Advanced Unix Programming
Lab 3

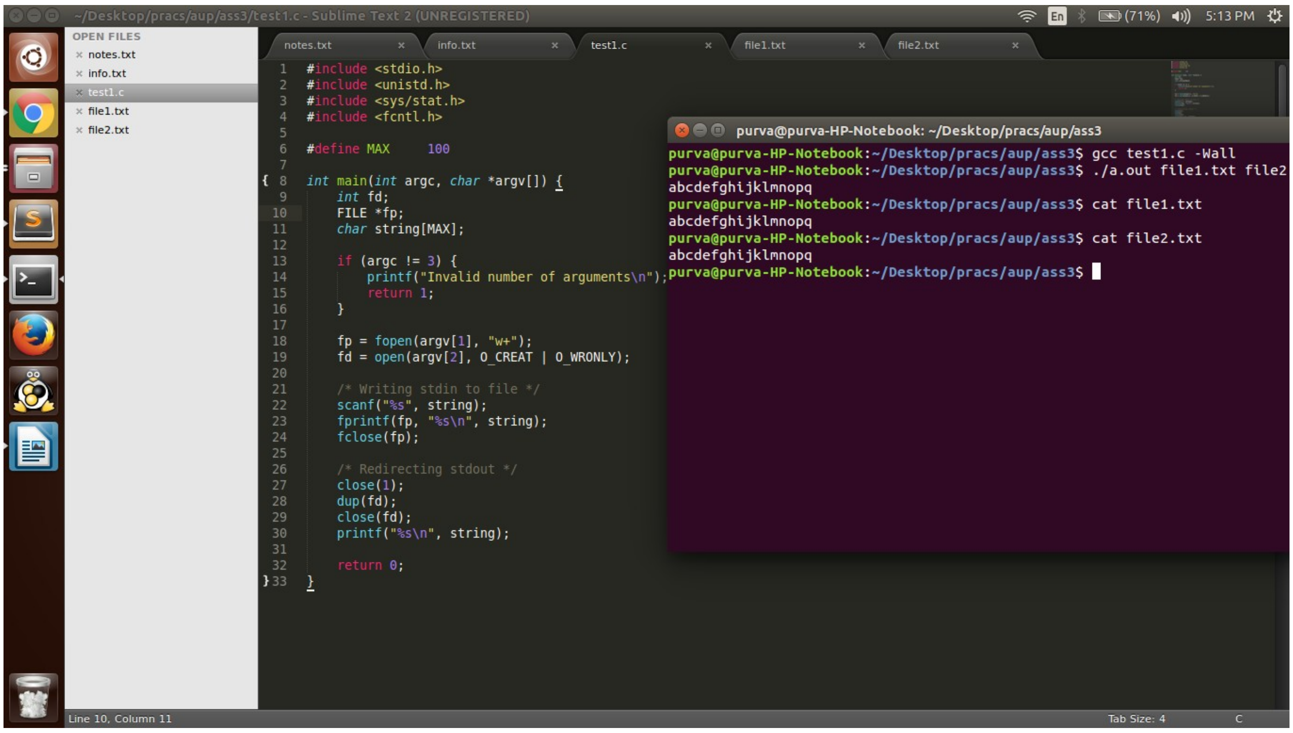
Purva Tendulkar : 111403049

Q1. Using dup function redirect stdin to file1 and stdout to file2. Read a line using scanf and write the same using printf. Verify the contents of both files.

Code :

```
#include <stdio.h>
#include <unistd.h>
#include <sys/stat.h>
#include <fcntl.h>

#define MAX 100

int main(int argc, char *argv[]) {
	int fd;
	FILE *fp;
	char string[MAX];

	if (argc != 3) {
		printf("Invalid number of arguments\n");
		return 1;
	}

	fp = fopen(argv[1], "w+");
	fd = open(argv[2], O_CREAT | O_WRONLY);

	/* Writing stdin to file */
	scanf("%s", string);
	fprintf(fp, "%s\n", string);
	fclose(fp);

	/* Redirecting stdout */
	close(1);
	dup(fd);
	close(fd);
	printf("%s\n", string);

	return 0;
}
```

Input and Output Screenshots :



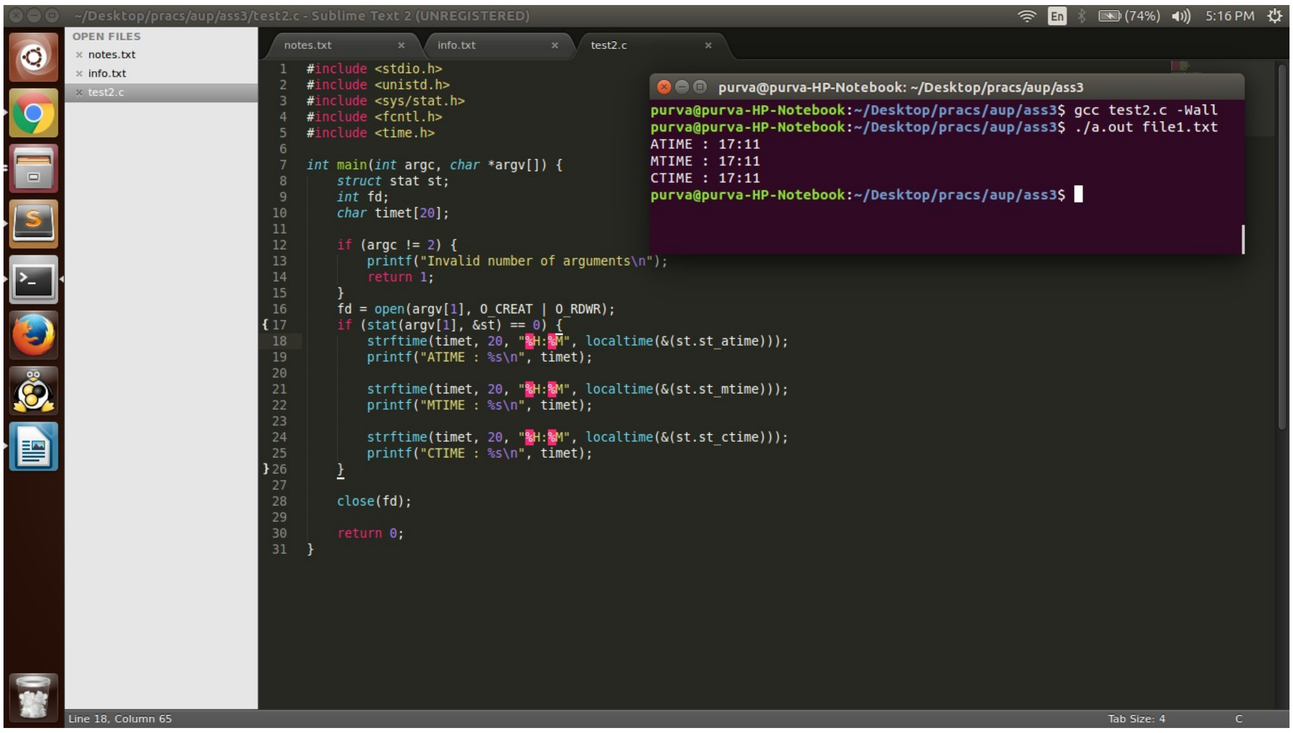
```c
#include <stdio.h>
#include <unistd.h>
#include <sys/stat.h>
#include <fcntl.h>

#define MAX     100

int main(int argc, char *argv[]) {
    int fd;
    FILE *fp;
    char string[MAX];

    if (argc != 3) {
        printf("Invalid number of arguments\n");
        return 1;
    }

    fp = fopen(argv[1], "w+");
    fd = open(argv[2], O_CREAT | O_WRONLY);

    /* Writing stdin to file */
    scanf("%s", string);
    fprintf(fp, "%s\n", string);
    fclose(fp);

    /* Redirecting stdout */
    close(1);
    dup(fd);
    close(fd);
    printf("%s\n", string);

    return 0;
}
```

```
purva@purva-HP-Notebook:~/Desktop/pracs/aup/ass3$ gcc test1.c -Wall
purva@purva-HP-Notebook:~/Desktop/pracs/aup/ass3$ ./a.out file1.txt file2
abcdefghijklmnopq
purva@purva-HP-Notebook:~/Desktop/pracs/aup/ass3$ cat file1.txt
abcdefghijklmnopq
purva@purva-HP-Notebook:~/Desktop/pracs/aup/ass3$ cat file2.txt
abcdefghijklmnopq
purva@purva-HP-Notebook:~/Desktop/pracs/aup/ass3$
```

Q2. Does calling stat function change any of the time values? Verify with a program.
Code :

```c
#include <stdio.h>
#include <unistd.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <time.h>

int main(int argc, char *argv[]) {
	struct stat st;
	int fd;
	char timet[20];

	if (argc != 2) {
		printf("Invalid number of arguments\n");
		return 1;
	}
	fd = open(argv[1], O_CREAT | O_RDWR);
	if (stat(argv[1], &st) == 0) {
		strftime(timet, 20, "%H:%M", localtime(&(st.st_atime)));
		printf("ATIME : %s\n", timet);

		strftime(timet, 20, "%H:%M", localtime(&(st.st_mtime)));
		printf("MTIME : %s\n", timet);

		strftime(timet, 20, "%H:%M", localtime(&(st.st_ctime)));
		printf("CTIME : %s\n", timet);
	}

	close(fd);
	return 0;
}
```

Input and Output Screenshots :



```
#include <stdio.h>
#include <unistd.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <time.h>

int main(int argc, char *argv[]) {
    struct stat st;
    int fd;
    char timet[20];

    if (argc != 2) {
        printf("Invalid number of arguments\n");
        return 1;
    }
    fd = open(argv[1], O_CREAT | O_RDWR);
    if (stat(argv[1], &st) == 0) {
        strftime(timet, 20, "%H:%M", localtime(&(st.st_atime)));
        printf("ATIME : %s\n", timet);

        strftime(timet, 20, "%H:%M", localtime(&(st.st_mtime)));
        printf("MTIME : %s\n", timet);

        strftime(timet, 20, "%H:%M", localtime(&(st.st_ctime)));
        printf("CTIME : %s\n", timet);
    }

    close(fd);

    return 0;
}
```

```
purva@purva-HP-Notebook:~/Desktop/pracs/aup/ass3$ gcc test2.c -Wall
purva@purva-HP-Notebook:~/Desktop/pracs/aup/ass3$ ./a.out file1.txt
ATIME : 17:11
MTIME : 17:11
CTIME : 17:11
purva@purva-HP-Notebook:~/Desktop/pracs/aup/ass3$
```

~/Desktop/pracs/aup/ass3/test2.c - Sublime Text 2 (UNREGISTERED)
(74%)
5:16 PM
OPEN FILES
notes.txt
info.txt
test2.c
purva@purva-HP-Notebook: ~/Desktop/pracs/aup/ass3
Line 18, Column 65
Tab Size: 4
C

Explanation :

1. File "file1.txt" was created at time 17:11 and was not accessed after that. Current time is 17:16. Thus invoking stat has not changed any of the time values.
2. Change time is updated by renaming the file.
3. Access time is updated when reading the contents of a file
4. Modify time is updated the file (opening for modification is not enough to change modify time)

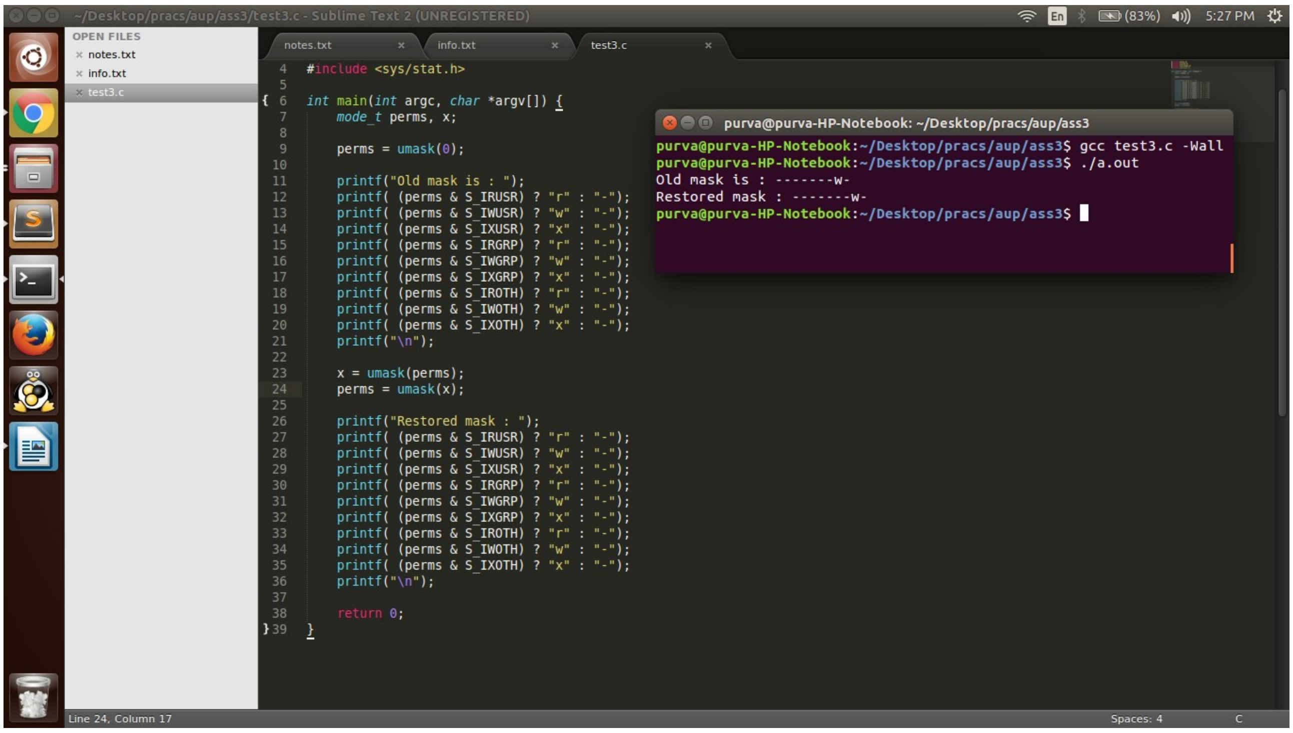
Q3. umask() always sets the process umask and, at the same time, returns a copy of the old umask. How can we obtain a copy of the current process umask while leaving it unchanged? Write a program to demonstrate.

Code :

```
#include <stdio.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/stat.h>

int main(int argc, char *argv[]) {
	mode_t perms, x;

	perms = umask(0);

	printf("Old mask is : ");
	printf( (perms & S_IRUSR) ? "r" : "-");
	printf( (perms & S_IWUSR) ? "w" : "-");
	printf( (perms & S_IXUSR) ? "x" : "-");
```

```
        printf( (perms & S_IRGRP) ? "r" : "-");
        printf( (perms & S_IWGRP) ? "w" : "-");
        printf( (perms & S_IXGRP) ? "x" : "-");
        printf( (perms & S_IROTH) ? "r" : "-");
        printf( (perms & S_IWOTH) ? "w" : "-");
        printf( (perms & S_IXOTH) ? "x" : "-");
        printf("\n");

        x = umask(perms);
        perms = umask(x);

        printf("Restored mask : ");
        printf( (perms & S_IRUSR) ? "r" : "-");
        printf( (perms & S_IWUSR) ? "w" : "-");
        printf( (perms & S_IXUSR) ? "x" : "-");
        printf( (perms & S_IRGRP) ? "r" : "-");
        printf( (perms & S_IWGRP) ? "w" : "-");
        printf( (perms & S_IXGRP) ? "x" : "-");
        printf( (perms & S_IROTH) ? "r" : "-");
        printf( (perms & S_IWOTH) ? "w" : "-");
        printf( (perms & S_IXOTH) ? "x" : "-");
        printf("\n");

        return 0;
}
```

Input and Output Screenshots :


~/Desktop/pracs/aup/ass3/test3.c - Sublime Text 2 (UNREGISTERED)
OPEN FILES
notes.txt
info.txt
test3.c
Line 24, Column 17
Spaces: 4
C

```
#include <sys/stat.h>

int main(int argc, char *argv[]) {
    mode_t perms, x;

    perms = umask(0);

    printf("Old mask is : ");
    printf( (perms & S_IRUSR) ? "r" : "-");
    printf( (perms & S_IWUSR) ? "w" : "-");
    printf( (perms & S_IXUSR) ? "x" : "-");
    printf( (perms & S_IRGRP) ? "r" : "-");
    printf( (perms & S_IWGRP) ? "w" : "-");
    printf( (perms & S_IXGRP) ? "x" : "-");
    printf( (perms & S_IROTH) ? "r" : "-");
    printf( (perms & S_IWOTH) ? "w" : "-");
    printf( (perms & S_IXOTH) ? "x" : "-");
    printf("\n");

    x = umask(perms);
    perms = umask(x);

    printf("Restored mask : ");
    printf( (perms & S_IRUSR) ? "r" : "-");
    printf( (perms & S_IWUSR) ? "w" : "-");
    printf( (perms & S_IXUSR) ? "x" : "-");
    printf( (perms & S_IRGRP) ? "r" : "-");
    printf( (perms & S_IWGRP) ? "w" : "-");
    printf( (perms & S_IXGRP) ? "x" : "-");
    printf( (perms & S_IROTH) ? "r" : "-");
    printf( (perms & S_IWOTH) ? "w" : "-");
    printf( (perms & S_IXOTH) ? "x" : "-");
    printf("\n");

    return 0;
}
```

```
purva@purva-HP-Notebook:~/Desktop/pracs/aup/ass3$ gcc test3.c -Wall
purva@purva-HP-Notebook:~/Desktop/pracs/aup/ass3$ ./a.out
Old mask is : -------w-
Restored mask : -------w-
purva@purva-HP-Notebook:~/Desktop/pracs/aup/ass3$
```

Explanation :

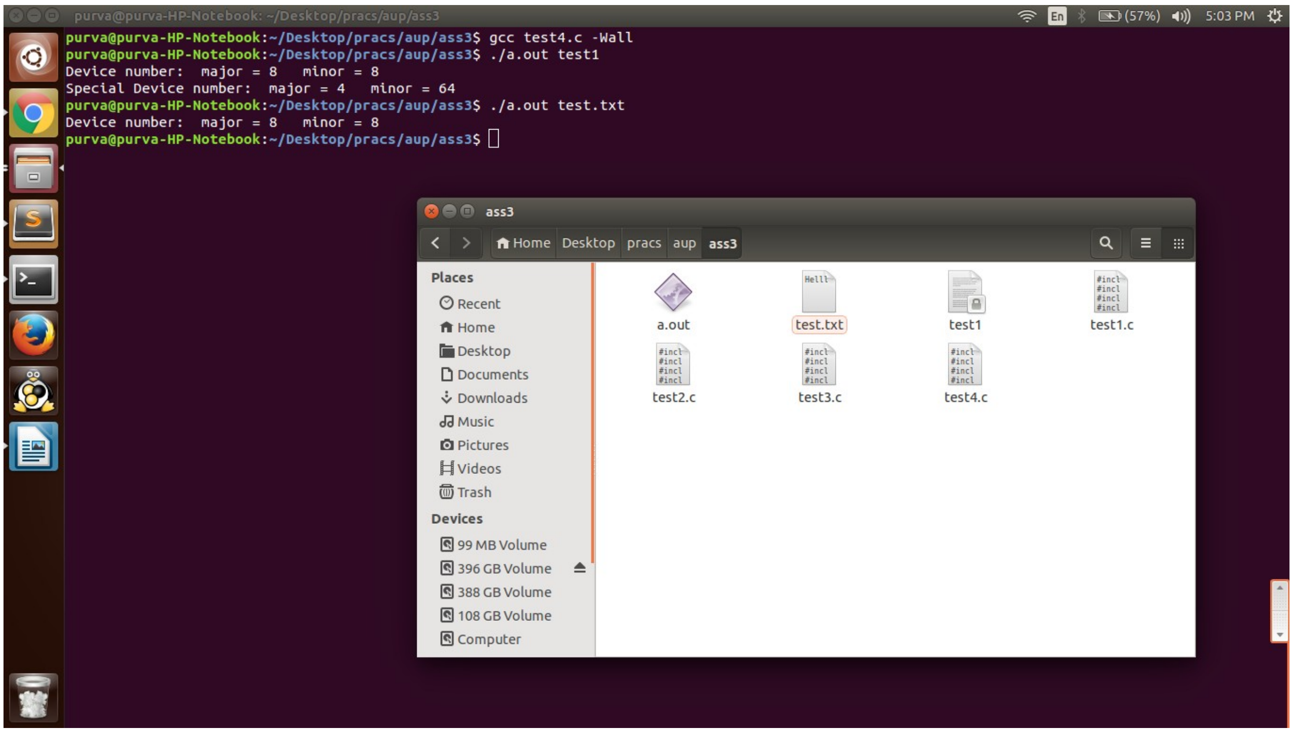
1. We can obtain umask and reset it to obtained value.
2. For this, it is necessary to call umask twice as is demonstrated in the program.

Q4. Display the device number for the filename input as command line argument. If it is a character or block special file, then display its major and minor numbers.

Code :

```c
#include <stdio.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/stat.h>

int main(int argc, char *argv[]) {
	if (argc != 2) {
		printf("Invalid number of arguments\n");
		return 1;
	}

	struct stat st;
	stat(argv[1], &st);

	// display device number
	printf("Device number:  major = %ld   minor = %ld\n", (long)major(st.st_dev),
(long)minor(st.st_dev));

	// check if character/block file and display resp major and minor numbers
	if (S_ISCHR(st.st_mode) || S_ISBLK(st.st_mode))
		printf("Special Device number:  major = %ld   minor = %ld\n",
			(long) major(st.st_rdev), (long) minor(st.st_rdev));

	return 0;
}
```

Input & Output Screenshots :



```
purva@purva-HP-Notebook:~/Desktop/pracs/aup/ass3$ gcc test4.c -Wall
purva@purva-HP-Notebook:~/Desktop/pracs/aup/ass3$ ./a.out test1
Device number:  major = 8   minor = 8
Special Device number:  major = 4   minor = 64
purva@purva-HP-Notebook:~/Desktop/pracs/aup/ass3$ ./a.out test.txt
Device number:  major = 8   minor = 8
purva@purva-HP-Notebook:~/Desktop/pracs/aup/ass3$ 
```