LAB 4

Rahul M Menon
CB.EN.P2CYS23015

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

```c
#include <stdio.h>
#include <stdarg.h>
void myprintf(const char *format, ...)
{
    va_list args;
    va_start(args, format);
    int paramCount = 0;
    while (*format)
    {
        if (*format == '%')
        {
            format++;
            switch (*format)
            {
                case 'c':
                    paramCount++;
                    putchar(va_arg(args, int));
                    break;
                case 'd':
                    paramCount++;
                    printf("%d", va_arg(args, int));
                    break;
                case 'f':
                    paramCount++;
                    printf("%f", va_arg(args, double));
                    break;
                case 's':
                    paramCount++;
                    fputs(va_arg(args, const char*), stdout);
                    break;
                default:
                    putchar(*format);
                    break;          } }
        else
            putchar(*format);
        format++;
    }
    va_end(args);
    printf("\nTotal count of parameters given: %d\n\n", paramCount);
}
int main()
{
    myprintf("%c   %d %f %s\n", 'X', 47, 333.28, "Hello WorldS");
return 0;
}
```

```
[09/03/23]seed@VM:~$ gedit myprint.c
[09/03/23]seed@VM:~$ gcc -o myprint myprint.c
[09/03/23]seed@VM:~$ ./myprint
X   47 333.280000 Hello World

Total count of parameters given: 4
```

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 newfile.

```c
#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;
}
```

```
[09/03/23]seed@VM:~$ gedit test.txt
[09/03/23]seed@VM:~$ gedit test1.txt
[09/03/23]seed@VM:~$ gedit merge.c
[09/03/23]seed@VM:~$ gcc -o merge merge.c
[09/03/23]seed@VM:~$ ./merge
Succesfully merged all .txt files data into merged.txt file.
```

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

```c
#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;
}
```

```
File  Actions  Edit  View  Help

┌──(rahul㉿kaliVM)-[~/Desktop]
└─$ ls
'C Assignmnt 2'   css_assgnmnt   css_sssh    'Lab3- SystemCalls.pdf'   new1.txt   'Secure Coding'
 categorize       css_exercise   getpid       makefile                 new2.txt    sourcelist.txt
 categorize.c     cssmakefile    Hello.java   makefile.save            new.bmp

┌──(rahul㉿kaliVM)-[~/Desktop]
└─$ ./categorize

┌──(rahul㉿kaliVM)-[~/Desktop]
└─$ ls
 bmp              categorize.c   css_sssh    'Lab3- SystemCalls.pdf'   new2.txt   'Secure Coding'
 c                css_assgnmnt   getpid       makefile                 new.bmp     sourcelist.txt
'C Assignmnt 2'   css_exercise   Hello.java   makefile.save            pdf         txt
 categorize       cssmakefile    java         new1.txt                 save

┌──(rahul㉿kaliVM)-[~/Desktop]
└─$ strace ./categorize
execve("./categorize", ["./categorize"], 0x7ffed00a9430 /* 54 vars */) = 0
brk(NULL)                               = 0x55faa723b000
mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7fe7dedc3000
access("/etc/ld.so.preload", R_OK)      = -1 ENOENT (No such file or directory)
openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3
newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=89218, ...}, AT_EMPTY_PATH) = 0
mmap(NULL, 89218, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7fe7dedad000
close(3)                                = 0
```

```
┌──(rahul㉿kaliVM)-[~/Desktop]
└─$ strace -d ./categorize
strace: ptrace_setoptions = 0×51
strace: PTRACE_GET_SYSCALL_INFO works
strace: new tcb for pid 2899, active tcbs:1
strace: [wait(0×80137f) = 2899] WIFSTOPPED,sig=SIGSTOP,EVENT_STOP (128)
strace: next_event: queued pid 2899
strace: next_event: dequeued pid 2899
strace: pid 2899 has TCB_STARTUP, initializing it
strace: [wait(0×80057f) = 2899] WIFSTOPPED,sig=SIGTRAP,EVENT_STOP (128)
strace: next_event: queued pid 2899
strace: next_event: dequeued pid 2899
strace: [wait(0×00127f) = 2899] WIFSTOPPED,sig=SIGCONT
strace: next_event: queued pid 2899
strace: next_event: dequeued pid 2899
strace: [wait(0×00857f) = 2899] WIFSTOPPED,sig=133
strace: next_event: queued pid 2899
strace: next_event: dequeued pid 2899
sstrace: [wait(0×00857f) = 2899] WIFSTOPPED,sig=133
strace: next_event: queued pid 2899
strace: next_event: dequeued pid 2899
strace: [wait(0×00857f) = 2899] WIFSTOPPED,sig=133
strace: next_event: queued pid 2899
strace: next_event: dequeued pid 2899
trace: seccomp filter disabled
strace: [wait(0×00857f) = 2899] WIFSTOPPED,sig=133
```

```
┌──(rahul㉿kaliVM)-[~/Desktop]
└─$ strace -Y ./categorize
execve("./categorize", ["./categorize"], 0×7ffc57c8d878 /* 54 vars */) = 0
brk(NULL)                               = 0×55620b577000
mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0×7fabe405d000
access("/etc/ld.so.preload", R_OK)      = -1 ENOENT (No such file or directory)
openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3
newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=89218, ...}, AT_EMPTY_PATH) = 0
mmap(NULL, 89218, PROT_READ, MAP_PRIVATE, 3, 0) = 0×7fabe4047000
close(3)                                = 0
openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libc.so.6", O_RDONLY|O_CLOEXEC) = 3
read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\220x\2\0\0\0\0\0"..., 832) = 832
pread64(3, "\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0"..., 784, 64) = 784
newfstatat(3, "", {st_mode=S_IFREG|0755, st_size=1926256, ...}, AT_EMPTY_PATH) = 0
pread64(3, "\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0"..., 784, 64) = 784
mmap(NULL, 1974096, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0×7fabe3e65000
mmap(0×7fabe3e8b000, 1396736, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0×26000) = 0×7fabe3e8b000
mmap(0×7fabe3fe0000, 344064, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0×17b000) = 0×7fabe3fe0000
mmap(0×7fabe4034000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0×1cf000) = 0×7fabe4034000
mmap(0×7fabe403a000, 53072, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0×7fabe403a000
close(3)                                = 0
mmap(NULL, 12288, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0×7fabe3e62000
arch_prctl(ARCH_SET_FS, 0×7fabe3e62740) = 0
set_tid_address(0×7fabe3e62a10)         = 4756
set_robust_list(0×7fabe3e62a20, 24)     = 0
rseq(0×7fabe3e63060, 0×20, 0, 0×53053053) = 0
mprotect(0×7fabe4034000, 16384, PROT_READ) = 0
```

```
┌──(rahul㉿kaliVM)-[~/Desktop]
└─$ strace -c ./categorize
% time     seconds  usecs/call     calls    errors syscall
------ ----------- ----------- --------- --------- ----------------
  0.00    0.000000           0        13           read
  0.00    0.000000           0         5           write
  0.00    0.000000           0         3           close
  0.00    0.000000           0         1           lseek
  0.00    0.000000           0         8           mmap
  0.00    0.000000           0         3           mprotect
  0.00    0.000000           0         1           munmap
  0.00    0.000000           0         3           brk
  0.00    0.000000           0         2           pread64
  0.00    0.000000           0         1         1 access
  0.00    0.000000           0         1           execve
  0.00    0.000000           0         8         8 mkdir
  0.00    0.000000           0         1           arch_prctl
  0.00    0.000000           0         2           getdents64
  0.00    0.000000           0         1           set_tid_address
  0.00    0.000000           0        19           openat
  0.00    0.000000           0        16           newfstatat
  0.00    0.000000           0         1           set_robust_list
  0.00    0.000000           0         1           prlimit64
  0.00    0.000000           0         1           getrandom
```

```
┌──(rahul㉿kaliVM)-[~/Desktop]
└─$ strace -t ./categorize
16:07:28 execve("./categorize", ["./categorize"], 0×7ffde7a37048 /* 54 vars */) = 0
16:07:28 brk(NULL)                      = 0×5627c4d7e000
16:07:28 mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0×7f9b89ec4000
16:07:28 access("/etc/ld.so.preload", R_OK) = -1 ENOENT (No such file or directory)
16:07:28 openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3
16:07:28 newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=89218, ...}, AT_EMPTY_PATH) = 0
16:07:28 mmap(NULL, 89218, PROT_READ, MAP_PRIVATE, 3, 0) = 0×7f9b89eae000
16:07:28 close(3)                       = 0
16:07:28 openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libc.so.6", O_RDONLY|O_CLOEXEC) = 3
16:07:28 read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\220x\2\0\0\0\0\0"..., 832) = 832
16:07:28 pread64(3, "\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0"..., 784, 64) = 784
16:07:28 newfstatat(3, "", {st_mode=S_IFREG|0755, st_size=1926256, ...}, AT_EMPTY_PATH) = 0
16:07:28 pread64(3, "\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0"..., 784, 64) = 784
16:07:28 mmap(NULL, 1974096, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0×7f9b89ccc000
16:07:28 mmap(0×7f9b89cf2000, 1396736, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0×26000) = 0×7f9b89cf2000
16:07:28 mmap(0×7f9b89e47000, 344064, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0×17b000) = 0×7f9b89e47000
16:07:28 mmap(0×7f9b89e9b000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0×1cf000) = 0×7f9b89e9b000
16:07:28 mmap(0×7f9b89ea1000, 53072, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0×7f9b89ea1000
16:07:28 close(3)                       = 0
16:07:28 mmap(NULL, 12288, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0×7f9b89cc9000
16:07:28 arch_prctl(ARCH_SET_FS, 0×7f9b89cc9740) = 0
16:07:28 set_tid_address(0×7f9b89cc9a10) = 7060
16:07:28 set_robust_list(0×7f9b89cc9a20, 24) = 0
16:07:28 rseq(0×7f9b89cca060, 0×20, 0, 0×53053053) = 0
16:07:28 mprotect(0×7f9b89e9b000, 16384, PROT_READ) = 0
```

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

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <dirent.h>
#include <sys/stat.h>
#include <time.h>

#define MAX 1000

void find_files(char *basePath, char *filename, int *count, char paths[MAX][MAX]);
void remove_duplicates(char paths[MAX][MAX], int count);

int main()
{
    char filename[MAX];
    char basePath[MAX];
    char paths[MAX][MAX];
    int count = 0;

    printf("Enter the directory path: ");
    scanf("%s", basePath);

    printf("Enter the filename to search for: ");
    scanf("%s", filename);

    find_files(basePath, filename, &count, paths);

    if (count == 0)
        printf("No files found with the name '%s'\\n", filename);
    else
        remove_duplicates(paths, count);

    return 0;
}

void find_files(char *basePath, char *filename, int *count, char paths[MAX][MAX])
{
    char path[MAX];
    struct dirent *dp;
    struct stat buffer;
    DIR *dir = opendir(basePath);

    if (!dir)
        return;

    while ((dp = readdir(dir)) != NULL)
    {
        if (strcmp(dp->d_name, ".") != 0 && strcmp(dp->d_name, "..") != 0)
        {
            strcpy(path, basePath);
            strcat(path, "/");
            strcat(path, dp->d_name);

            if (stat(path, &buffer) == 0 && S_ISDIR(buffer.st_mode))
                find_files(path, filename, count, paths);
            else if (strcmp(dp->d_name, filename) == 0)
            {
                printf("File found: %s\\n", path);
                printf("Folder: %s\\n", basePath);
                printf("Creation time: %s\\n", ctime(&buffer.st_ctime));
                strcpy(paths[*count], path);
                (*count)++;
            }
        }
    }

```

```c
    closedir(dir);
}

void remove_duplicates(char paths[MAX][MAX], int count)
{
    char ch;
    int i;

    for (i = 0; i < count; i++)
    {
        printf("\\nDo you want to keep or delete file '%s'? (k/d): ", paths[i]);
        scanf(" %c", &ch);

        if (ch == 'd' || ch == 'D')
        {
            if (remove(paths[i]) == 0)
                printf("File '%s' deleted successfully.\\n", paths[i]);
            else
                printf("Unable to delete file '%s'.\\n", paths[i]);
        }
        else
            printf("File '%s' kept.\\n", paths[i]);
    }
}
```

```
[09/03/23]seed@VM:~$ cd Desktop
[09/03/23]seed@VM:~/Desktop$ gedit hello.txt
[09/03/23]seed@VM:~/Desktop$ cd
[09/03/23]seed@VM:~$ ./duplicates
Enter the directory path: /home/seed/Desktop
Enter the filename to search for: hello.txt
File found: /home/seed/Desktop/hello.txt\nFolder: /home/seed/Desktop\nCreation time: Sun Sep  3 13:30:22 2023
\n\nDo you want to keep or delete file '/home/seed/Desktop/hello.txt'? (k/d): d
```