



Sir Syed CASE Institute of Technology

Muhammad Talha Ramzan

2330-0141

Bs Ai

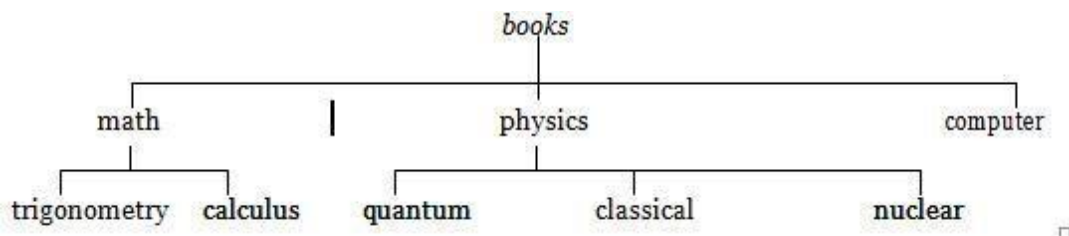
Submitted to:

Aymen Fatima

Lab # 02

Task no 01

Create the following directory structure:



```
ubuntu@ubuntu: ~/Desktop/book
ubuntu@ubuntu:~/Desktop$ mkdir book
ubuntu@ubuntu:~/Desktop$ cd book
ubuntu@ubuntu:~/Desktop/book$ mkdir math physic computer
ubuntu@ubuntu:~/Desktop/book$ cd math
ubuntu@ubuntu:~/Desktop/book/math$ mk dir trigonometry calculus
mk: command not found
ubuntu@ubuntu:~/Desktop/book/math$ mkdir trigonometry calculus
ubuntu@ubuntu:~/Desktop/book/math$ cd..
cd.: command not found
ubuntu@ubuntu:~/Desktop/book/math$ cd ..
ubuntu@ubuntu:~/Desktop/book$ cd physic
ubuntu@ubuntu:~/Desktop/book/physic$ mkdir quantum classical nuclear
ubuntu@ubuntu:~/Desktop/book/physic$ cd ..
Command 'tree' not found, but can be installed with:
sudo apt install tree
ubuntu@ubuntu:~/Desktop/book$ sudo apt install tree
Reading package lists... Done
Building dependency tree... Done
```

```

ubuntu@ubuntu:~/Desktop/book$ cd ..
ubuntu@ubuntu:~/Desktop$ tree book
book
├── computer
├── math
│   ├── calculus
│   └── trigonometry
└── physic
    ├── classical
    ├── nuclear
    └── quantum

```

Command I use to perform the task :

Open the terminal in Ubuntu and run the following commands. First, create the main directory named books using the command `mkdir books`. Then, inside books, create three subdirectories: math, physics, and computer using `mkdir math ,physics ,computer`. Next, create trigonometry and calculus inside the math directory with `mkdir trigonometry books/math/calculus`. Similarly, create quantum, classical, and nuclear inside the physics directory using `mkdir quantum classical nuclear`. Finally, to verify the directory structure, use the command `tree books`. If the tree command is not installed, install it first by running `sudo apt install tree` , then re-run `tree books` to check the structure.

Task no 02

Write a program that prints odd numbers from 1 to 10 and print their sum, Compile and run it using gcc.

```

ubuntu@ubuntu:~$ gcc oddnumber_print.c -o oddnumber
ubuntu@ubuntu:~$ ./oddnumber
odd number from 1 to 10 :
13579
sum of odd number : 25
ubuntu@ubuntu:~$ S

```

Commands I use to perform the task:

First, we ran the `nano filename.c` command, which opened the compiler. Then, we wrote the code and saved it by pressing `Ctrl + O` and then `Enter`. After that, we ran the command `gcc filename -o executable_file`. The next command was `./executable_file`, which displayed the final output.

Task no 03

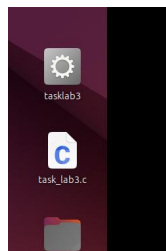
Student Grade Calculation System

```

ubuntu@ubuntu:~/Desktop$
ubuntu@ubuntu:~/Desktop$ gcc task_lab3.c -o tasklab3
ubuntu@ubuntu:~/Desktop$ ./tasklab3
enter the number of the student:2
enter marks for 2 subjects :
subject 1: 87
subject 2: 59

average marks: 73.00
grade: C
ubuntu@ubuntu:~/Desktop$ S

```



```
ubuntu@ubuntu: ~/Desktop
GNU nano 7.2 task_lab3.c *
#include<stdio.h>
float calculateAverage(int marks[], int numsubject){
int sum=0;
for(int i=0; i< numsubject; i++){
sum += marks[i];
}
return (float)sum/numsubject;
}
char getgrade(float avg){
if (avg >= 90) return 'A';
else if (avg >= 80) return 'B';
else if (avg >= 70) return 'C';
else if (avg >= 60) return 'D';
else return 'F';
}
int main(){
int numsubject;
```

```
ubuntu@ubuntu: ~/Desktop
GNU nano 7.2 task_lab3.c *
else if (avg >= 60) return 'D';
else return 'F';
}
int main(){
int numsubject;
printf("enter the number of the student:");
scanf("%d",&numsubject);
int marks[numsubject];
printf("enter marks for %d subjects : \n", numsubject);
for(int i=0; i < numsubject; i++){
printf("subject %d: ", i + 1);
scanf("%d", &marks[i]);
}
float avg = calculateAverage(marks,numsubject);
char grade = getgrade(avg);
printf("\naverage marks: %.2f\n",avg);
printf("grade: %c\n",grade);
return 0;
}
^G Help ^O Write Out ^W Where Is ^K Cut ^T Execute ^C Location
^X Exit ^R Read File ^\ Replace ^U Paste ^J Justify ^_ Go To Line
```

Commands I use to perform the task:

First, we ran the nano filename.c command, which opened the compiler. Then, we wrote the code and saved it by pressing Ctrl + O and then Enter. After that, we ran the command gcc filename -o executable_file. The next command was ./executable_file, which displayed the final output.