousif@TousifVM:~/sclab$ tree
.
├── 3lab
│   ├── 3lab3.c
│   ├── 3lab3.o
│   ├── 3lab.c
│   ├── 3lab.o
│   ├── a.out
│   ├── file1.txt
│   ├── file2.txt
│   ├── file3.txt
│   └── merged.txt
└── 0 directories, 10 files
tousif@TousifVM:~/sclab$

```

```

tousif@TousifVM:~/sclab$ gcc 3lab3.c -o 3lab3
tousif@TousifVM:~/sclab$ ./3lab3
tousif@TousifVM:~/sclab$ tree
.
├── 3lab
├── 3lab3
├── 3lab3.c
├── 3lab3.o
├── 3lab.c
├── 3lab.o
├── a.out
├── c
│   ├── 3lab3.c
│   └── 3lab.c
├── file1.txt
├── file2.txt
├── file3.txt
├── merged.txt
├── o
│   ├── 3lab3.o
│   └── 3lab.o
├── out
│   └── a.out
└── txt
    ├── file1.txt
    ├── file2.txt
    ├── file3.txt
    └── merged.txt

4 directories, 20 files
tousif@TousifVM:~/sclab$

```

4. Given a directory, write a program that will find all files with the same name in the directory and its sub directories. Show their name, which folder they are in and what day they were created. Expand the program to remove all duplicate copies based on user input. That is, ask the user if each one of the files is to be kept or deleted. Based on user input, perform the appropriate action.

Source Code :-

```

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/types.h>
#include <dirent.h>
char dirnames[100][100][100]={0};
int tree(const char *path, int level)
{

```



```

int k=1;
struct dirent *dp;
DIR *dir = opendir(path);
if (!dir)
return 0;
while((dp = readdir(dir)) != NULL)
{
if (strcmp(dp->d_name, ".") != 0 && strcmp(dp->d_name, "..")
!= 0)
strcpy(dirnames[level][k++], dp->d_name);
}
closedir(dir);
return k;
}
int main()
{
static int k=0, arr[100];
struct dirent *dp;
int ch;
char dirnm[10]={0}, filnm[10]={0}, command[50]={0};
do
{
printf("\nChoose an option\n\n");
printf("1. Listing Files & Directories in tree view\n");
printf("2. Remove Duplicate files\n");
printf("3. Exit\n\n");
printf("Enter the choice: ");
scanf("%d", &ch);
if(ch == 1)
{
system("tree -D");
}
if(ch == 2)
{
system("tree -D");
DIR *dir = opendir(".");
if(!dir)
return 0;
while((dp = readdir(dir)) != NULL)
{

```

```

char *ptr1=strtok(dp->d_name, ".");
char *ptr2=strtok(NULL, ".");
if (strcmp(dp->d_name, ".") != 0 && strcmp(dp->d_name, "..")
!= 0 &&
ptr2==NULL)
{
strcpy(dirnames[k++][0], dp->d_name);
}
}
for(int i=0; i<k; i++)
arr[i] = tree(dirnames[i][0], i);
for(int i=0; i<k-1; i++)
{
for(int j=1; j<arr[i]; j++)
{
for(int x=i+1; x<k; x++)
{
for(int y=1; y<arr[x]; y++)
{
if(strcmp(dirnames[i][j], dirnames[x][y])==0)
printf("%s %s have duplicate files in %s\n", dirnames[i][0],
dirnames[x][0], dirnames[i][j]);
}
}
}
}
int flag=1;
printf("\nEnter the directory name of duplicate file: ");
scanf("%s", dirnm);
printf("\nEnter the filename: ");
scanf("%s", filnm);
for(int i=0; i<k; i++)
{
if(strcmp(dirnames[i][0], dirnm)==0)
{
flag=0;
break;
}
}
if(flag)

```

```

{
printf("Directory not found\n");
exit(0);
}
for(int i=0; i<k; i++)
{
for(int j=1; j<arr[i]; j++)
{
if(strcmp(dirnames[i][j], filenm)==0)
{
flag=0;
break;
}
}
}
if(flag)
{
printf("File not found\n");
exit(0);
}
else
{
char temp[100]={0};
strcpy(command, "rm ");
strcat(temp, dirnm);
strcat(temp, "/");
strcat(temp, filenm);
strcat(command, temp);
int stat = system(command);
if(!stat)
printf("%s from %s successfully removed..!\n", filenm, dirnm);
}
}
if(ch == 3)
{
exit(0);
}
}while(ch<3);
return 0;
}

```

## Output :-

```
tousif@TousifVM:~/sclab$ gcc 3lab4.c -o 3lab4
tousif@TousifVM:~/sclab$ ./3lab4
```

Choose an option

1. Listing Files & Directories in tree view
2. Remove Duplicate files
3. Exit

Enter the choice: 1

```
[Oct 25 11:31] .
├── [Oct  9 18:14] 3lab
├── [Oct 24 11:39] 3lab3
├── [Oct 24 11:38] 3lab3.c
├── [Oct 24 11:12] 3lab3.o
├── [Oct 25 11:31] 3lab4
├── [Oct 25 11:30] 3lab4.c
├── [Oct  9 18:14] 3lab.c
├── [Oct 24 11:12] 3lab.o
├── [Oct 24 11:14] a.out
├── [Oct 24 11:18] c
│   ├── [Oct 24 11:31] 3lab3.c
│   └── [Oct 24 11:31] 3lab.c
├── [Oct 25 10:25] dum1
├── [Oct 25 11:31] dumm
│   ├── [Oct 25 11:31] dum2
│   └── [Oct 25 11:28] dum2.c
├── [Oct  8 22:50] file1.txt
├── [Oct  9 18:02] file2.txt
├── [Oct  8 23:08] file3.txt
├── [Oct  9 18:14] merged.txt
├── [Oct 24 11:18] o
│   ├── [Oct 24 11:31] 3lab3.o
│   └── [Oct 24 11:31] 3lab.o
├── [Oct 24 11:18] out
│   └── [Oct 24 11:31] a.out
├── [Oct 24 11:18] txt
│   ├── [Oct 24 11:31] file1.txt
│   ├── [Oct 24 11:31] file2.txt
│   ├── [Oct 24 11:31] file3.txt
│   └── [Oct 24 11:31] merged.txt
```

5 directories, 25 files

```

Enter the choice: 2
[Oct 25 11:31] .
├── [Oct 9 18:14] 3lab
├── [Oct 24 11:39] 3lab3
├── [Oct 24 11:38] 3lab3.c
├── [Oct 24 11:12] 3lab3.o
├── [Oct 25 11:31] 3lab4
├── [Oct 25 11:30] 3lab4.c
├── [Oct 9 18:14] 3lab.c
├── [Oct 24 11:12] 3lab.o
├── [Oct 24 11:14] a.out
├── [Oct 24 11:18] c
│   ├── [Oct 24 11:31] 3lab3.c
│   └── [Oct 24 11:31] 3lab.c
├── [Oct 25 10:25] dum1
├── [Oct 25 11:31] dumm
│   ├── [Oct 25 11:31] dum2
│   └── [Oct 25 11:28] dum2.c
├── [Oct 8 22:50] file1.txt
├── [Oct 9 18:02] file2.txt
├── [Oct 8 23:08] file3.txt
├── [Oct 9 18:14] merged.txt
├── [Oct 24 11:18] o
│   ├── [Oct 24 11:31] 3lab3.o
│   └── [Oct 24 11:31] 3lab.o
├── [Oct 24 11:18] out
│   └── [Oct 24 11:31] a.out
└── [Oct 24 11:18] txt
    ├── [Oct 24 11:31] file1.txt
    ├── [Oct 24 11:31] file2.txt
    ├── [Oct 24 11:31] file3.txt
    └── [Oct 24 11:31] merged.txt

5 directories, 25 files

Enter the directory name of duplicate file: dumm

Enter the filename: dum2
dum2 from dumm successfully removed..!

Choose an option

1. Listing Files & Directories in tree view
2. Remove Duplicate files
3. Exit

Enter the choice: 3
tousif@TousifVM:~/sclab$

```