**LAB EXAM QUESTIONS:**

1. **Write an assembly language program for adding two 8-bit data A7 A6 A5 A4 A3 A2 A1 A0 and B7 B6 B5 B4 B3 B2 B1 B0 using 8085 processor.**

LDA 0000

MOV B,A

LDA 0001

ADD B

STA 0002

HLT



1. **Write an assembly language program for subtraction of two 8-bit data A7 A6 A5 A4 A3 A2 A1 A0 and B7 B6 B5 B4 B3 B2 B1 B0 using 8085 processor.**

LDA 0000

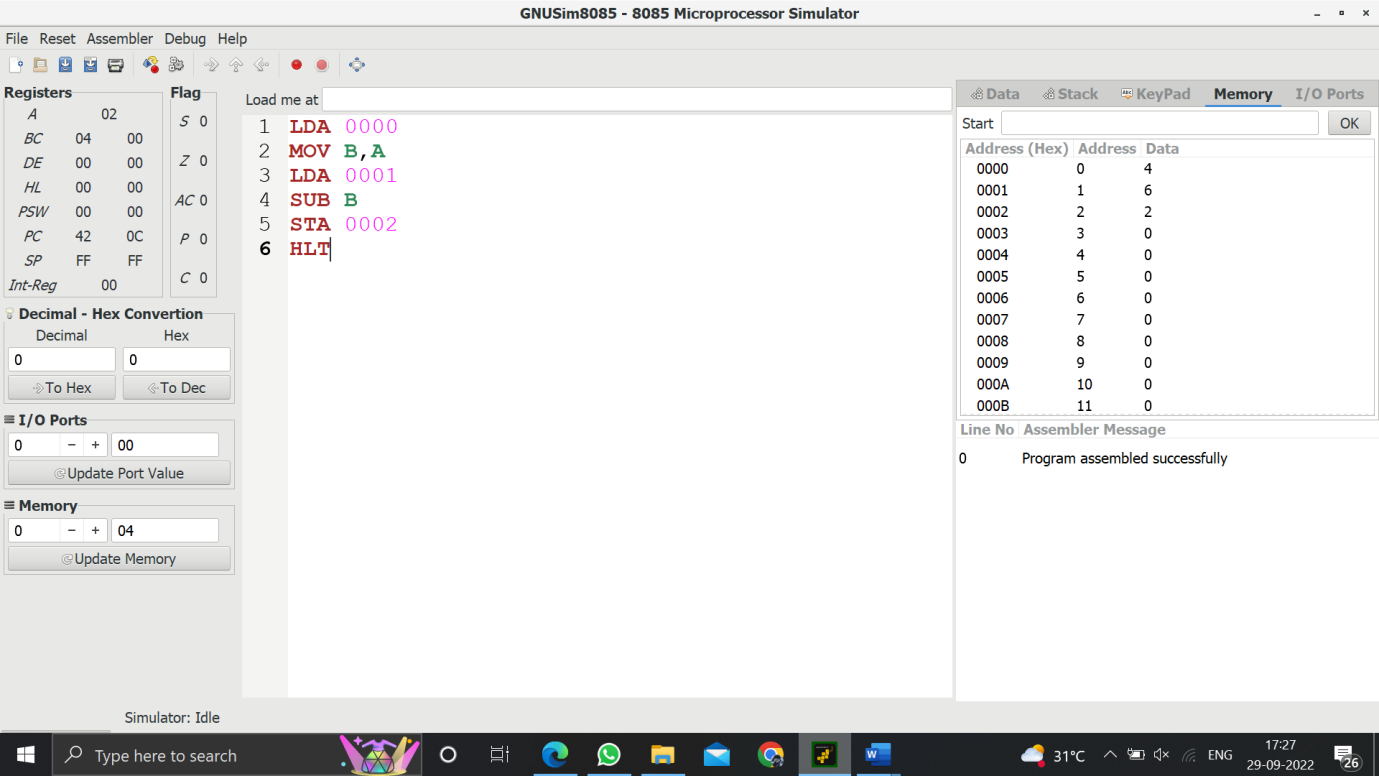
MOV B,A

LDA 0001

SUB B

STA 0002

HLT



1. **Write an assembly language program for adding two 16-bit datausing 8086 processor.**

LDA 0000

MOV B,A

LDA 0001

MOV C,A

LDA 0002

MOV D,A

LDA 0003

ADD C

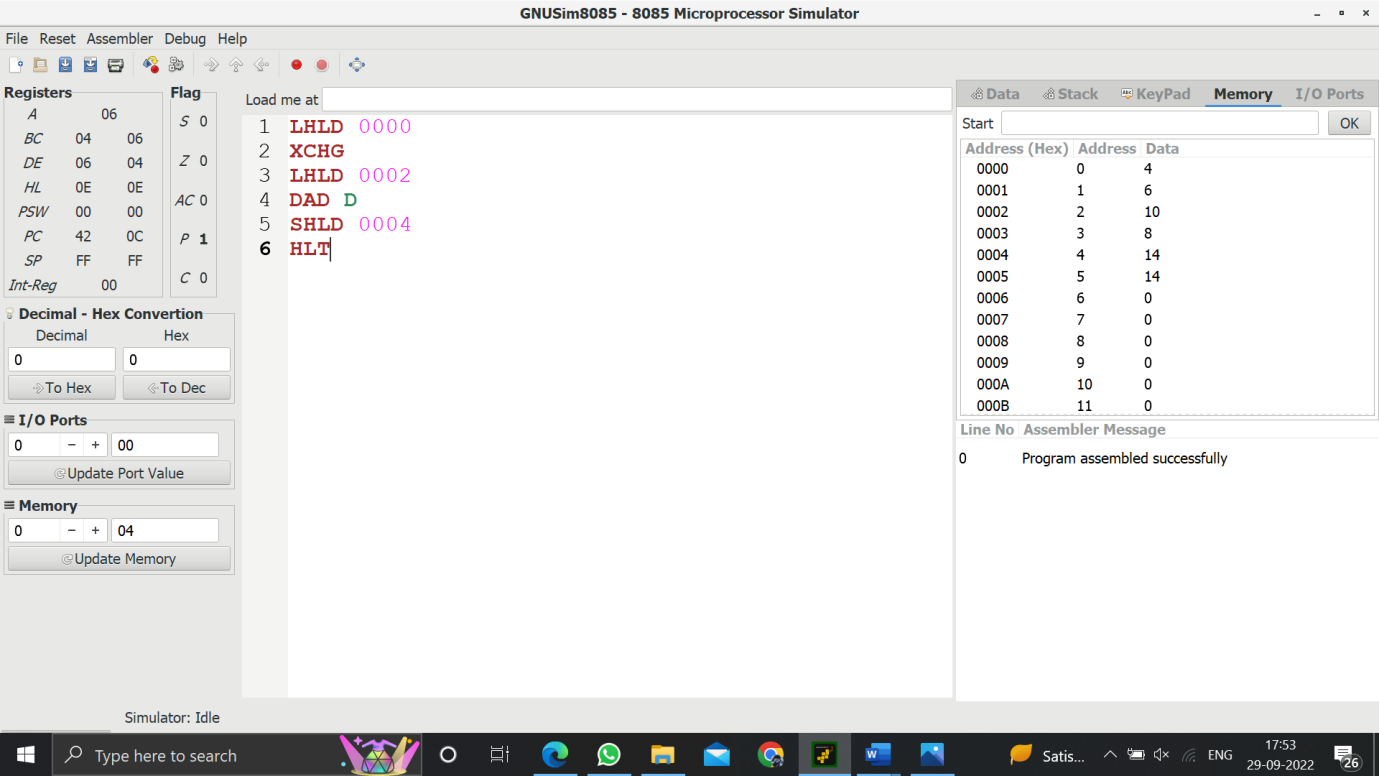
STA 0005

MOV A,D

ADD B

STA 0004

HLT



1. **Write an assembly language program for subtracting two 16-bit datausing 8086 processor.**

LDA 0000

MOV B,A

LDA 0001

MOV C,A

LDA 0002

MOV D,A

LDA 0003

SUB C

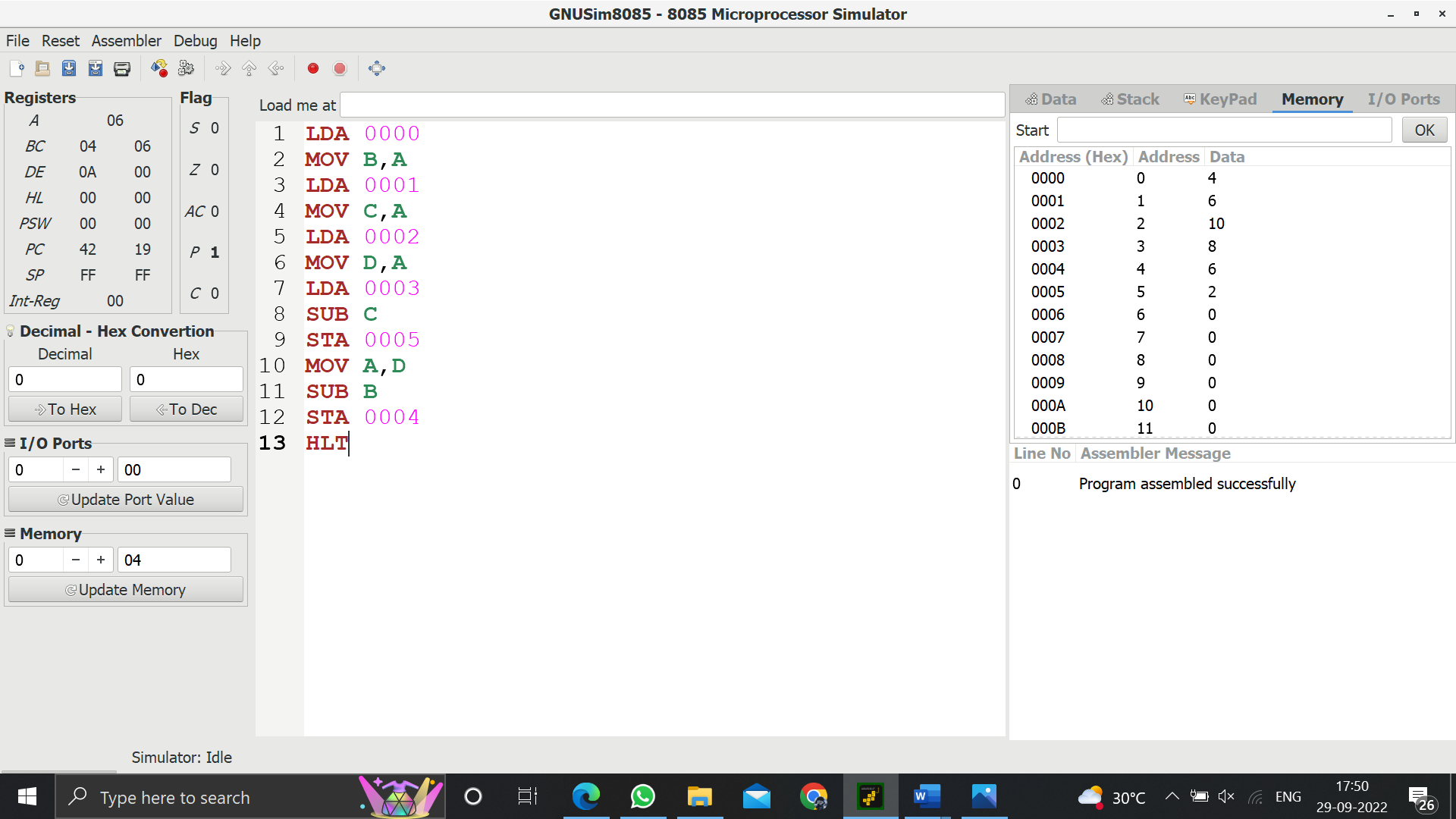
STA 0005

MOV A,D

SUB B

STA 0004

HLT



1. **Write an assembly language program for multiplication of two 8-bit data A7 A6 A5 A4 A3 A2 A1 A0 and B7 B6 B5 B4 B3 B2 B1 B0 using 8085 processor.**

;< 8 BIT MULTIPLICATION>

LDA 8500

MOV B,A

LDA 8501

MOV C,A

CPI 00

JZ LOOP

XRA A

LOOP1: ADD B

DCR C

JZ LOOP

JMP LOOP1

LOOP: STA 8502

HLT



1. **Write an assembly language program for division of two 8-bit data A7 A6 A5 A4 A3 A2 A1 A0 and B7 B6 B5 B4 B3 B2 B1 B0 using 8085 processor.**

;< 8 BIT DIVISION>

LDA 8501

MOV B,A

LDA 8500

MVI C,00

LOOP: CMP B

JC LOOP1

SUB B

INR C

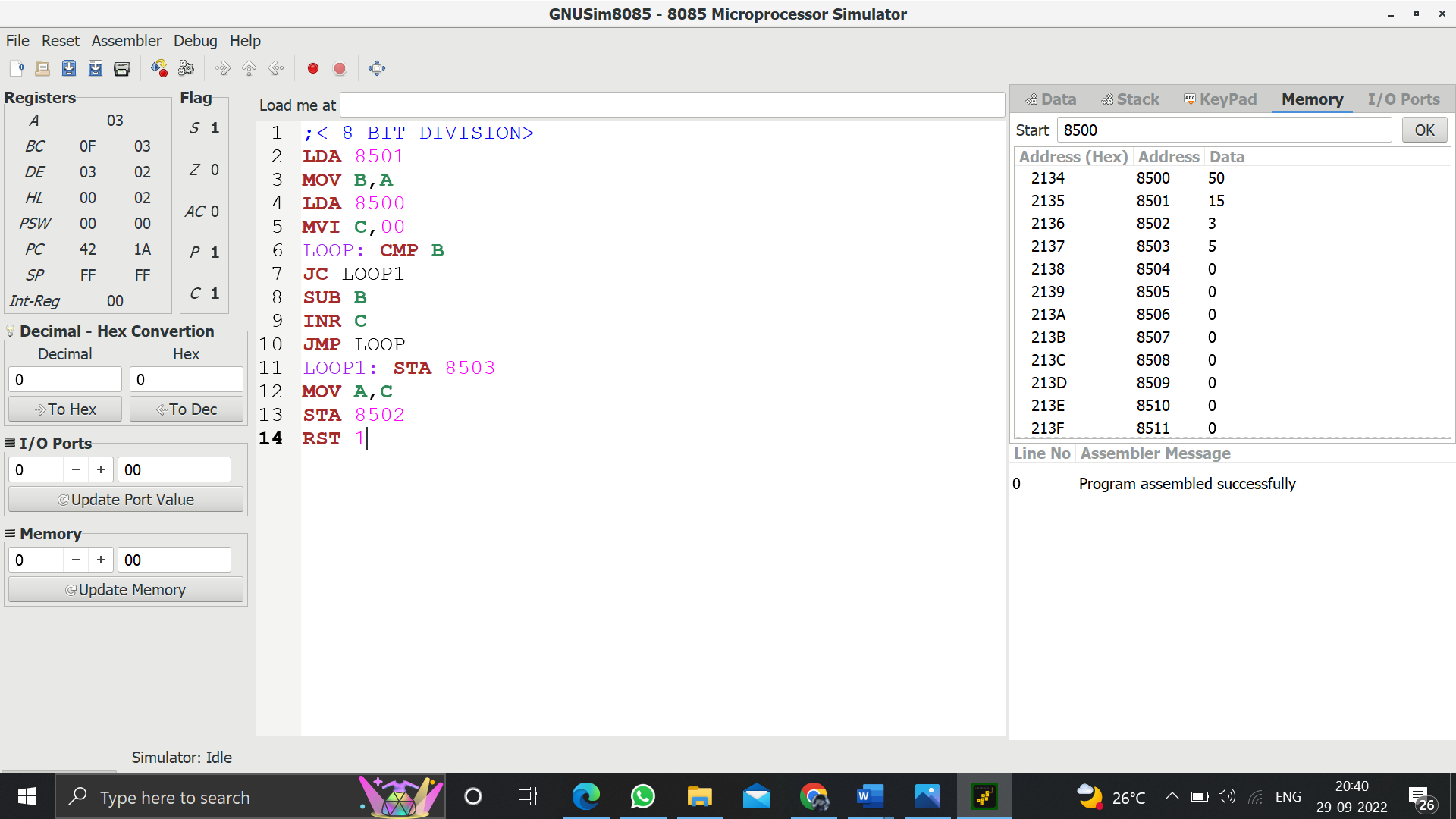
JMP LOOP

LOOP1: STA 8503

MOV A,C

STA 8502

RST 1



1. **Write an assembly language program for multiplying two 16-bit datausing 8086 processor.**

;<16 BIT MULTIPLICATION>

LHLD 0003

SPHL

LHLD 0005

XCHG

MVI A,00

LXI H,0000

LXI B,0000

LOOP1: DAD SP

JNC LOOP2

INX B

LOOP2: DCX D

CMP E

JNC SKIP

CMP D

SKIP: JNZ LOOP1

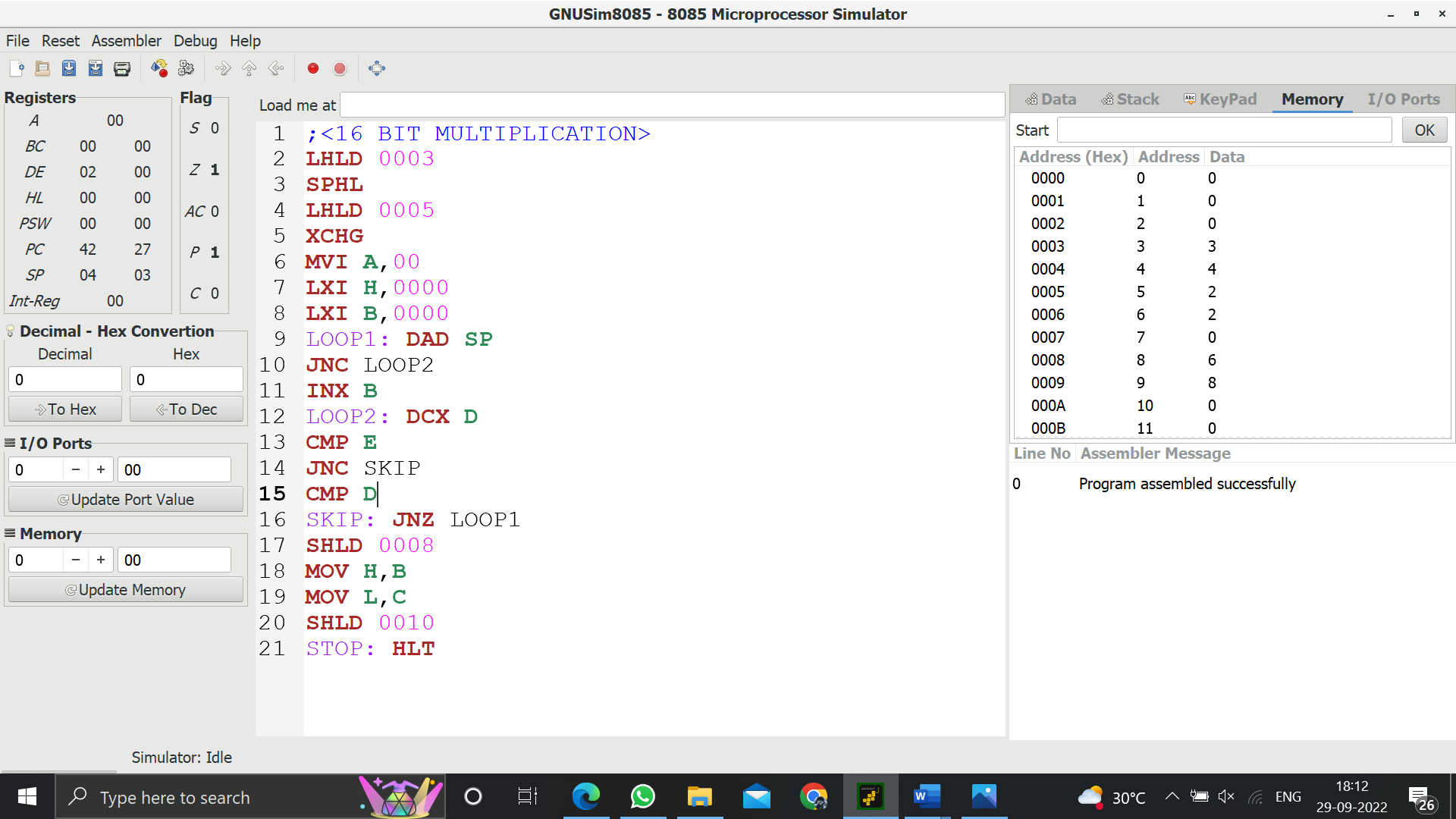
SHLD 0008

MOV H,B

MOV L,C

SHLD 0010

STOP: HLT



1. **Write an assembly language program for dividing two 16-bit datausing 8086 processor.**

LXI B,0000H

LHLD 2052

XCHG

LHLD 2050

LOOP: MOV A,L

SUB E

MOV L,A

MOV A,H

SBB D

MOV H,A

JC SKIP

INX B

JMP LOOP

SKIP: DAD D

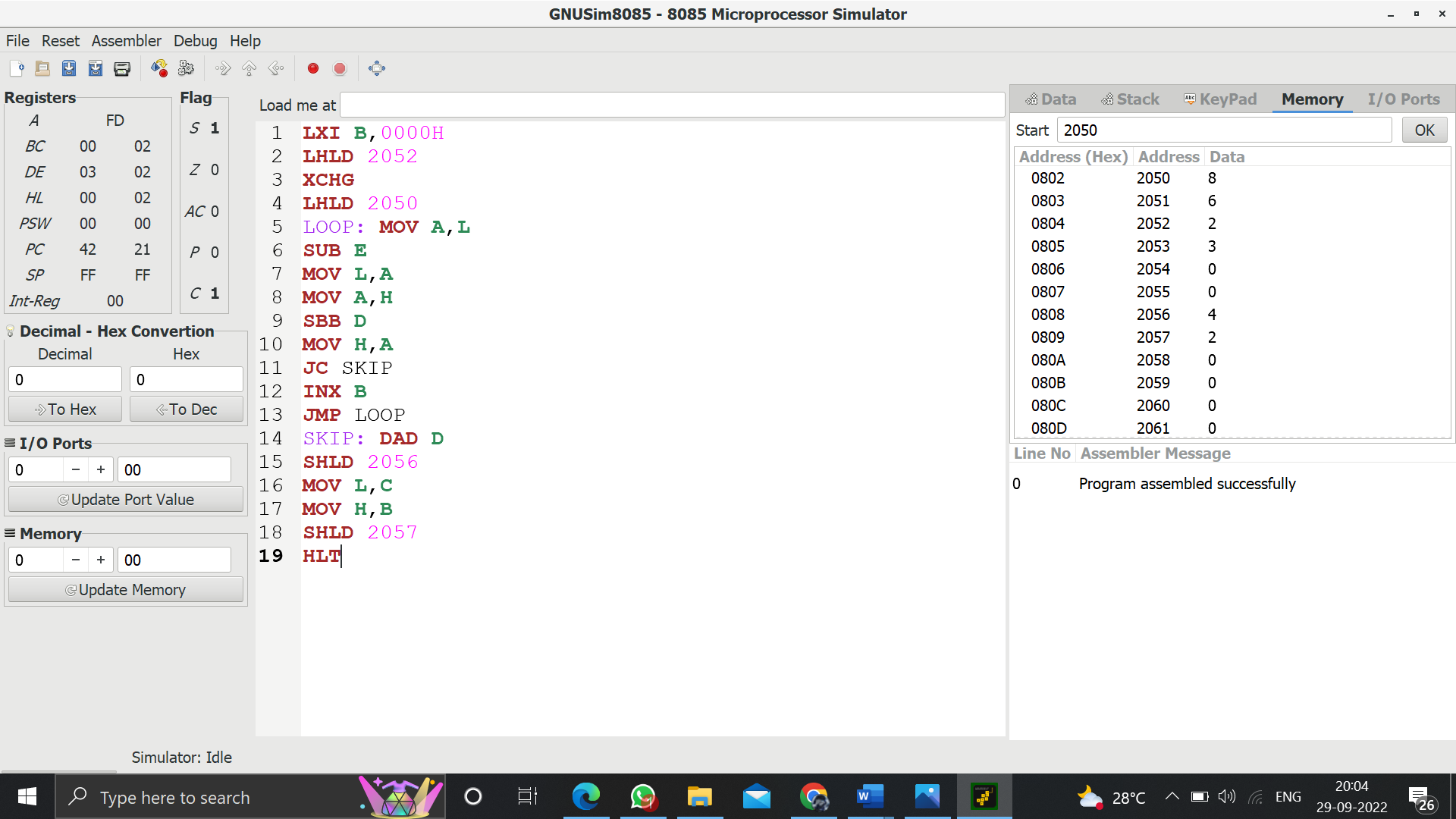
SHLD 2056

MOV L,C

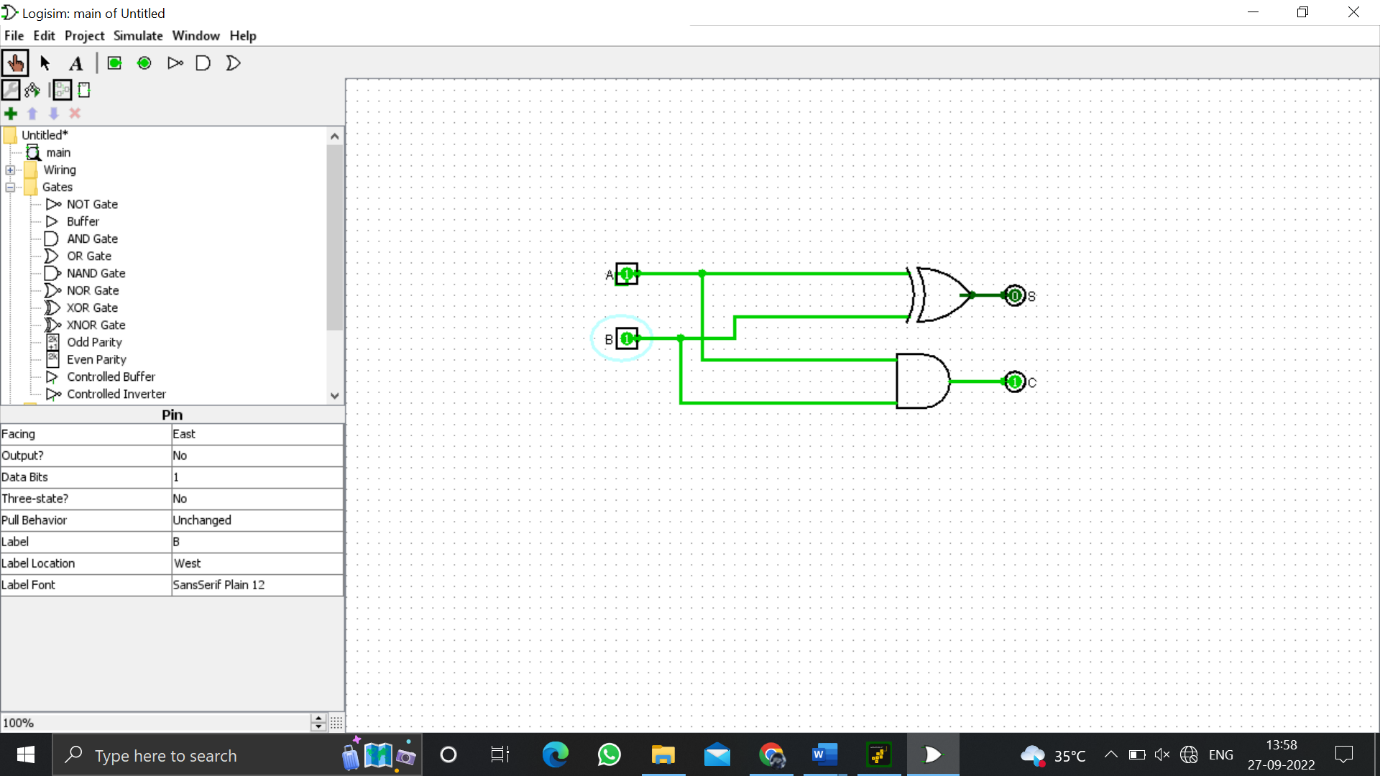
MOV H,B

SHLD 2057

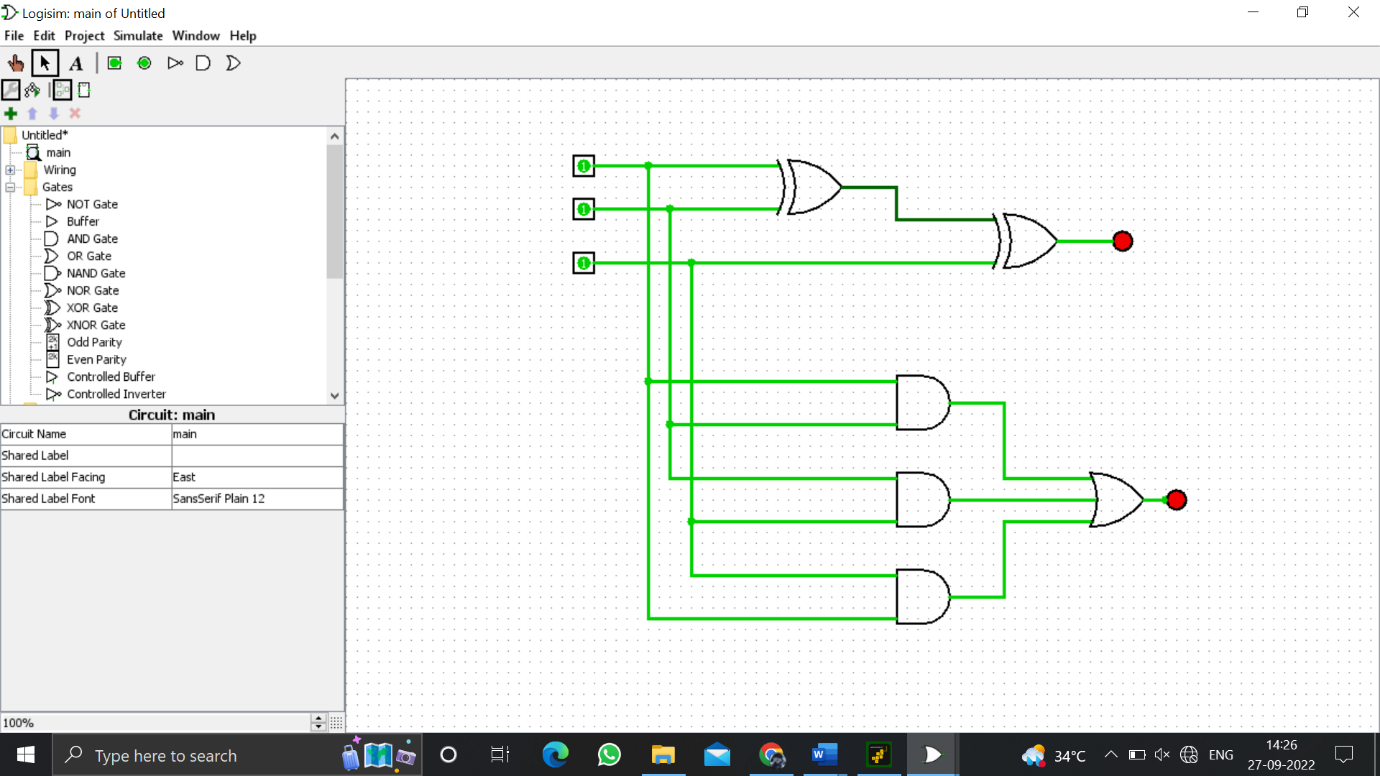
HLT



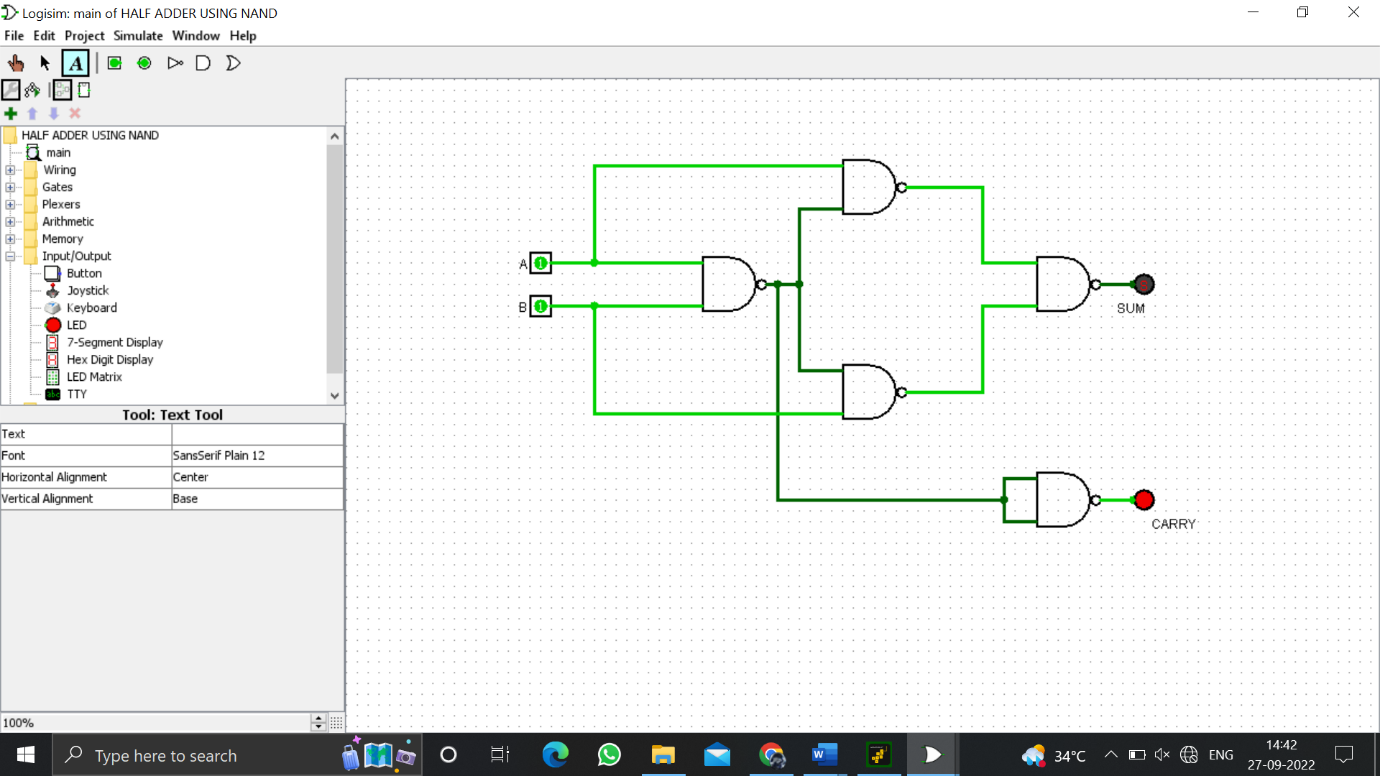
1. **Design and implement 2-bit half adder using logisim simulator.**



1. **Design and implement 3-bit full adder using logisim simulator.**



1. **Design and implement 2-bit half adder with NAND using logisim simulator.**



1. **Write an assembly language program to find factorial of n in the given number.**

LXI H,2000

MOV B,M

MVI D,01H

FACTORIAL: CALL MULTIPLY

DCR B

JNZ FACTORIAL

INX H

MOV M,D

HLT

MULTIPLY: MOV E,B

MVI A,00H

MULTIPLYLOOP: ADD D

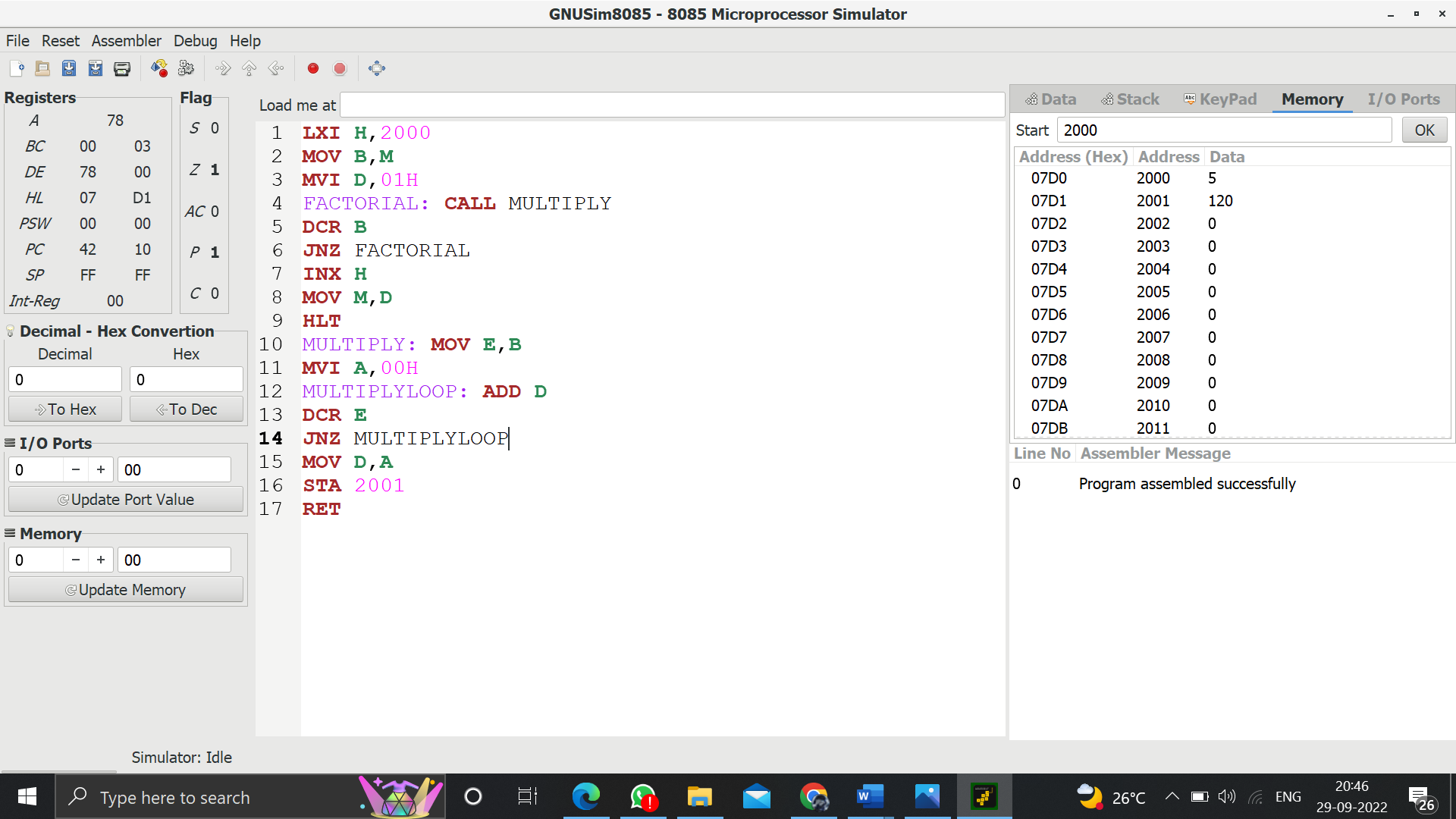
DCR E

JNZ MULTIPLYLOOP

MOV D,A

STA 2001

RET



1. **Write an assembly language program to find the largest number in an array.**

lxi h,7000

mov c,m

inx h

mov b,m

dcr c

loop: inx h

mov a,m

cmp b

jc skip

mov b,a

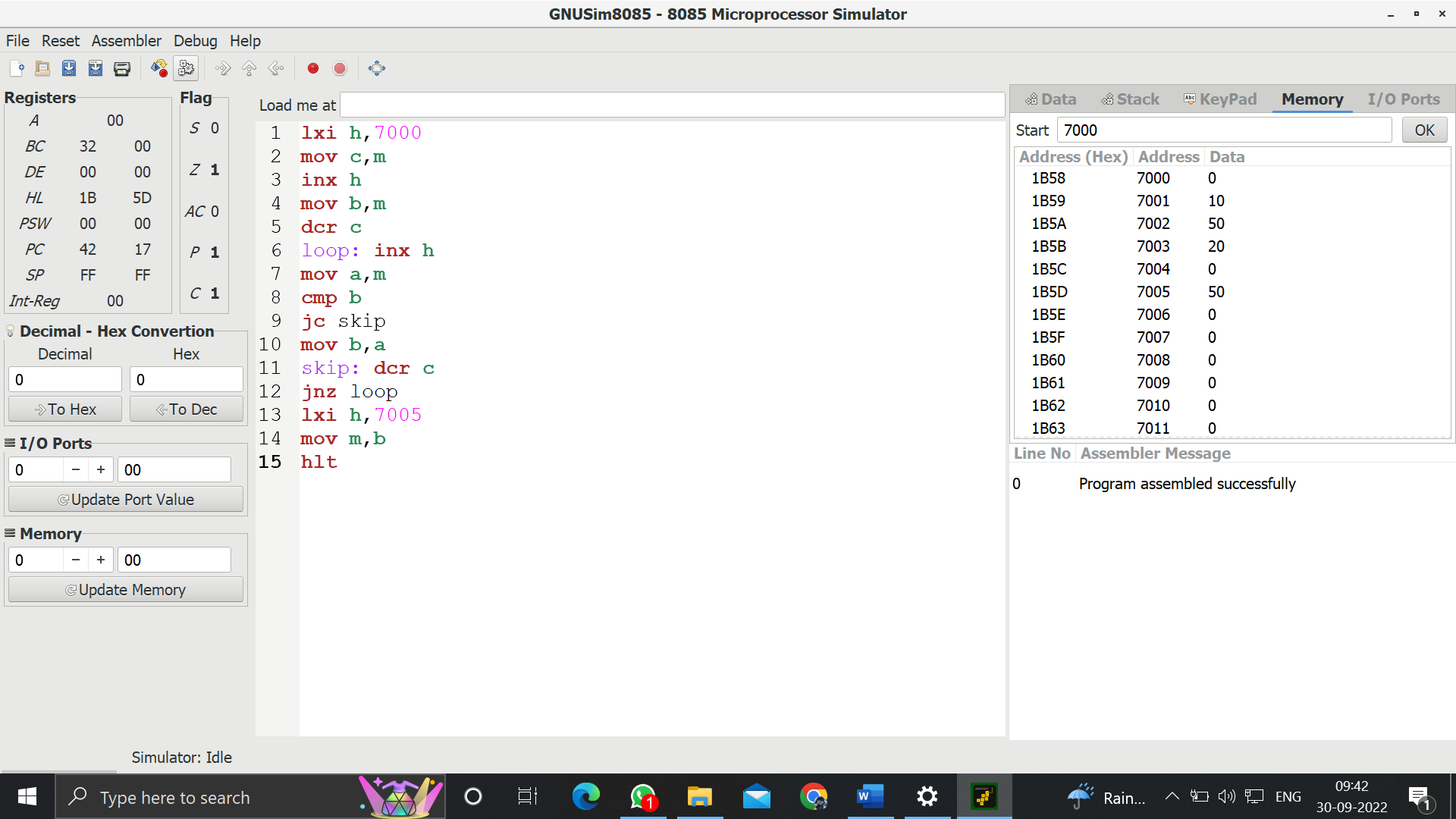
skip: dcr c

jnz loop

lxi h,7005

mov m,b

hlt



1. **Design of 2 stage pipeline for addition and subtraction of two numbers using any high level language.**

a=int(input("Enter number 1:"))

b=int(input("Enter number 2:"))

c=4

f=0

ch=int(input("1.Add, 2.Sub, 3.Mul, 4.Div : "))

if ch==1:

res=a+b

elifch==2:

res=a-b

elifch==3:

res=a\*b

elifch==4:

if b==0:

print("Denominator can't be zero")

print("Wrong input")

f=1

else:

res=a/b

else:

print("Wrong input")

f=1

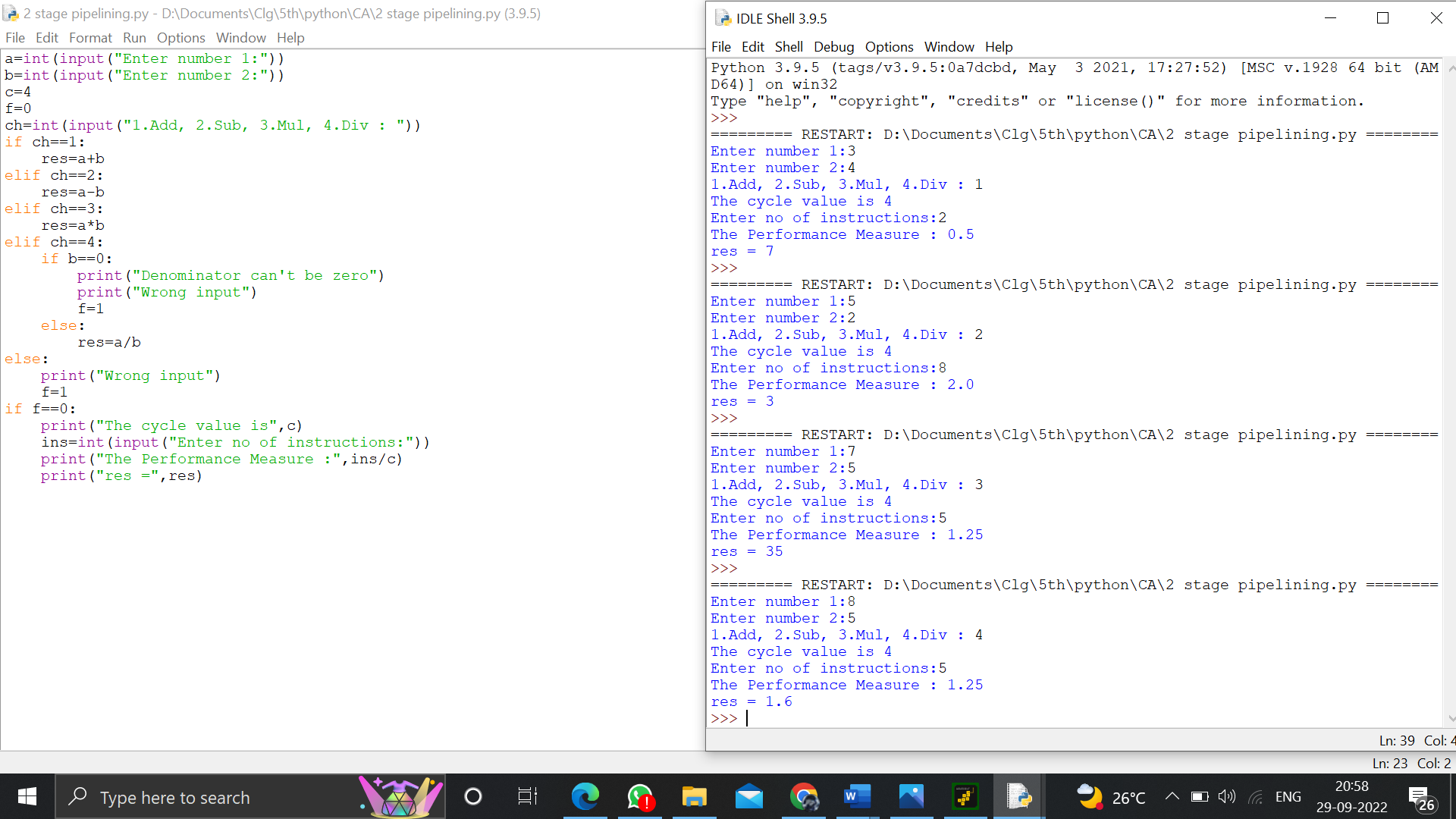
if f==0:

print("The cycle value is",c)

ins=int(input("Enter no of instructions:"))

print("The Performance Measure :",ins/c)

print("res =",res)



1. **Design of 3 stage pipeline for AND, OR, NAND of two numbers using any high level language.**

a=int(input("Enter number 1:"))

b=int(input("Enter number 2:"))

c=4

rand=a&b

ror=a|b

rnand=~rand

print("The cycle value is",c)

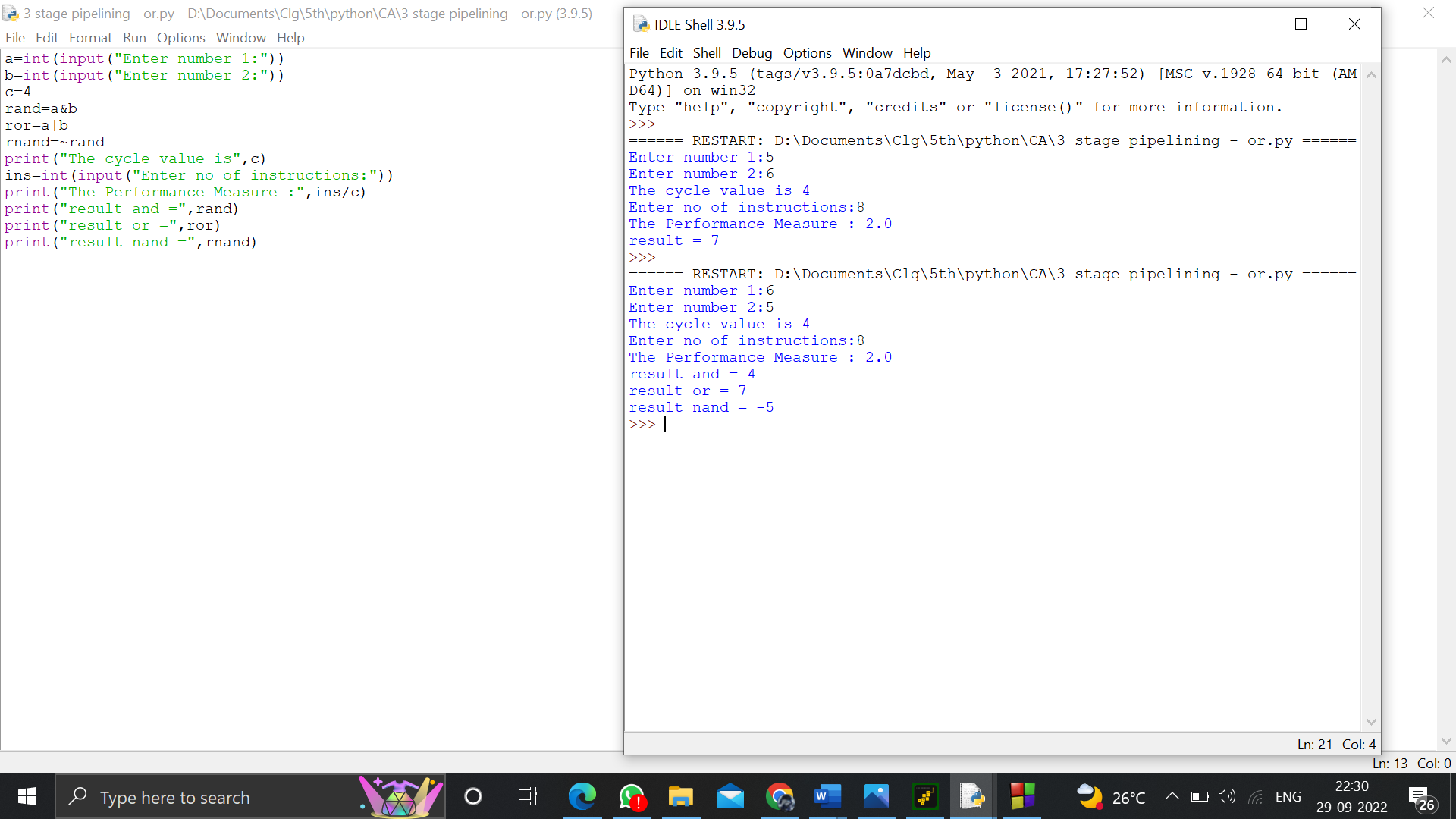
ins=int(input("Enter no of instructions:"))

print("The Performance Measure :",ins/c)

print("result and =",rand)

print("result or =",ror)

print("result nand =",rnand)



1. **Design of 4 stage pipeline for multiplication and division of two numbers using any high level language.**

a=int(input("Enter number 1:"))

b=int(input("Enter number 2:"))

c=3

f=0

ch=int(input("1.Add, 2.Sub, 3.Mul, 4.Div : "))

if ch==1:

print("Performing Addition operation:")

res=a+b

elifch==2:

print("Performing Subtraction operation:")

res=a-b

elifch==3:

print("Performing Multiplication operation:")

res=a\*b

elifch==4:

print("Performing Division operation:")

if b==0:

print("Denominator can't be zero")

print("Wrong input")

f=1

else:

res=a/b

else:

print("Wrong input")

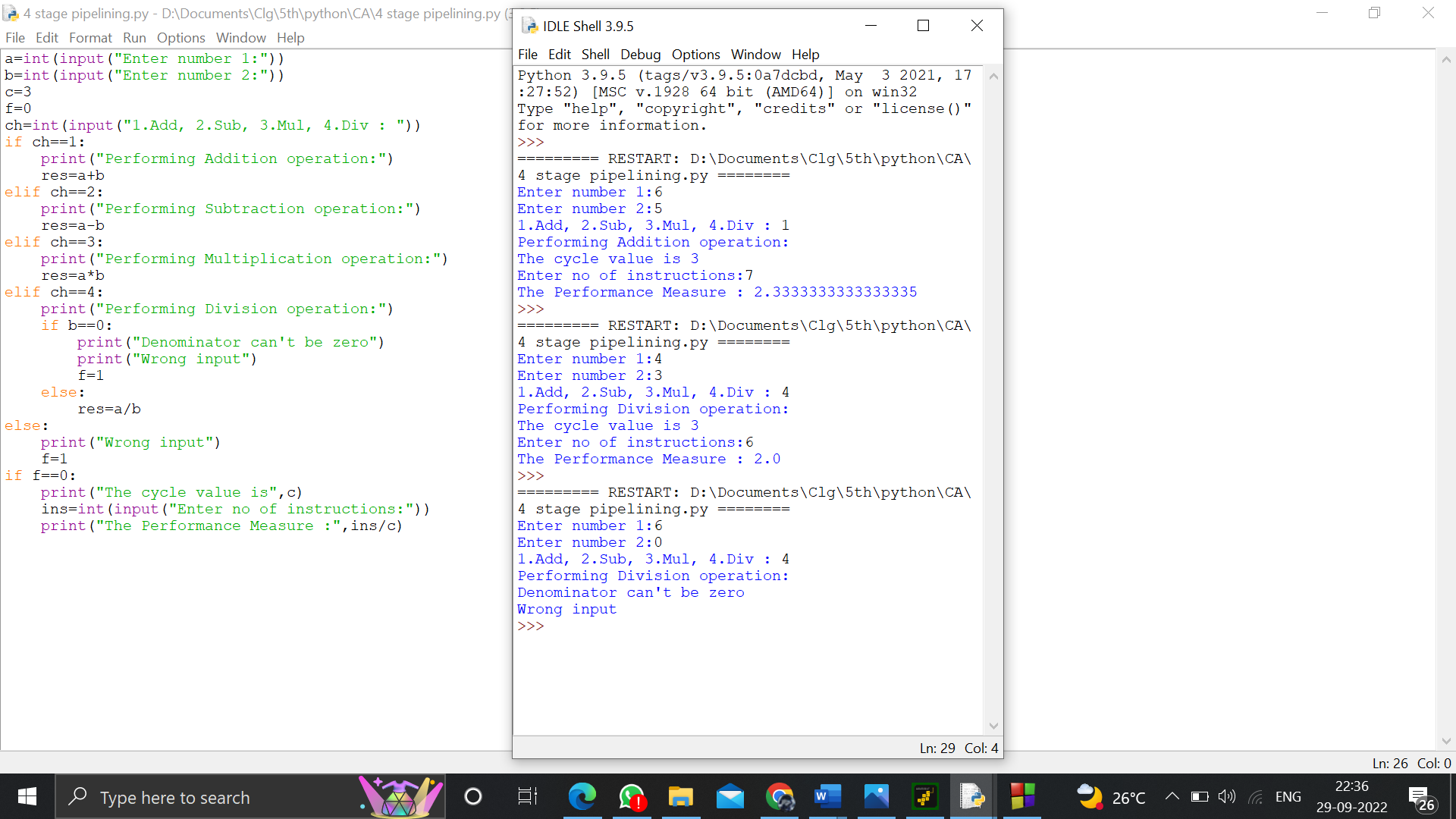
f=1

if f==0:

print("The cycle value is",c)

ins=int(input("Enter no of instructions:"))

print("The Performance Measure :",ins/c)



1. **Write a program to perform Booth’s multiplication of two signed numbers using any high level language.**

//BOOTHS ALGORITHM

#include <stdio.h>

#include <math.h>

int a = 0,b = 0, c = 0, a1 = 0, b1 = 0, com[5] = { 1, 0, 0, 0, 0};

int anum[5] = {0}, anumcp[5] = {0}, bnum[5] = {0};

int acomp[5] = {0}, bcomp[5] = {0}, pro[5] = {0}, res[5] = {0};

void binary(){

a1 = fabs(a);

b1 = fabs(b);

int r, r2, i, temp;

for (i = 0; i< 5; i++){

r = a1 % 2;

a1 = a1 / 2;

r2 = b1 % 2;

b1 = b1 / 2;

anum[i] = r;

anumcp[i] = r;

bnum[i] = r2;

if(r2 == 0){

bcomp[i] = 1;

}

if(r == 0){

acomp[i] =1;

}

}

//part for two's complementing

c = 0;

for ( i = 0; i< 5; i++){

res[i] = com[i]+ bcomp[i] + c;

if(res[i] >= 2){

c = 1;

}

else

c = 0;

res[i] = res[i] % 2;

}

for (i = 4; i>= 0; i--){

bcomp[i] = res[i];

}

//in case of negative inputs

if (a <0){

c = 0;

for (i = 4; i>= 0; i--){

res[i] = 0;

}

for ( i = 0; i< 5; i++){

res[i] = com[i] + acomp[i] + c;

if (res[i] >= 2){

c = 1;

}

else

c = 0;

res[i] = res[i]%2;

}

for (i = 4; i>= 0; i--){

anum[i] = res[i];

anumcp[i] = res[i];

}

}

if(b < 0){

for (i = 0; i< 5; i++){

temp = bnum[i];

bnum[i] = bcomp[i];

bcomp[i] = temp;

}

}

}

void add(int num[]){

int i;

c = 0;

for ( i = 0; i< 5; i++){

res[i] = pro[i] + num[i] + c;

if (res[i] >= 2){

c = 1;

}

else{

c = 0;

}

res[i] = res[i]%2;

}

for (i = 4; i>= 0; i--){

pro[i] = res[i];

printf("%d",pro[i]);

}

printf(":");

for (i = 4; i>= 0; i--){

printf("%d", anumcp[i]);

}

}

void arshift(){//for arithmetic shift right

int temp = pro[4], temp2 = pro[0], i;

for (i = 1; i<5 ;i++){//shift the MSB of product

pro[i-1] = pro[i];

}

pro[4] = temp;

for (i = 1; i<5 ;i++){//shift the LSB of product

anumcp[i-1] = anumcp[i];

}

anumcp[4] = temp2;

printf("\nAR-SHIFT: ");//display together

for (i = 4; i>= 0; i--){

printf("%d",pro[i]);

}

printf(":");

for(i = 4; i>= 0; i--){

printf("%d", anumcp[i]);

}

}

void main(){

int i, q = 0;

printf("\t\tBOOTH'S MULTIPLICATION ALGORITHM");

printf("\nEnter two numbers to multiply: ");

printf("\nBoth must be less than 16");

//simulating for two numbers each below 16

do{

printf("\nEnter A: ");

scanf("%d",&a);

printf("Enter B: ");

scanf("%d", &b);

}while(a >=16 || b >=16);

printf("\nExpected product = %d", a \* b);

binary();

printf("\n\nBinary Equivalents are: ");

printf("\nA = ");

for (i = 4; i>= 0; i--){

printf("%d", anum[i]);

}

printf("\nB = ");

for (i = 4; i>= 0; i--){

printf("%d", bnum[i]);

}

printf("\nB'+ 1 = ");

for (i = 4; i>= 0; i--){

printf("%d", bcomp[i]);

}

printf("\n\n");

for (i = 0;i< 5; i++){

if (anum[i] == q){//just shift for 00 or 11

printf("\n-->");

arshift();

q = anum[i];

}

else if(anum[i] == 1 && q == 0){//subtract and shift for 10

printf("\n-->");

printf("\nSUB B: ");

add(bcomp);//add two's complement to implement subtraction

arshift();

q = anum[i];

}

else{//add ans shift for 01

printf("\n-->");

printf("\nADD B: ");

add(bnum);

arshift();

q = anum[i];

}

}

printf("\nProduct is = ");

for (i = 4; i>= 0; i--){

printf("%d", pro[i]);

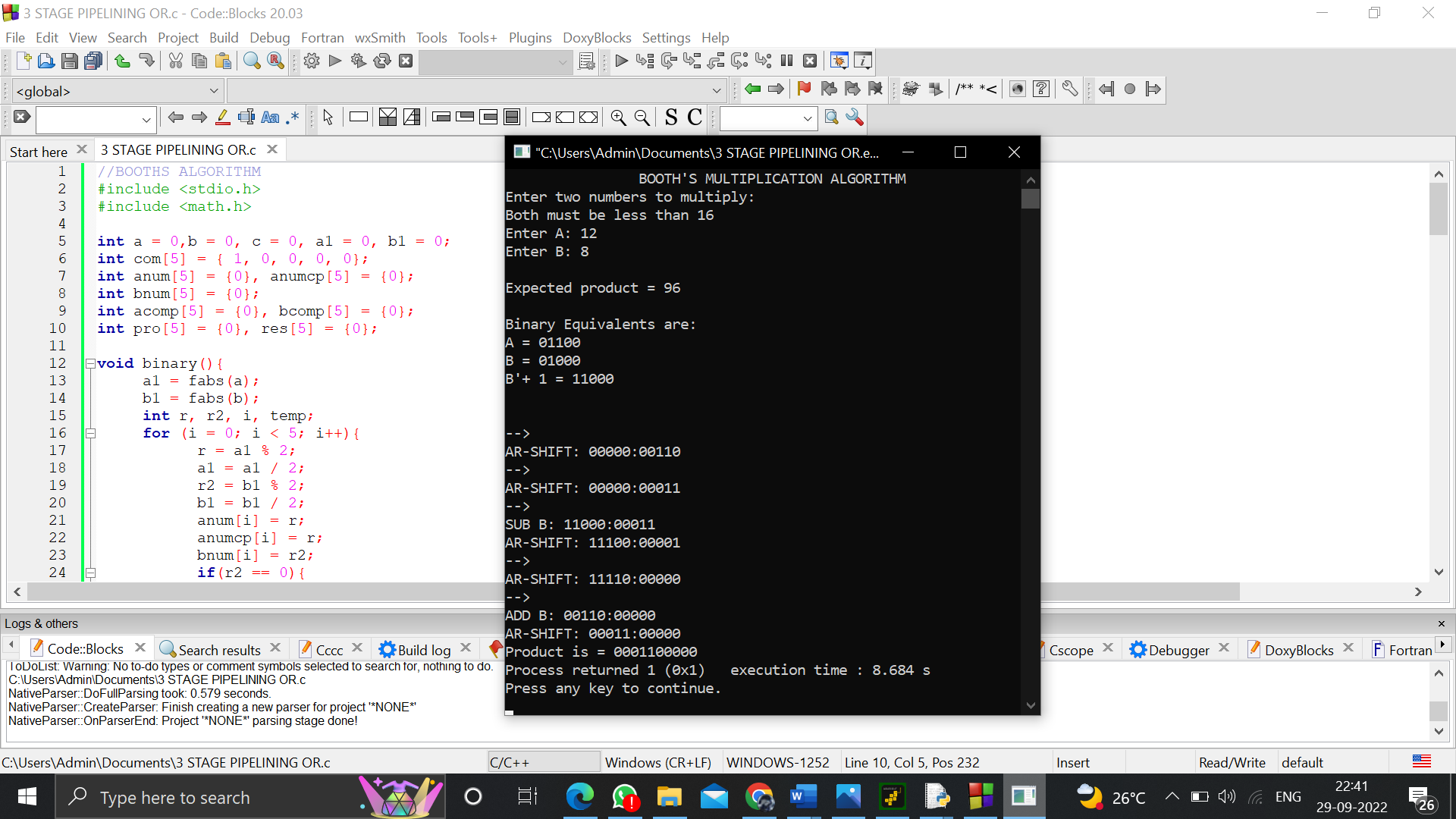
}

for (i = 4; i>= 0; i--){

printf("%d", anumcp[i]);

}

}



1. **Write a program to perform Restoring Division of two numbers using any high level language.**

#include<stdlib.h>

#include<stdio.h>

int acum[100]={0} ;

void add(int acum[],int b[],int n);

int q[100],b[100];

int main()

{

int x,y;

printf("Enter the Number and Divisor :");

scanf("%d%d",&x,&y);

int i=0;

while(x>0||y>0)

{

if(x>0)

{

q[i]=x%2;

x=x/2;

}

else

{

q[i]=0;

}

if(y>0)

{

b[i]=y%2;

y=y/2;

}

else

{

b[i]=0;

}

i++;

}

int n=i;

int bc[50];

printf("\n");

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

{

if(b[i]==0)

{

bc[i]=1;

}

else

{

bc[i]=0;

}

}

bc[n]=1;

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

{

if(bc[i]==0)

{

bc[i]=1;

i=n+2;

}

else

{

bc[i]=0;

}

}

int l;

b[n]=0;

int k=n;

int n1=n+n-1;

int j,mi=n-1;

for(i=n;i!=0;i--)

{

for(j=n;j>0;j--)

{

acum[j]=acum[j-1];

}

acum[0]=q[n-1];

for(j=n-1;j>0;j--)

{

q[j]=q[j-1];

}

add(acum,bc,n+1);

if(acum[n]==1)

{

q[0]=0;

add(acum,b,n+1);

}

else

{

q[0]=1;

}

}

printf("\nQuoient : ");

for( l=n-1;l>=0;l--)

{

printf("%d",q[l]);

}

printf("\nRemainder : ");

for( l=n;l>=0;l--)

{

printf("%d",acum[l]);

}

return 0;

}

void add(int acum[],int bo[],int n)

{

int i=0,temp=0,sum=0;

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

{

sum=0;

sum=acum[i]+bo[i]+temp;

if(sum==0)

{

acum[i]=0;

temp=0;

}

else if (sum==2)

{

acum[i]=0;

temp=1;

}

else if(sum==1)

{

acum[i]=1;

temp=0;

}

else if(sum==3)

{

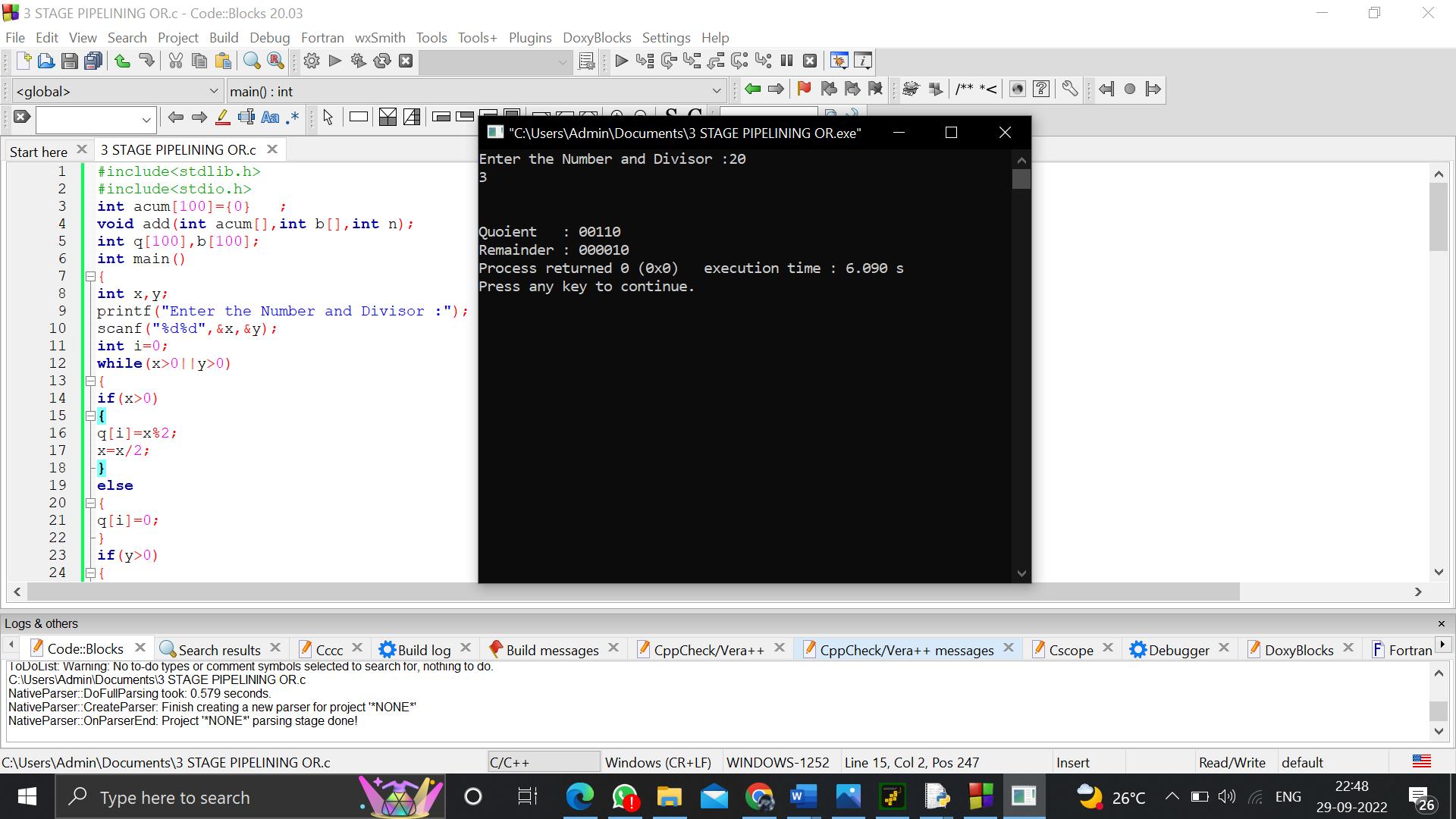
acum[i]=1;

temp=1;

}

}

}

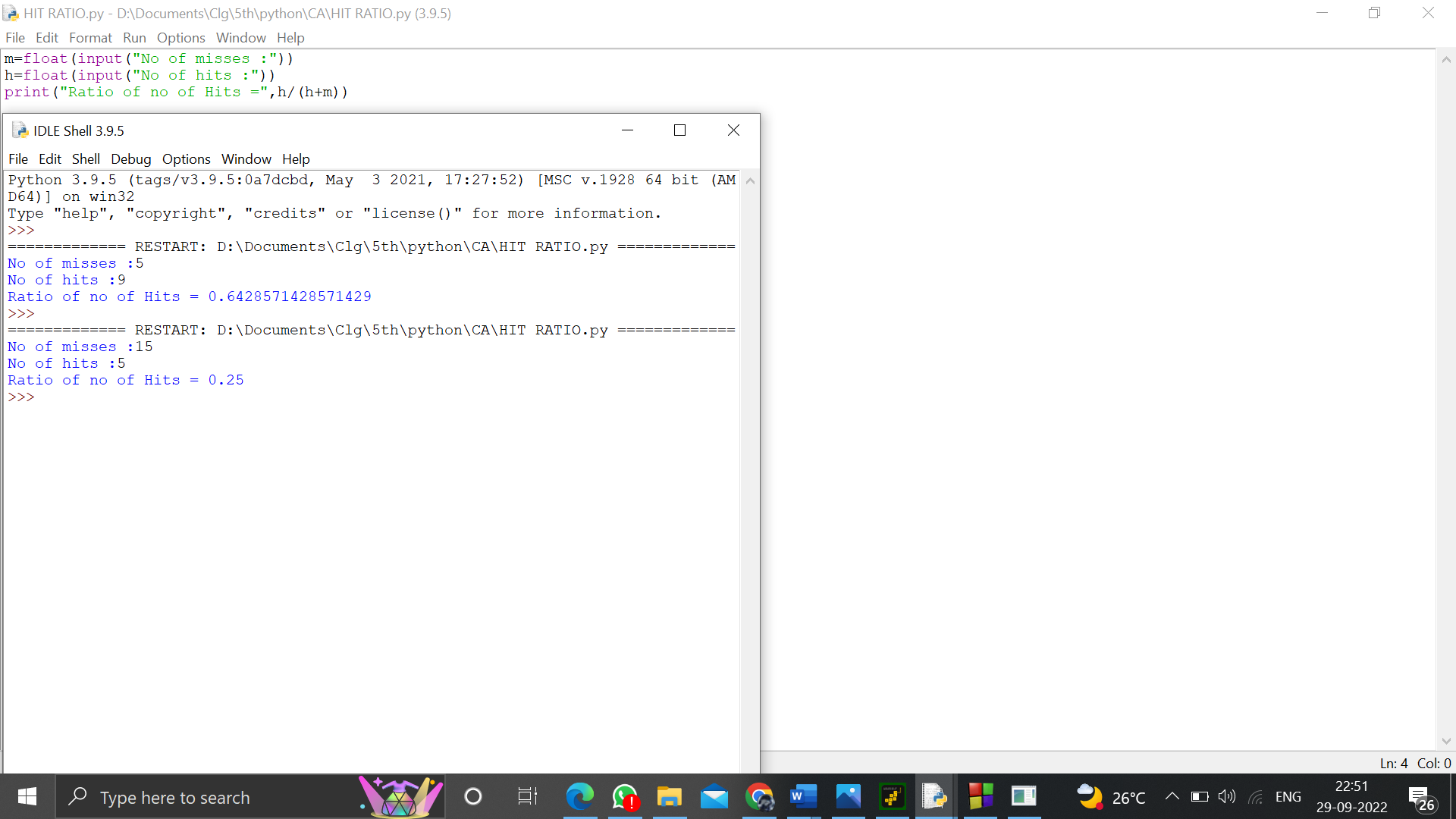


1. **Write a program to find the Hit ratio for the given number of Hits and Misses in Cache memory using any high level language.**

m=float(input("No of misses :"))

h=float(input("No of hits :"))

print("Ratio of no of Hits =",h/(h+m))



1. **Write an assembly language program to find 1’s and 2’s complement of 8 bit number.**

;< COMPLEMENT >

lda 0000

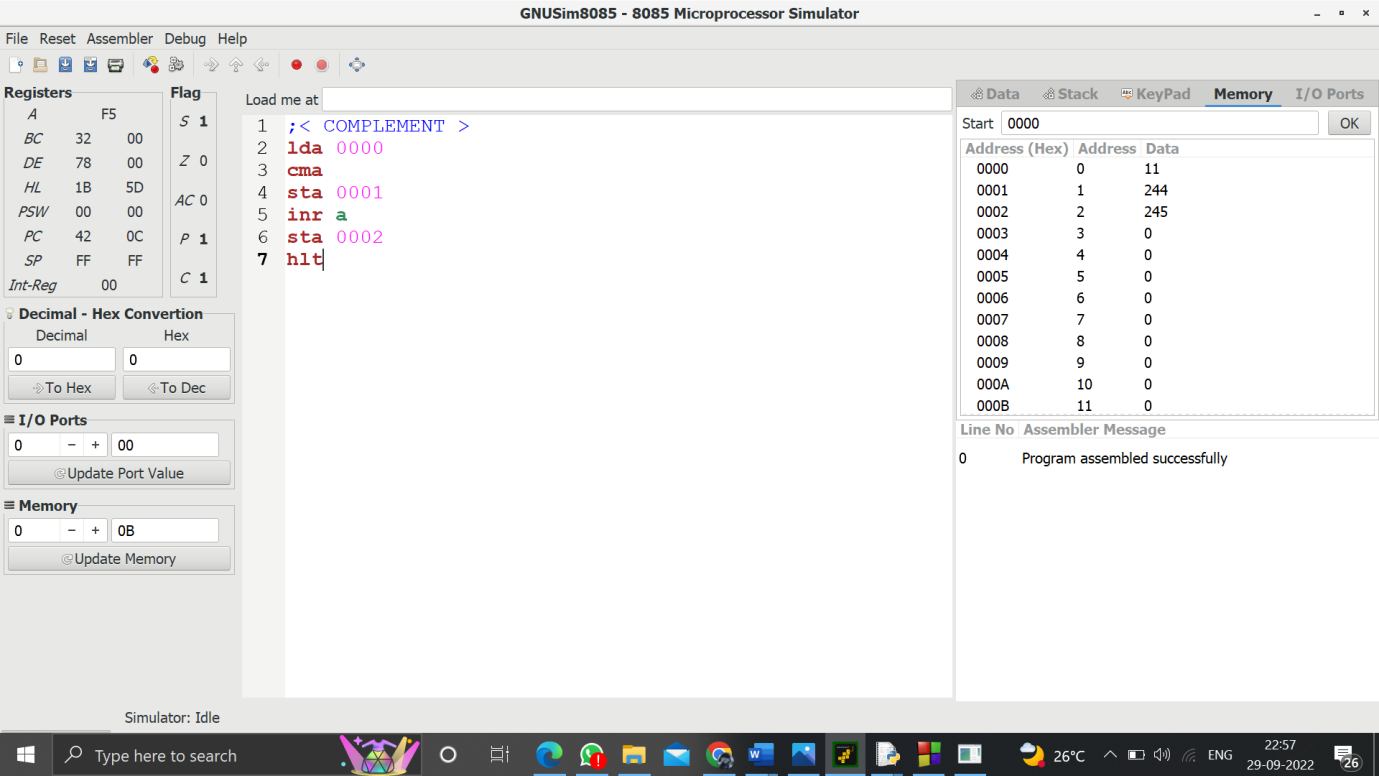
cma

sta 0001

inr a

sta 0002

hlt



1. **Write a program to convert Decimal number to Binary number using any high level language.**

n=int(input("Enter decimal :"))

sum=0

m=1

while n>0:

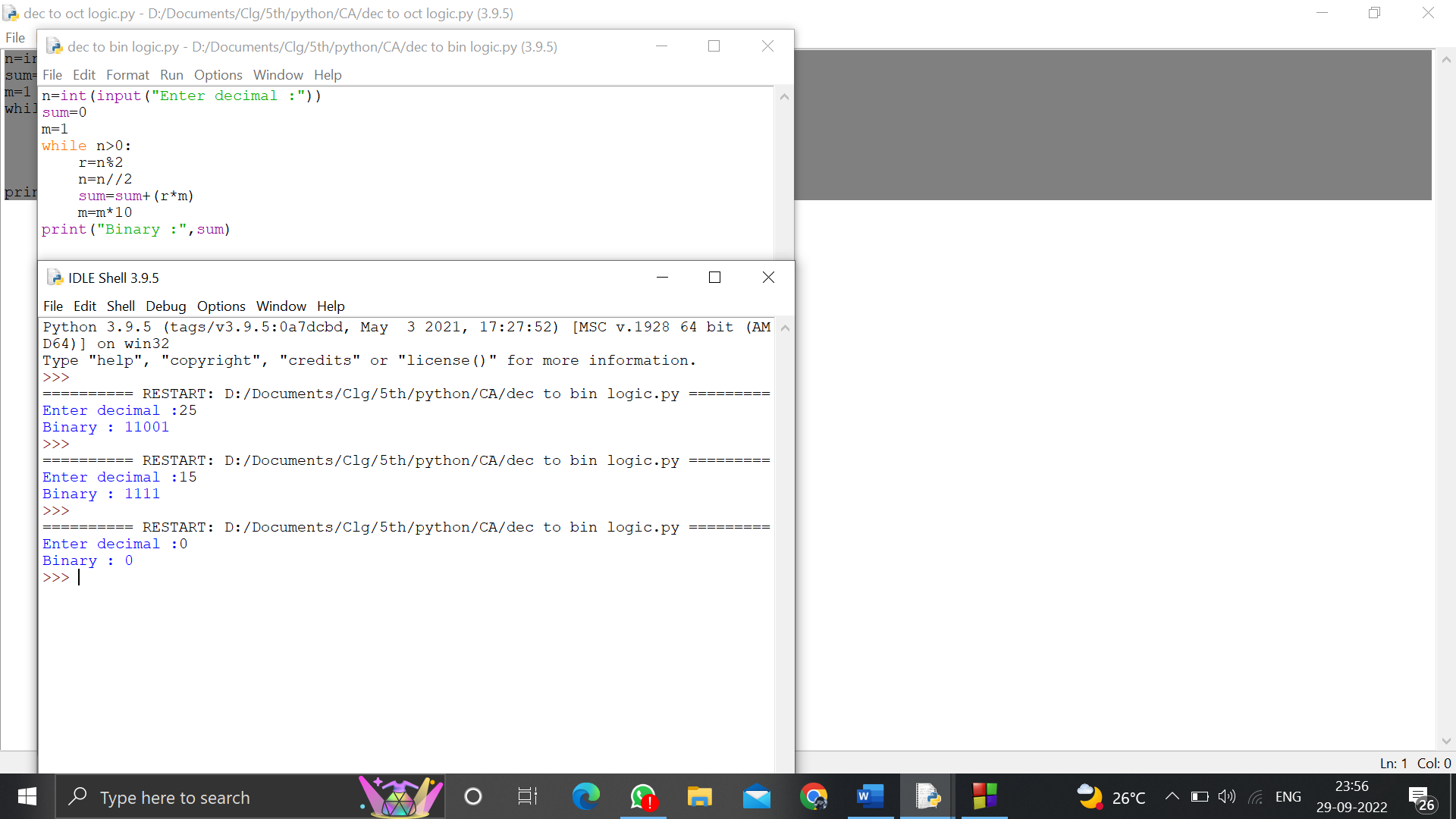
r=n%2

n=n//2

sum=sum+(r\*m)

m=m\*10

print("Binary :",sum)



1. **Write a program to convert Decimal number to an Octal number using any high level language.**

n=int(input("Enter decimal :"))

sum=0

m=1

while n>0:

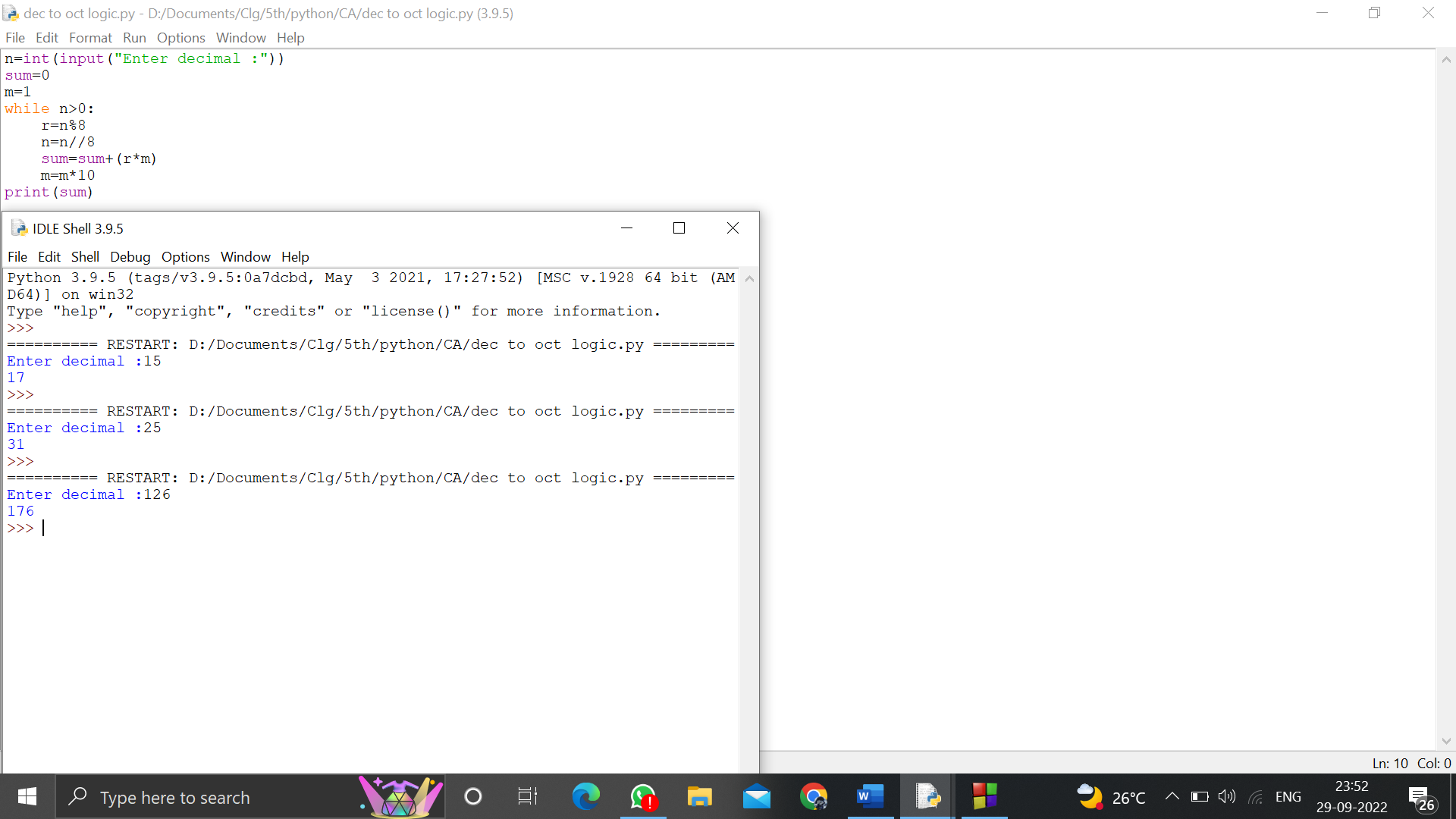
r=n%8

n=n//8

sum=sum+(r\*m)

m=m\*10

print(sum)



1. **Write a program to convert Binary number to Decimal number using any high level language.**

n=int(input("Enter Binary :"))

sum=0

a=0

while n>0:

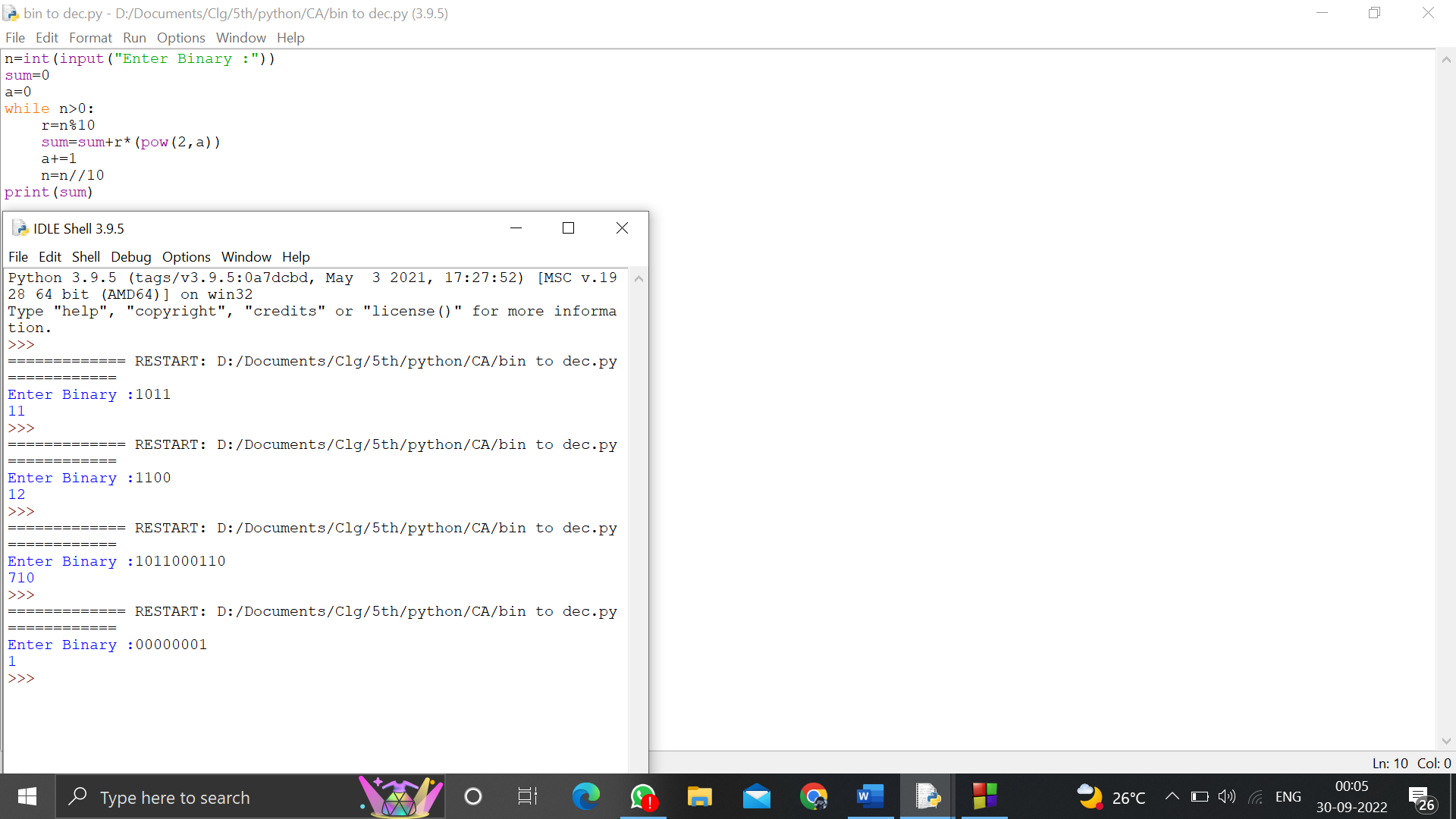
r=n%10

sum=sum+r\*(pow(2,a))

a+=1

n=n//10

print(sum)



1. **Write a program to find the CPU performance of a processor using any high level language.**

p=int(input("Enter no of processors:"))

ct=[]

for i in range(0,p):

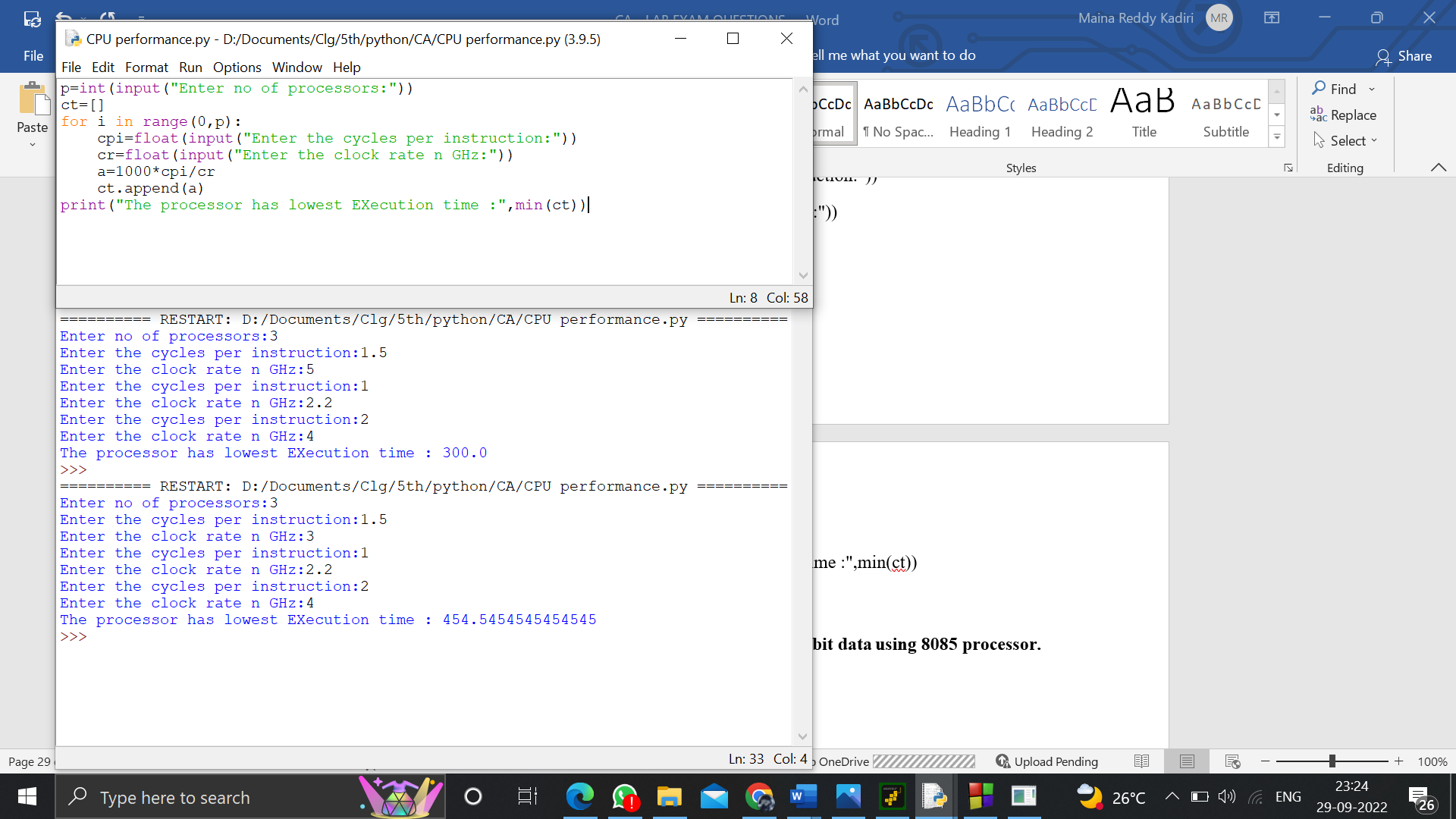
cpi=float(input("Enter the cycles per instruction:"))

cr=float(input("Enter the clock rate n GHz:"))

a=1000\*cpi/cr

ct.append(a)

print("The processor has lowest EXecution time :",min(ct))



1. **Write an assembly language program to swap two 8-bit datausing 8085 processor.**

LDA 0000

MOV B,A

LDA 0001

MOV C,A

STA 0003

MOV A,B

STA 0004

HLT

