1.Identify the system calls to copy the content of one file to another and illustrate the same using a C program.

Program :-

#include<stdio.h>

#include<unistd.h>

#include<sys/types.h>

int main()

{

pid\_t p;

printf("before fork\n");

p=fork();

if(p==0)

{

printf("I am child having id %d\n",getpid());

printf("My parent's id is %d\n",getppid());

}

else{

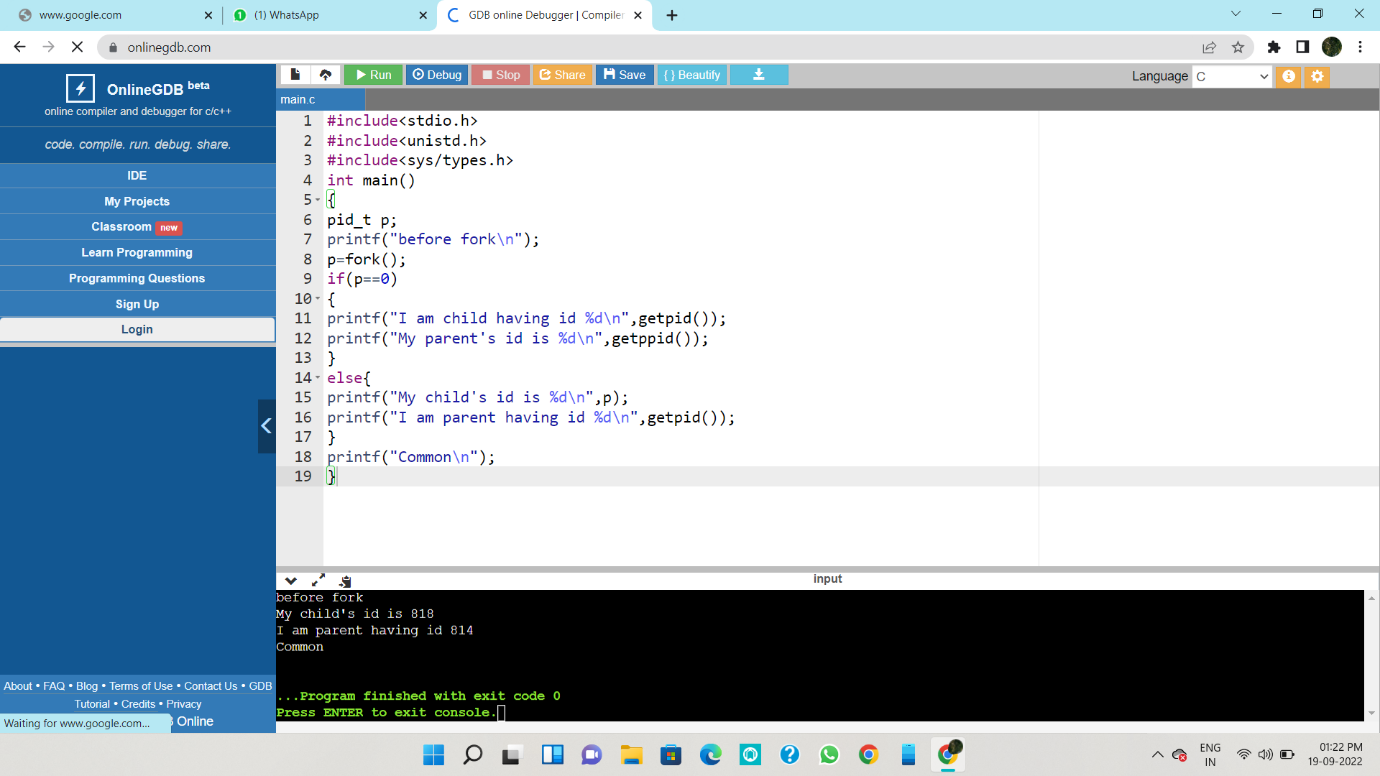
printf("My child's id is %d\n",p);

printf("I am parent having id %d\n",getpid());

}

printf("Common\n");

}

Out put :-

2 Identify the system calls to copy the content of one file to another and illustrate the same using a C program.

Program :-

#include <stdio.h>

#include <stdlib.h>

int main()

{

FILE \*sourceFile;

FILE \*destFile;

char sourcePath[100];

char destPath[100];

char ch;

printf("Enter source file path: ");

scanf("%s", sourcePath);

printf("Enter destination file path: ");

scanf("%s", destPath);

sourceFile = fopen(sourcePath, "r");

destFile = fopen(destPath, "w");

if (sourceFile == NULL || destFile == NULL)

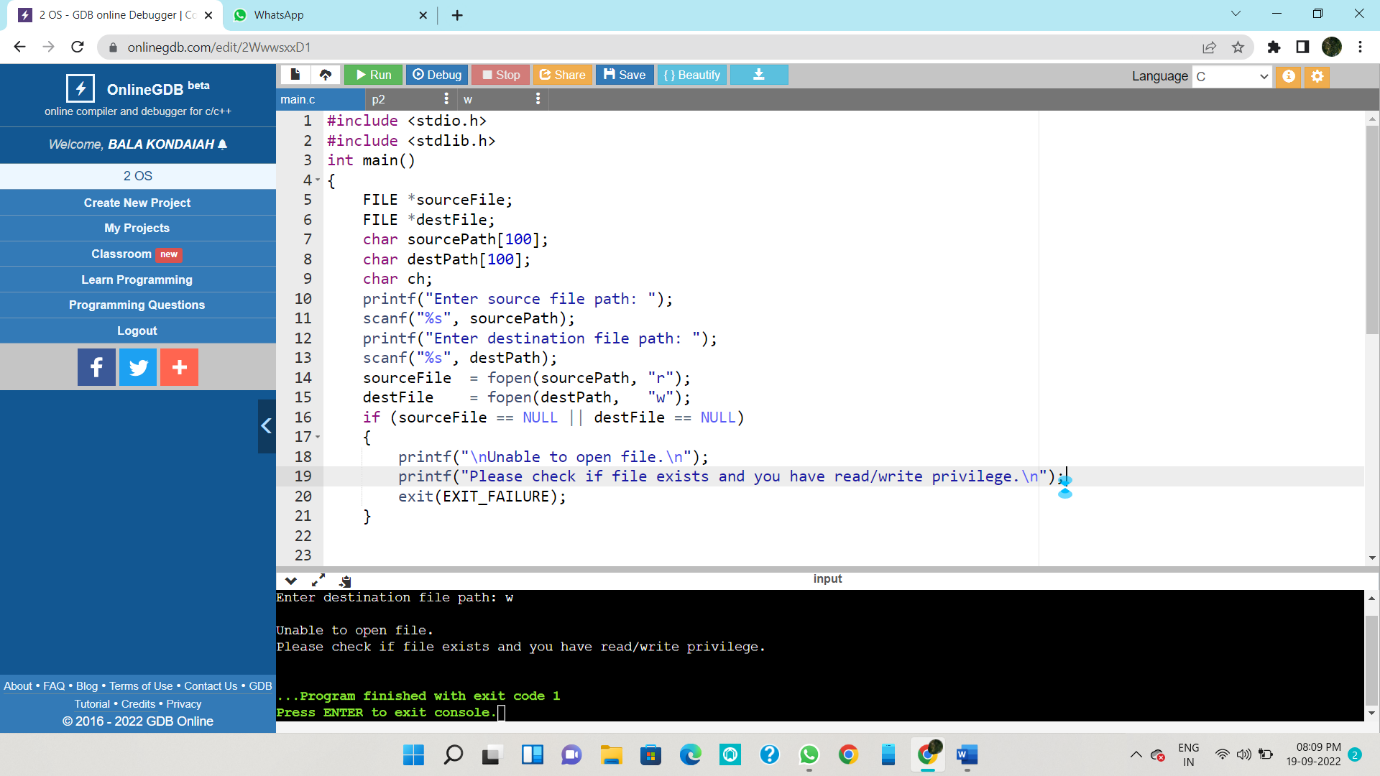
{

printf("\nUnable to open file.\n");

printf("Please check if file exists and you have read/write privilege.\n");

exit(EXIT\_FAILURE);

}

Output:-

3. Design a CPU scheduling program with C using First Come First Served technique

Program:-

#include<stdio.h>

int main()

{

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

printf("Enter total number of processes(maximum 20):");

scanf("%d",&n);

printf("nEnter Process Burst Timen");

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];

}

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

{

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

avwt+=wt[i];

avtat+=tat[i];

}

avwt/=i;

avtat/=i;

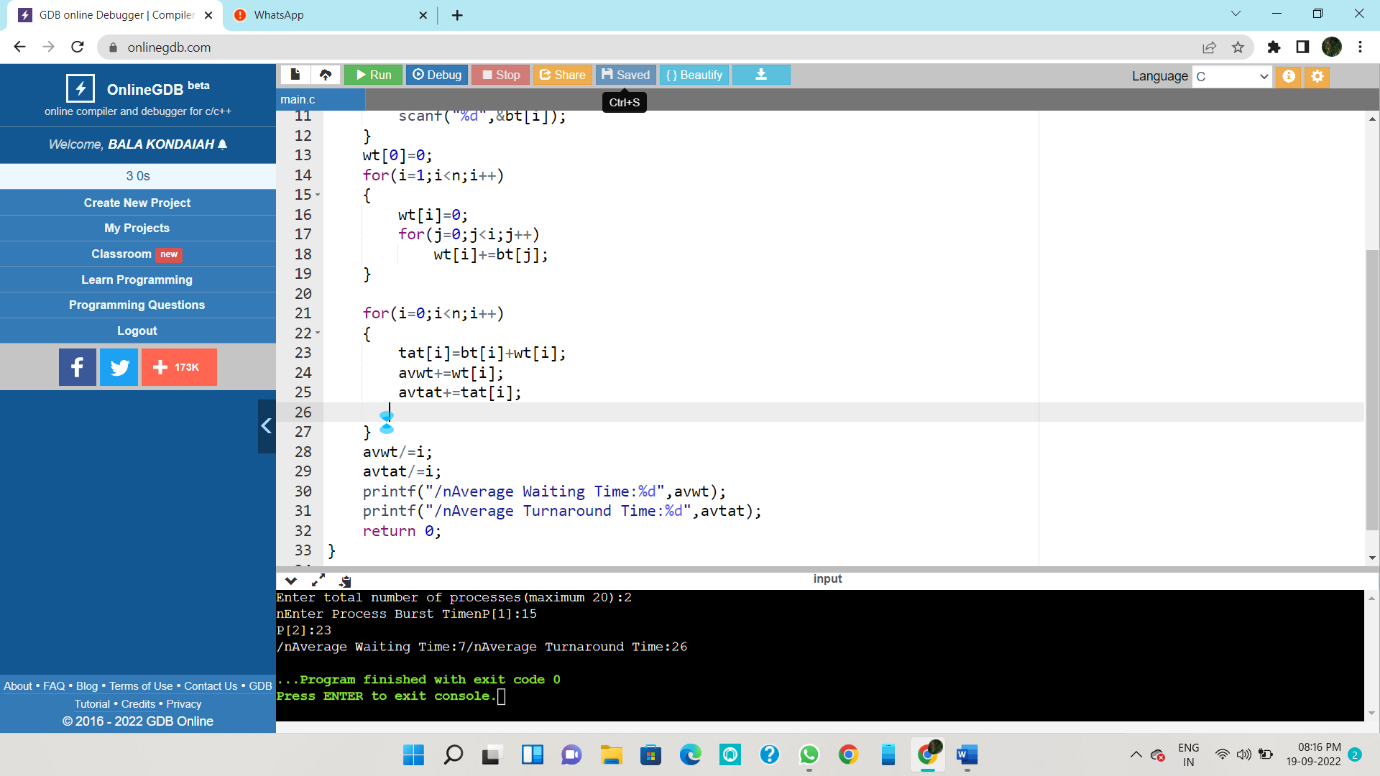
printf("Average Waiting Time:%d", avwt);

printf("Average Turnaround Time:%d",avtat);

return 0;

}

Output:-



4. Construct a scheduling program with C that selects the waiting process with the smallest execution time to execute next.

Program :-

#include<stdio.h>

int 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;

}

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;

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;

total=0;

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

{

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

total+=tat[i];

}

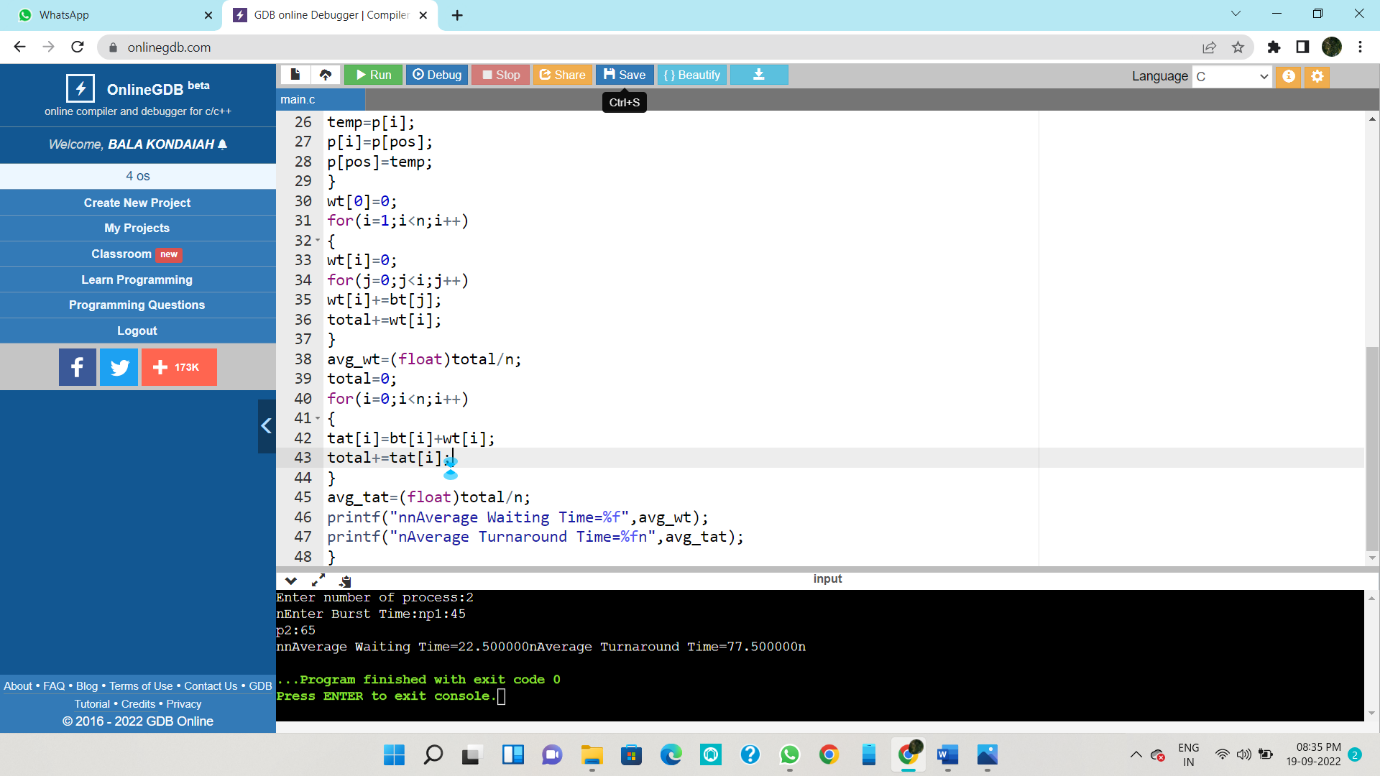
avg\_tat=(float)total/n;

printf("nnAverage Waiting Time=%f",avg\_wt);

printf("nAverage Turnaround Time=%fn",avg\_tat);

}

Output:-



5. Construct a scheduling program with C that selects the waiting process with the highest priority to execute next.

Program:-

#include <stdio.h>

void swap(int \*a,int \*b)

{

int temp=\*a;

\*a=\*b;

\*b=temp;

}

int main()

{

int n;

printf("Enter Number of Processes: ");

scanf("%d",&n);

int b[n],p[n],index[n];

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

{

printf("Enter Burst Time and Priority Value for Process %d: ",i+1);

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

index[i]=i+1;

}

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

{

int a=p[i],m=i;

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

{

if(p[j] > a)

{

a=p[j];

m=j;

}

}

swap(&p[i], &p[m]);

swap(&b[i], &b[m]);

swap(&index[i],&index[m]);

}

int t=0;

printf("Order of process Execution is\n");

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

{

printf("P%d is executed from %d to %d\n",index[i],t,t+b[i]);

t+=b[i];

}

printf("\n");

printf("Process Id Burst Time Wait Time TurnAround Time\n");

int wait\_time=0;

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

{

printf("P %d %d %d %d\n",index[i],b[i],wait\_time,wait\_time + b[i]);

wait\_time += b[i];

}

return 0;

}

Output:-

