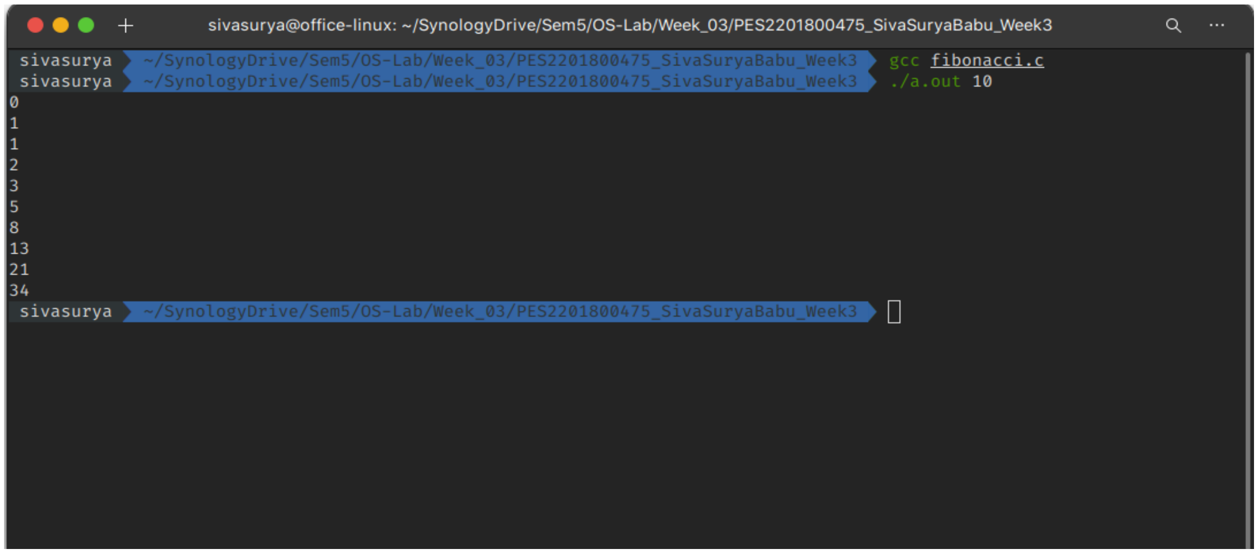
# OS Lab Week 3

Siva Surya Babu
PES2201800475

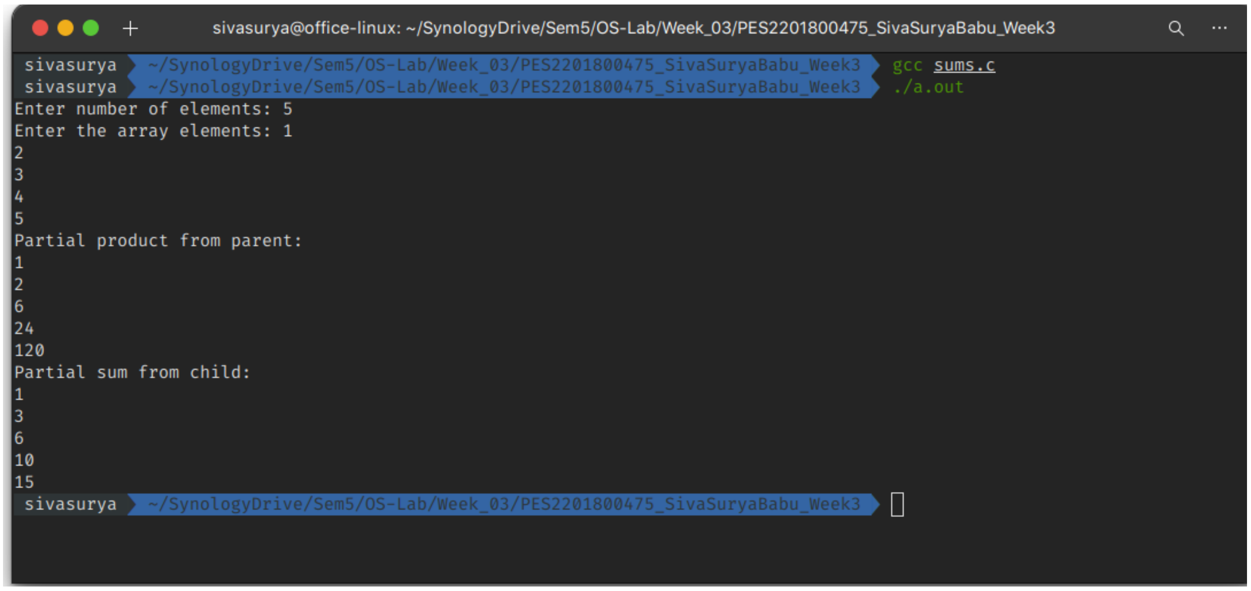
## Submission 1:

Question 1



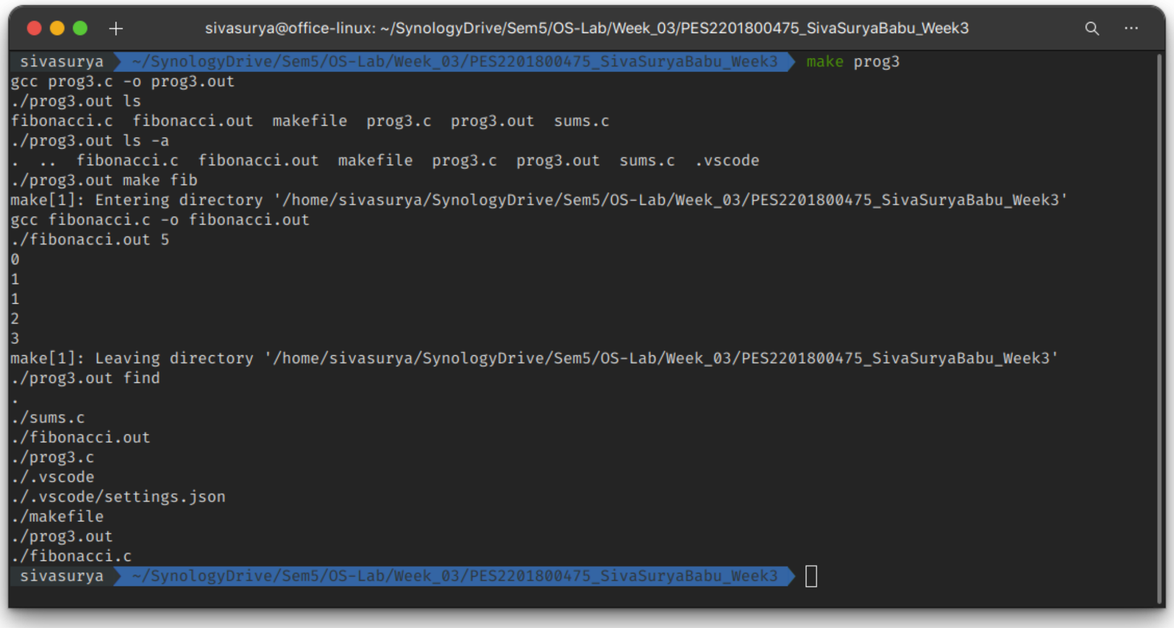
```
sivasurya@office-linux: ~/SynologyDrive/Sem5/OS-Lab/Week_03/PES2201800475_SivaSuryaBabu_Week3
 sivasurya  ~/SynologyDrive/Sem5/OS-Lab/Week_03/PES2201800475_SivaSuryaBabu_Week3  gcc fibonacci.c
 sivasurya  ~/SynologyDrive/Sem5/OS-Lab/Week_03/PES2201800475_SivaSuryaBabu_Week3  ./a.out 10
0
1
1
2
3
5
8
13
21
34
 sivasurya  ~/SynologyDrive/Sem5/OS-Lab/Week_03/PES2201800475_SivaSuryaBabu_Week3 
```

Question 2



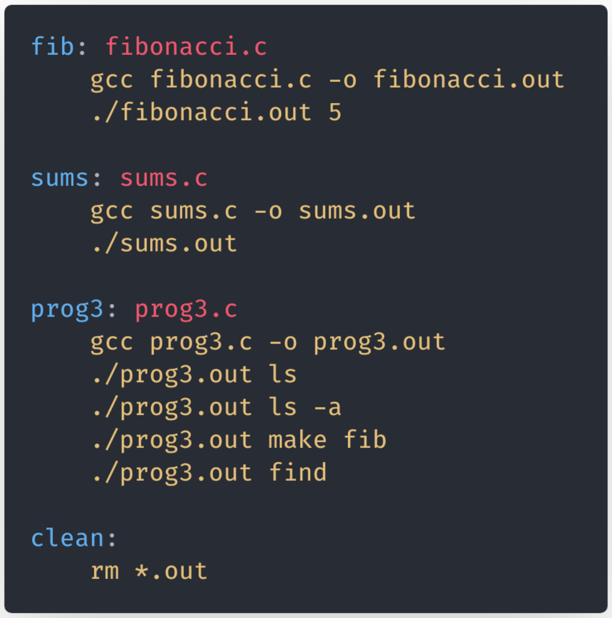
```
sivasurya@office-linux: ~/SynologyDrive/Sem5/OS-Lab/Week_03/PES2201800475_SivaSuryaBabu_Week3
 sivasurya  ~/SynologyDrive/Sem5/OS-Lab/Week_03/PES2201800475_SivaSuryaBabu_Week3  gcc sums.c
 sivasurya  ~/SynologyDrive/Sem5/OS-Lab/Week_03/PES2201800475_SivaSuryaBabu_Week3  ./a.out
Enter number of elements: 5
Enter the array elements: 1
2
3
4
5
Partial product from parent:
1
2
6
24
120
Partial sum from child:
1
3
6
10
15
 sivasurya  ~/SynologyDrive/Sem5/OS-Lab/Week_03/PES2201800475_SivaSuryaBabu_Week3 
```

Question 3



```
sivasurya@office-linux: ~/SynologyDrive/Sem5/OS-Lab/Week_03/PES2201800475_SivaSuryaBabu_Week3
 sivasurya  ~/SynologyDrive/Sem5/OS-Lab/Week_03/PES2201800475_SivaSuryaBabu_Week3  make prog3
gcc prog3.c -o prog3.out
./prog3.out ls
fibonacci.c  fibonacci.out  makefile  prog3.c  prog3.out  sums.c
./prog3.out ls -a
.  ..  fibonacci.c  fibonacci.out  makefile  prog3.c  prog3.out  sums.c  .vscode
./prog3.out make fib
make[1]: Entering directory '/home/sivasurya/SynologyDrive/Sem5/OS-Lab/Week_03/PES2201800475_SivaSuryaBabu_Week3'
gcc fibonacci.c -o fibonacci.out
./fibonacci.out 5
0
1
1
2
3
make[1]: Leaving directory '/home/sivasurya/SynologyDrive/Sem5/OS-Lab/Week_03/PES2201800475_SivaSuryaBabu_Week3'
./prog3.out find
.
./sums.c
./fibonacci.out
./prog3.c
./.vscode
./.vscode/settings.json
./makefile
./prog3.out
./fibonacci.c
 sivasurya  ~/SynologyDrive/Sem5/OS-Lab/Week_03/PES2201800475_SivaSuryaBabu_Week3 
```

All output is coming at once because I have used a makefile



```
fib: fibonacci.c
    gcc fibonacci.c -o fibonacci.out
    ./fibonacci.out 5

sums: sums.c
    gcc sums.c -o sums.out
    ./sums.out

prog3: prog3.c
    gcc prog3.c -o prog3.out
    ./prog3.out ls
    ./prog3.out ls -a
    ./prog3.out make fib
    ./prog3.out find

clean:
    rm *.out
```

## Submission 2:

1. **What is the role of the init process on UNIX and Linux systems in regard to process termination?**

   **Ans.** Init is the ancestor of all other processes (can be direct or indirect ancestor) and it gets all the abandoned processes. Init then invokes wait() on a regular basis to release the abandoned processes' PID and the process entry table

2. **What is a subreaper process?**

   **Ans.** A subreaper fulfills the role of init(1) for its descendant processes. Upon termination of a process that is orphaned and marked as having a subreaper, the nearest still living ancestor subreaper will receive a SIGCHLD signal and be able to wait(2) on the process to discover its termination status.

3. **What causes a defunct process on the Linux system and how can you avoid it ?**

   **Ans.** The process that has either completed its task or has been corrupted or killed, but its child processes are still running or these parent processes are monitoring its child process. There are a few ways to avoid this and they are using the wait() system call or ignoring the SIGCHLD signal or by using a signal handler

4. **How can you identify zombie processes on the Linux system?**

   **Ans.** Zombie processes can be identified using the ps command. ps aux lists all the processes. If a process has the letter Z under the STAT column or has <defunct> next to its name, it is a zombie process

5. **What does child process inherit from its parent?**

   **Ans.** The child process is an identical copy of the parent process and it inherits most the attributes a few examples are - signal handling settings, file descriptors, memory segments