PROGRAM1:FORK()

#include <stdio.h>

//#include <sys/types.h>

#include <unistd.h>

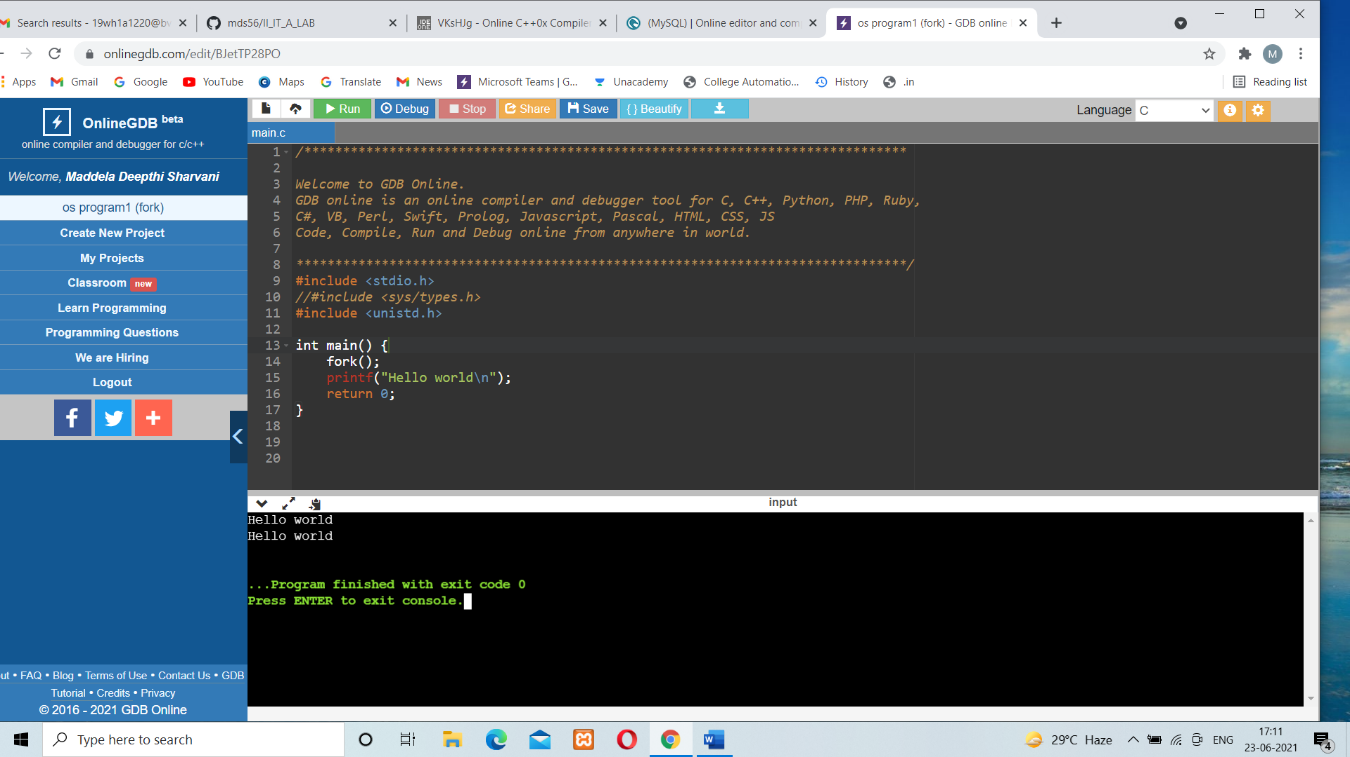
int main() {

fork();

printf("Hello world\n");

return 0;

}



PROGRAM 2:PARENT ID()

#include <stdio.h>

//#include <sys/types.h>

#include <unistd.h>

int main() {

int pid = fork();

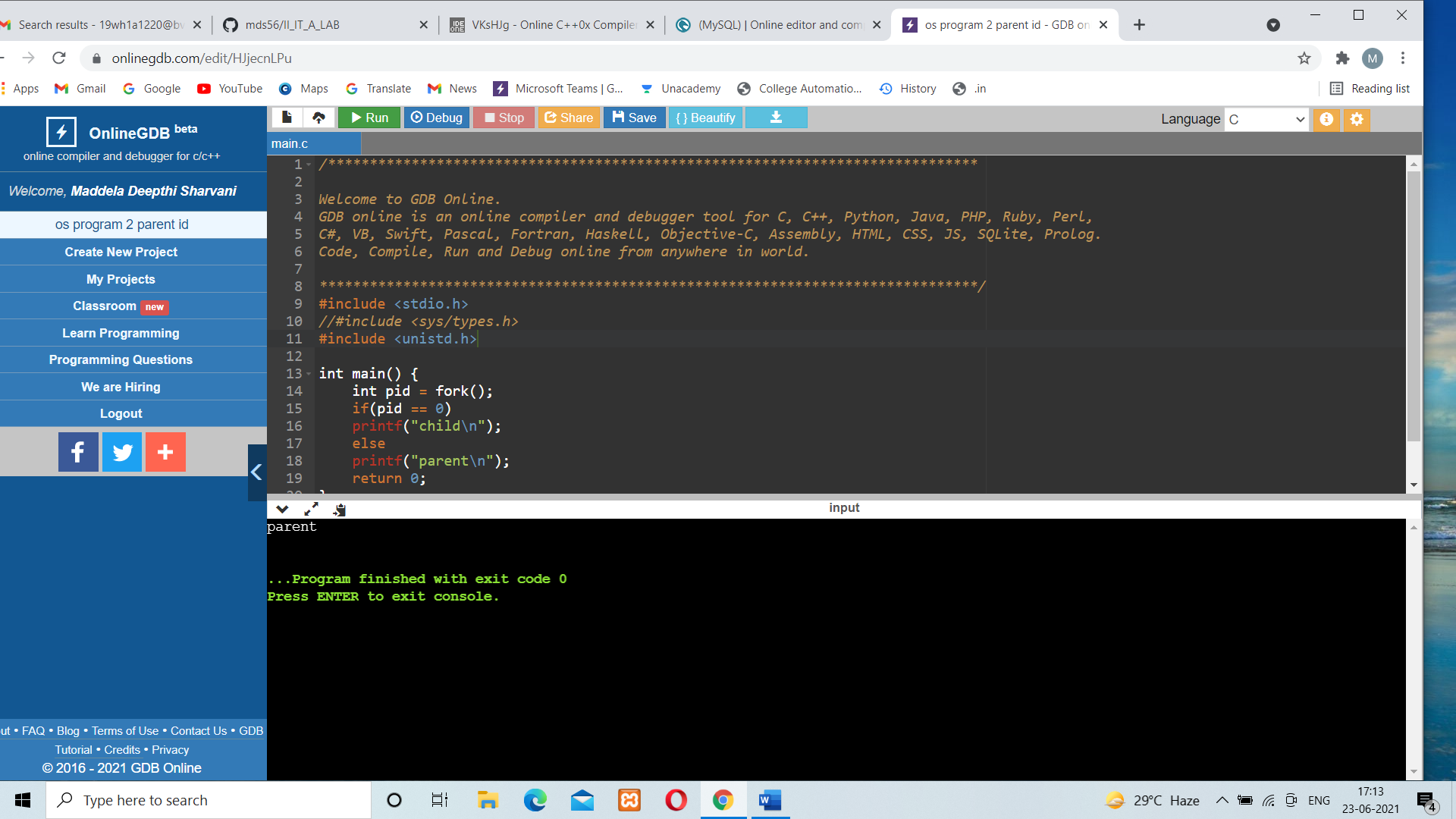
if(pid == 0)

printf("child\n");

else

printf("parent\n");

return 0;}



PROGRAM 3:MULTIPLE FORK()

#include <stdio.h>

#include <sys/types.h>

#include <unistd.h>

int main()

{

fork();

fork();

fork();

printf("Hello\n");

return 0;

}

PROGRAM-4:PID VALUE()

#include <stdio.h>

#include <unistd.h>

int main()

{

int p\_id,p\_pid;

p\_id=getpid(); /\*process id\*/

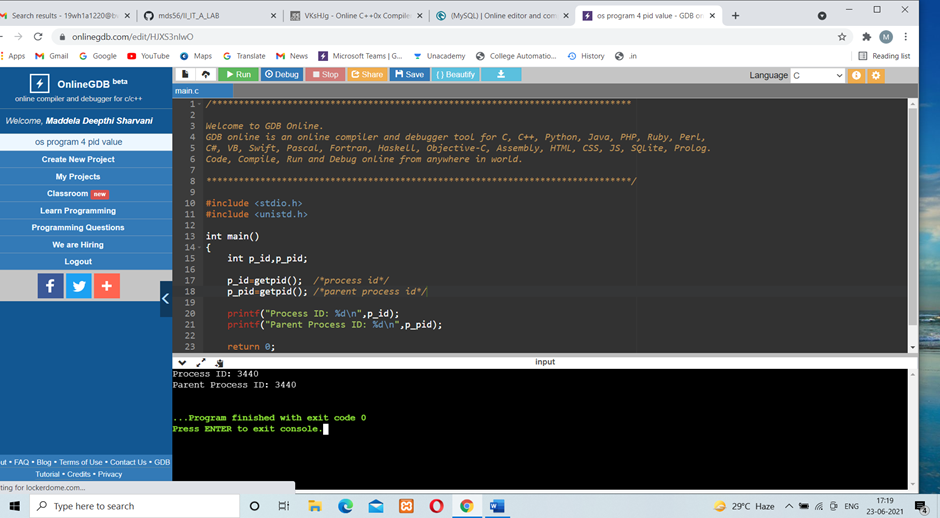
p\_pid=getpid(); /\*parent process id\*/

printf("Process ID: %d\n",p\_id);

printf("Parent Process ID: %d\n",p\_pid);

return 0;

}



PROGRAM 5:FCFS

#include<stdio.h>

int main(){

int n,bt[20],wt[20],tat[20],averagewt=0,i,j;

float averagetat=0;

printf("Enter number of processes:");

scanf("%d",&n);

printf("\nEnter Burst Time of\n");

for(i=0;i<n;i++)

{

printf("P[%d]:",i+1);

scanf("%d",&bt[i]);

}

wt[0]=0;

for(i=1;i<n;i++)

{

wt[i]=0;

for(j=0;j<i;j++)

wt[i]+=bt[j]; }//ct=tat//

printf("\nProcess\t\tBurst Time\tWaiting Time\tTurnaround Time");

for(i=0;i<n;i++)

{

tat[i]=bt[i]+wt[i];

averagewt+=wt[i];

averagetat+=tat[i];

printf("\nP[%d]\t\t%d\t\t%d\t\t%d",i+1,bt[i],wt[i],tat[i]);

}

averagewt/=i;

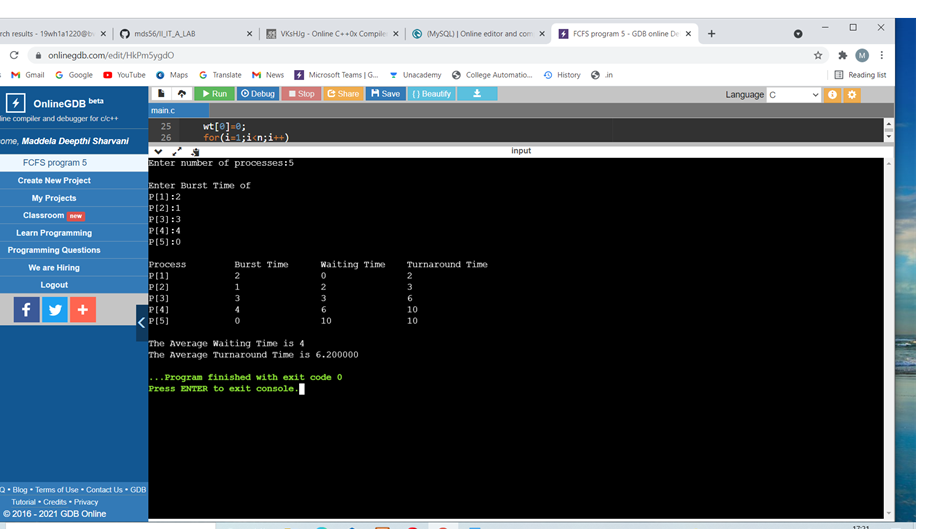
averagetat/=i;

printf("\n\nThe Average Waiting Time is %d",averagewt);

printf("\nThe Average Turnaround Time is %f",averagetat);

return 0;

}



PROGRAM 6:SJF

#include<stdio.h>

void main()

{

int bt[20],p[20],wt[20],tat[20],i,j,n,total=0,pos,temp;

float avg\_wt,avg\_tat;

printf("Enter number of process:");

scanf("%d",&n);

printf("\nEnter Burst Time:\n");

for(i=0;i<n;i++)

{

printf("p%d:",i+1);

scanf("%d",&bt[i]);

p[i]=i+1; //contains process number

} //sorting burst time in ascending order using selection sort

for(i=0;i<n;i++)

{ pos=i;

for(j=i+1;j<n;j++)

{

if(bt[j]<bt[pos])

pos=j;

}

temp=bt[i];

bt[i]=bt[pos];

bt[pos]=temp;

temp=p[i];

p[i]=p[pos];

p[pos]=temp;

}

wt[0]=0; //waiting time for first process will be zero

//calculate waiting time

for(i=1;i<n;i++)

{

wt[i]=0;

for(j=0;j<i;j++)

wt[i]+=bt[j];

total+=wt[i];

}

avg\_wt=(float)total/n; //average waiting time

total=0;

printf("\nProcess\t Burst Time \tWaiting Time\tTurnaround Time");

for(i=0;i<n;i++)

{

tat[i]=bt[i]+wt[i]; //calculate turnaround time

total+=tat[i];

printf("\np%d\t\t %d\t\t %d\t\t\t%d",p[i],bt[i],wt[i],tat[i]);

}

avg\_tat=(float)total/n; //average turnaround time

printf("\n\nAverage Waiting Time=%.2f",avg\_wt);

printf("\nAverage Turnaround Time=%.2f\n",avg\_tat);

}

