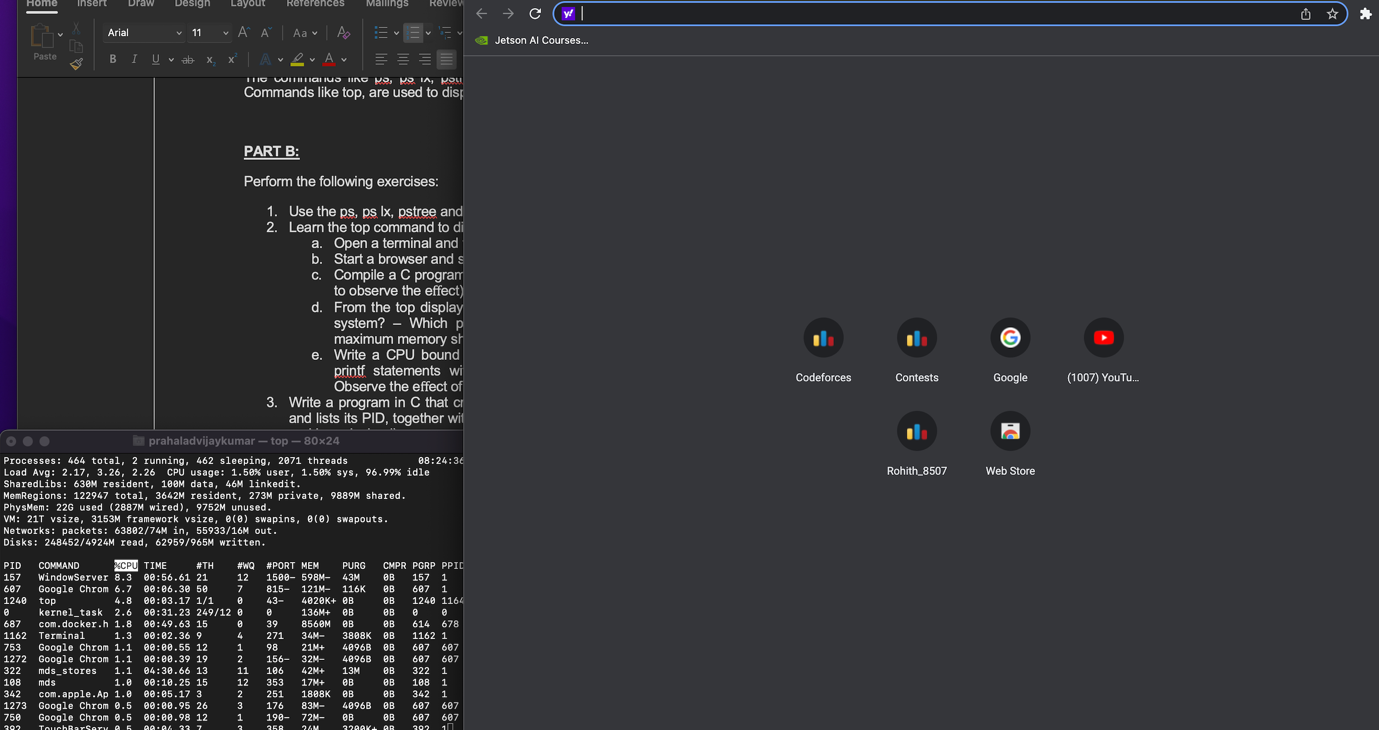
**Lab -2-Part B**

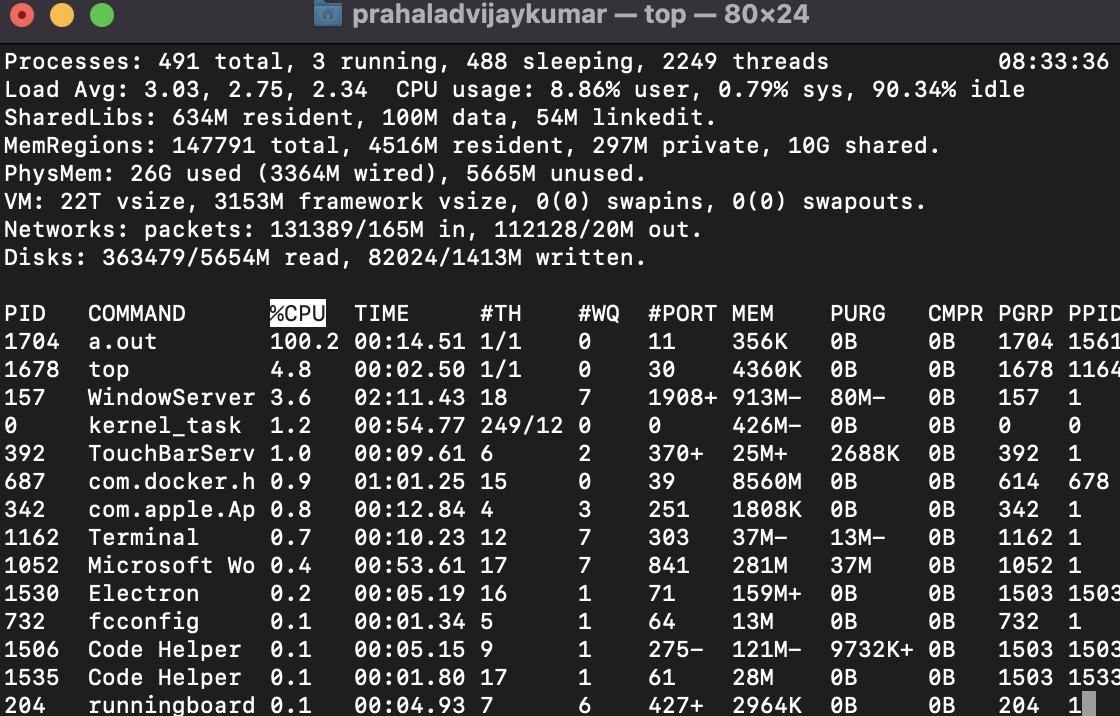
Prahalad Vijaykumar

2003128

2)(a) top command acts like an activity monitor providing info about all the running process. Displays a summary of list of process or threads running

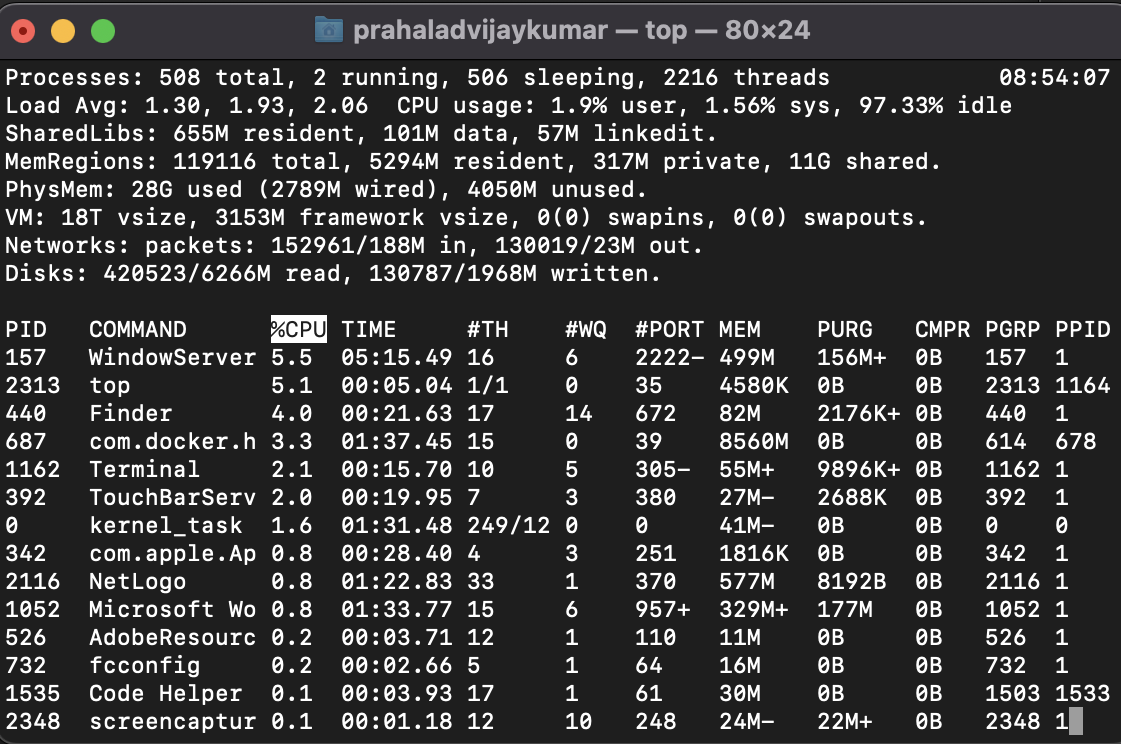
(b) on opening the chrome browser for a specific point of time chrome browser was displayed at the top of the list when the top command was running



(c) It become the most used process with cpu taking upto more than 1 of the process for execution (reaching more than 100%).

The object executable file becomes the most cpu consuming

(d)

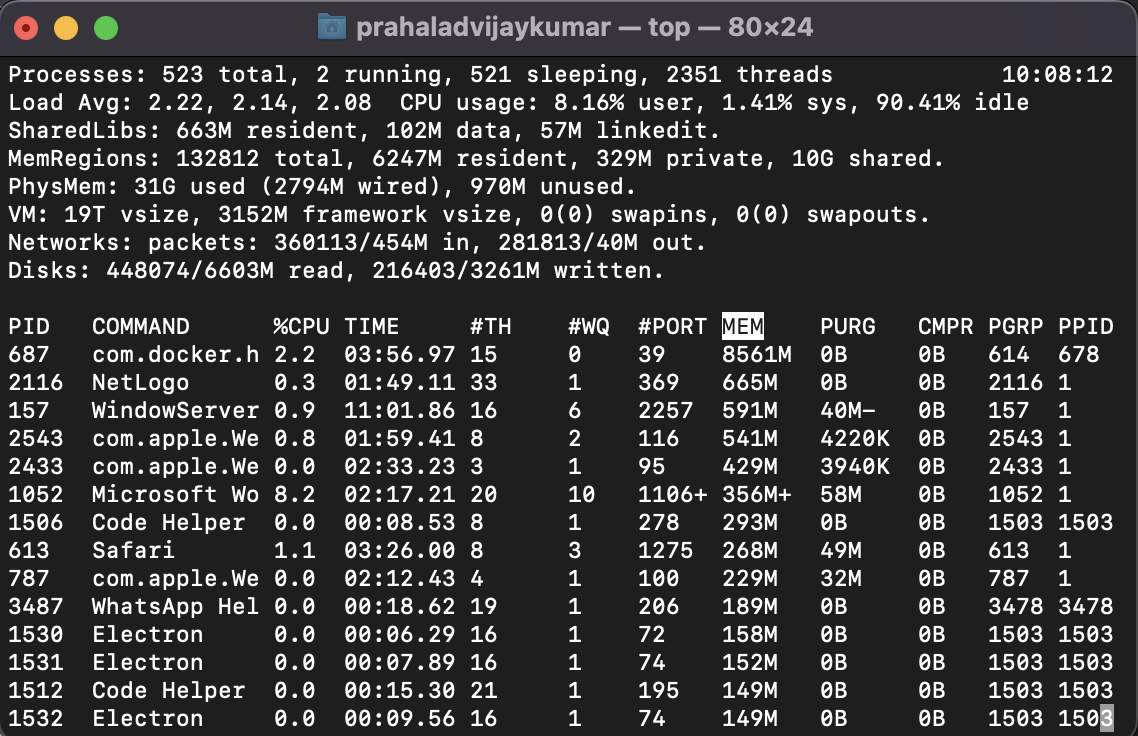


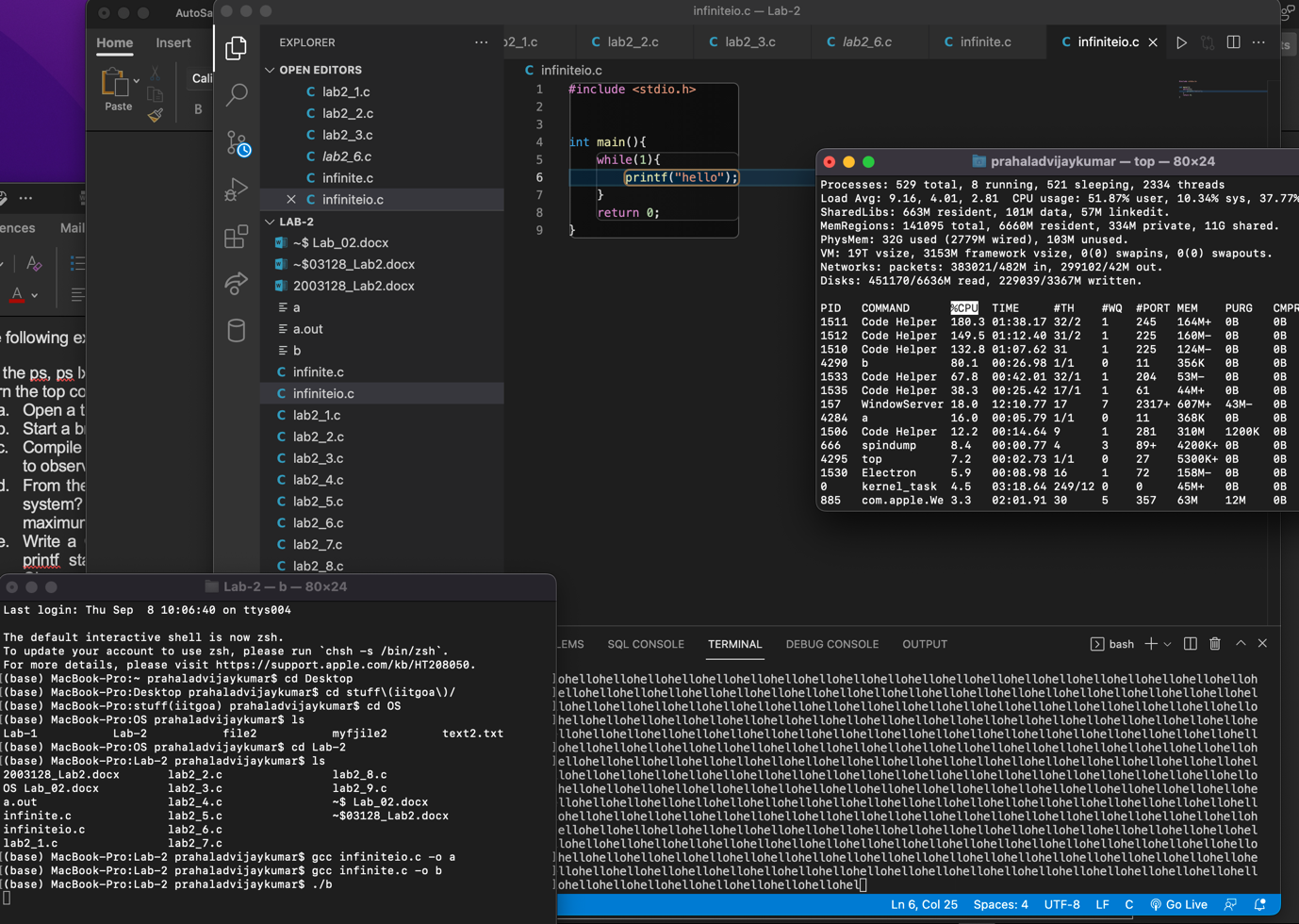
32 GB ram

Unused or free memory : 4 GB approx.

Window Server takes up more CPU consumption. This is a process in mac which manages all windows which are open

Docker takes up most memory usage



e)

b is the object file of CPU bound code

a is the object file of I/O bound code

the CPU bound takes up more CPU than the code with I/O bound. Since I/O bound is slower than CPU bound due to the output section .more percentage of CPU is spend towards the CPU bound

3)

#include<stdio.h>

#include<sys/types.h>

#include<sys/wait.h>

#include<stdlib.h>

#include <unistd.h>

void killchild(){

pid\_t pid = getpid();

printf("Child Process:%d\n",pid);

printf("Child is Terminated\n");

exit(256);

//state of exit ranges from 0-255 and -1 if there is an interrupt

}

int main(){

int pid=fork();

int status=0;

if(pid>0){

pid = wait(&status);

int num=WEXITSTATUS(status);

printf("State of Process Termination:%d\n",num);

}

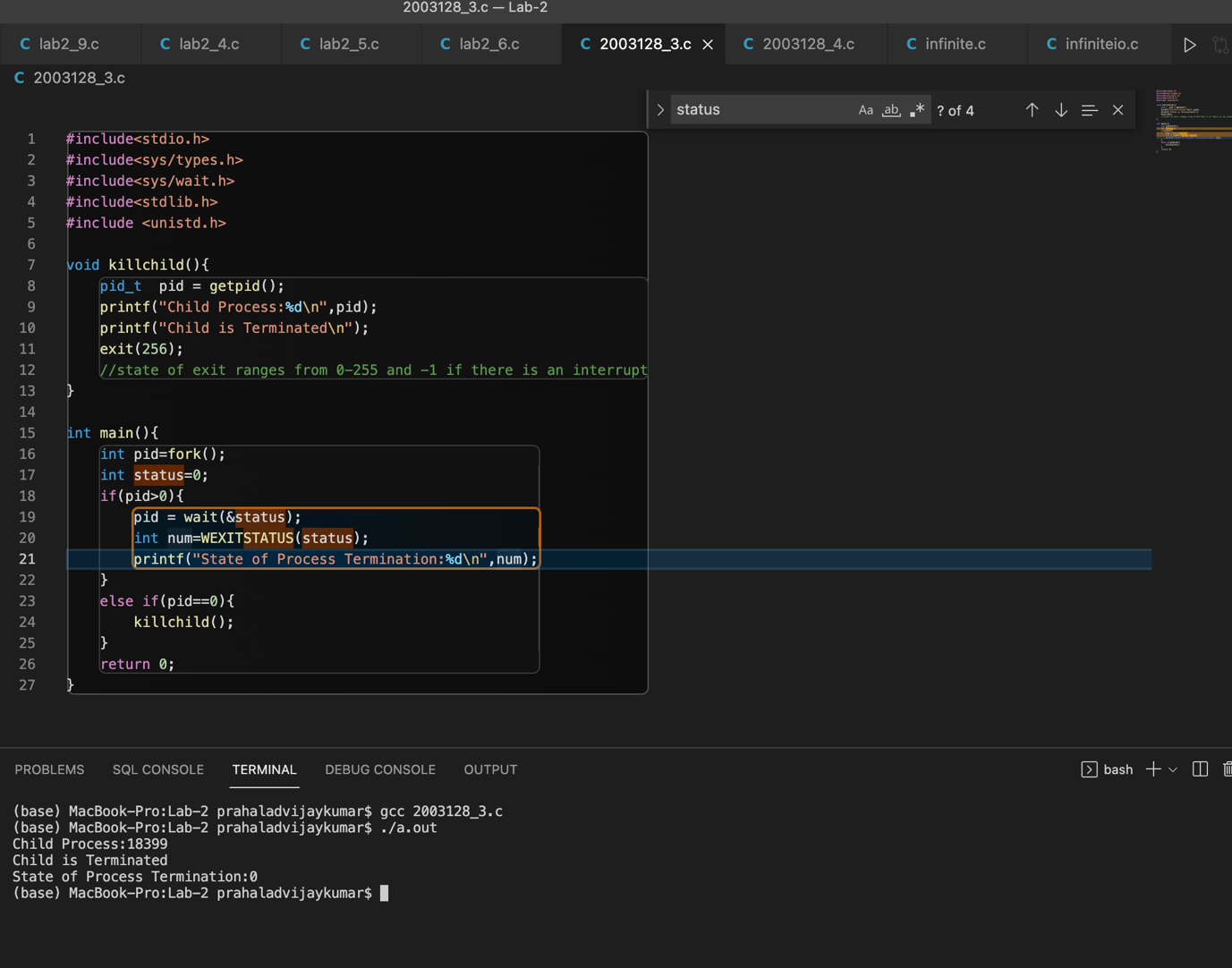
else if(pid==0){

killchild();

}

return 0;

}



4)

#include<stdio.h>

#include<sys/types.h>

#include<sys/wait.h>

#include<stdlib.h>

#include <unistd.h>

int main(){

int x=10;

pid\_t processid;

int t=2;

while(t--){

processid=getpid();

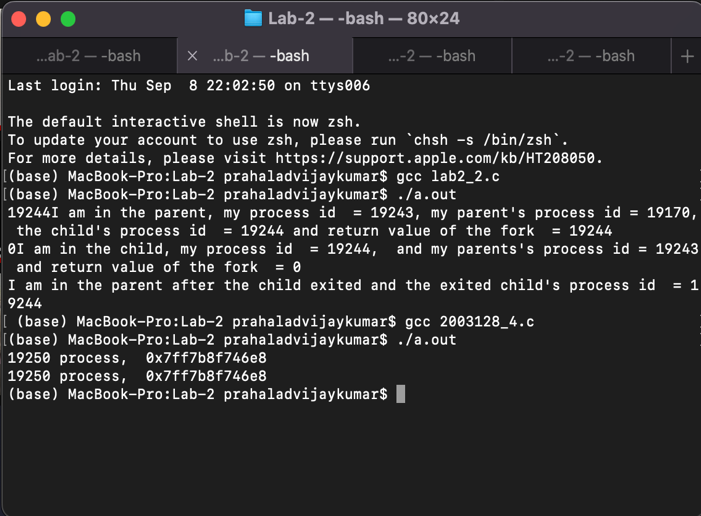
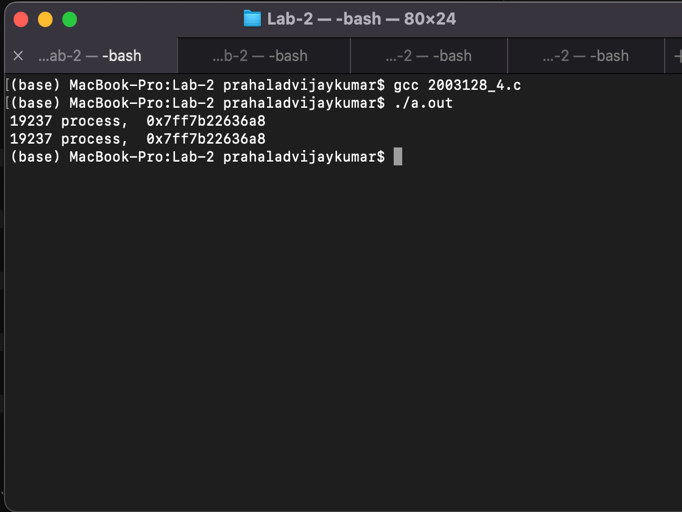
printf("%d process, %p \n",processid,&x);

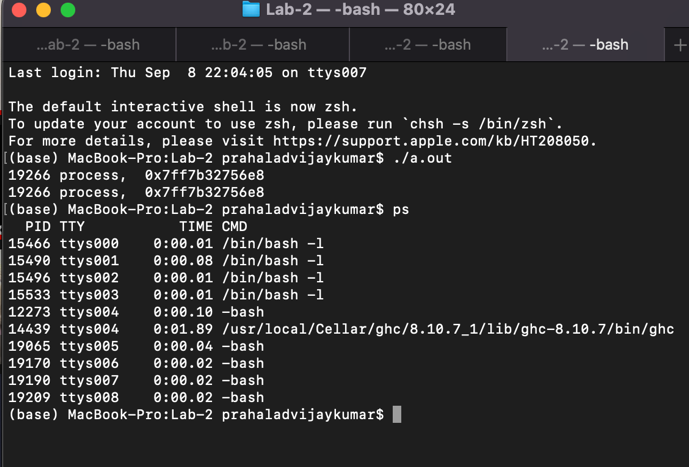
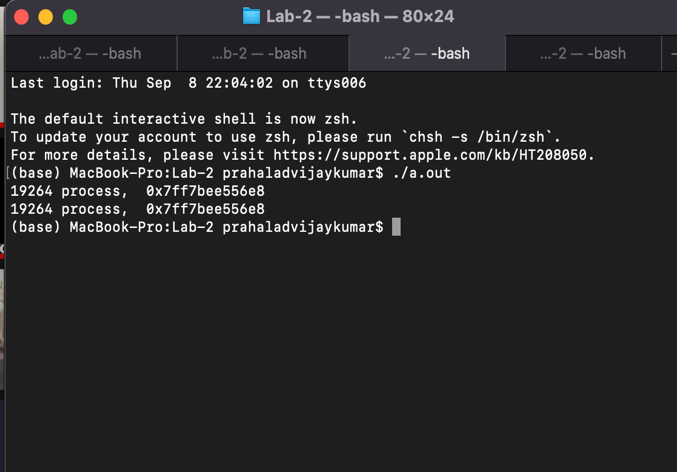
}

return 0;

}

Opened 4 terminals and each process executes the same code and found out that the address creates a different copy on running the same executable file . Each process creates its own copy.





Now if the process have parent child relation

#include<stdio.h>

#include<sys/types.h>

#include<sys/wait.h>

#include<stdlib.h>

#include <unistd.h>

int main(){

fork();

fork();

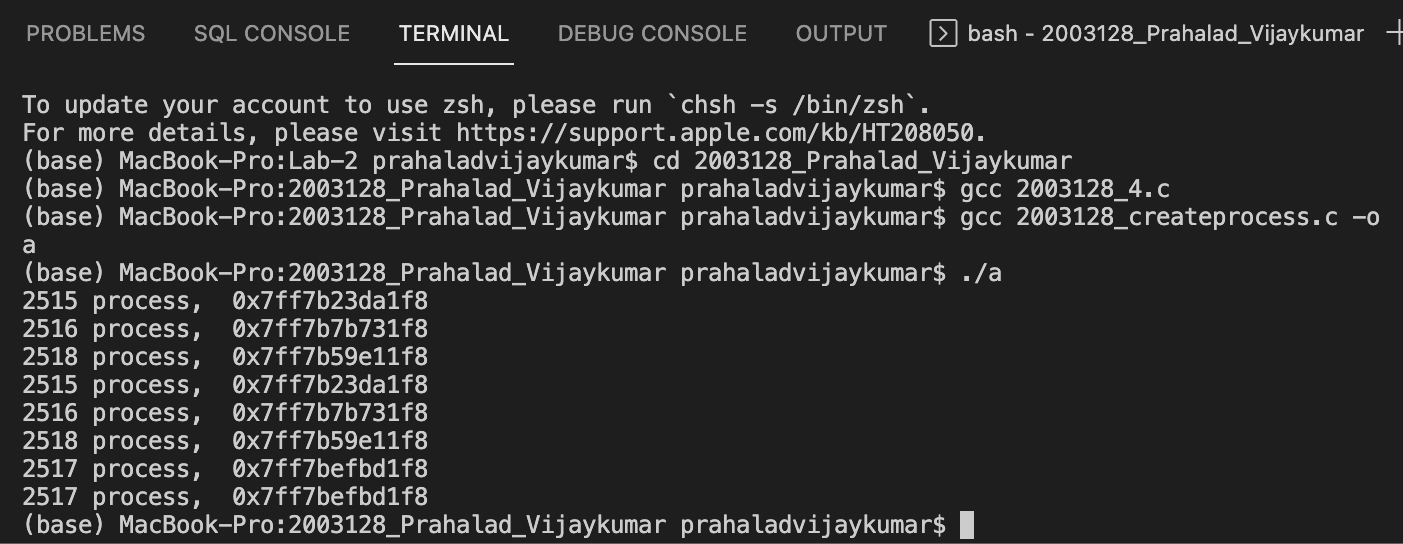
execlp("./a.out","./a.out",NULL);

}

a.out is the object file of 2003128\_4.c

and running the parent and child process 2003128\_createprocess.c

Here create four process with parent child and executes the same file.obj is the object file mentioned above . on executing them I received different memory location created for parent and child and not accessing the same memory location.



Accesses separate memory block everytime a new process is created!!!!