**SET-1**

1. Write a program to simulate the page replacement algorithm First in First Out (FIFO)

#include<stdio.h>

int main()

{

int i, j, k, f, pf=0, count=0, rs[25], m[10], n;

printf("\n Enter the length of reference string : ");

scanf("%d",&n);

printf("\n Enter the reference string : ");

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

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

printf("\n Enter no. of frames : ");

scanf("%d",&f);

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

m[i]=-1;

printf("\n The Page Replacement Process is : \n");

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

{

for(k=0;k<f;k++)

{

if(m[k]==rs[i])

break;

}

if(k==f)

{

m[count++]=rs[i];

pf++;

}

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

printf("\t%d",m[j]);

if(k==f)

printf("\tPF No. --> %d",pf);

printf("\n");

if(count==f)

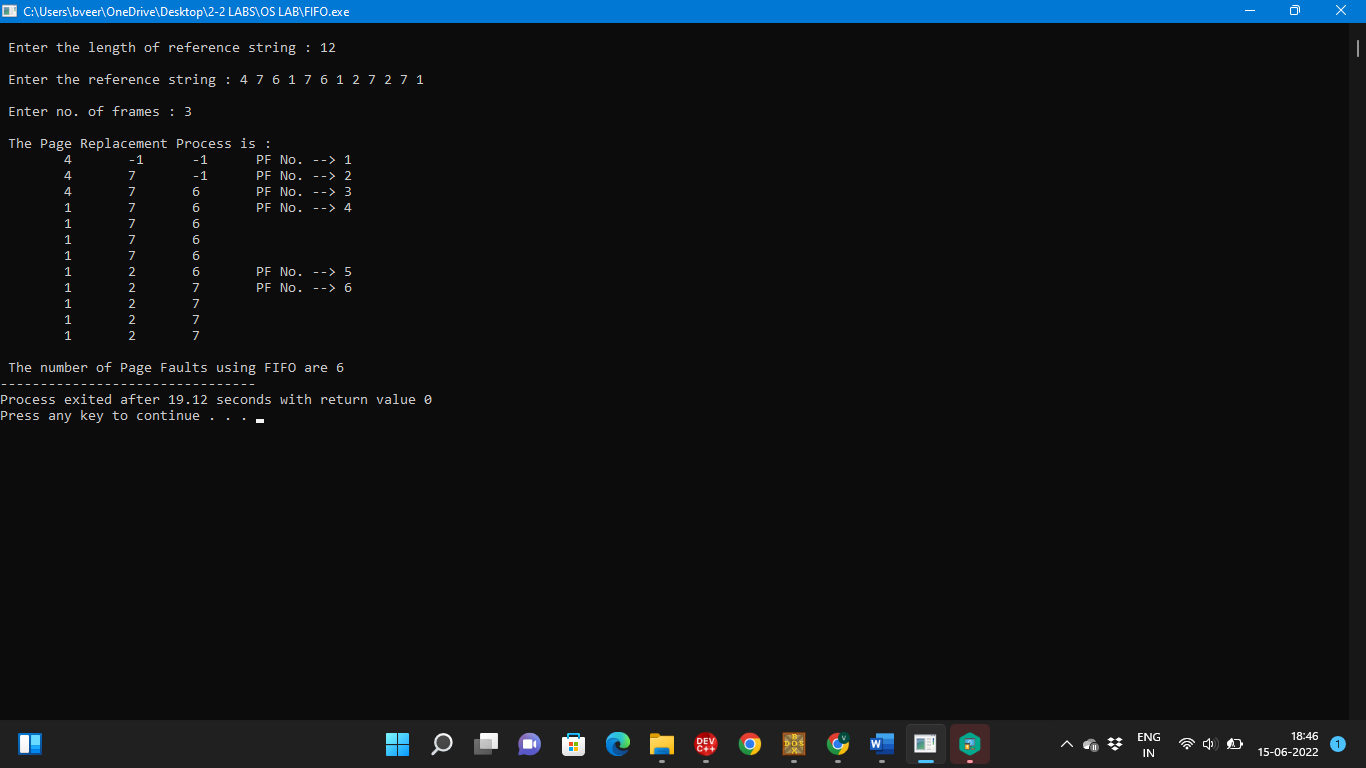
count=0;

}

printf("\n The number of Page Faults using FIFO are %d",pf);

}

OUTPUT



1. Write an assembly language program (ALP) to print numbers from 0 to9.

DATA SEGMENT

DATA ENDS

CODE SEGMENT

ASSUME CS:CODE, DS:DATA

BEGIN: MOV AX,DATA

MOV DS,AX

MOV CX,10

MOV DL,48

L1:MOV AH,2

INT 21H

INC DL

LOOP L1

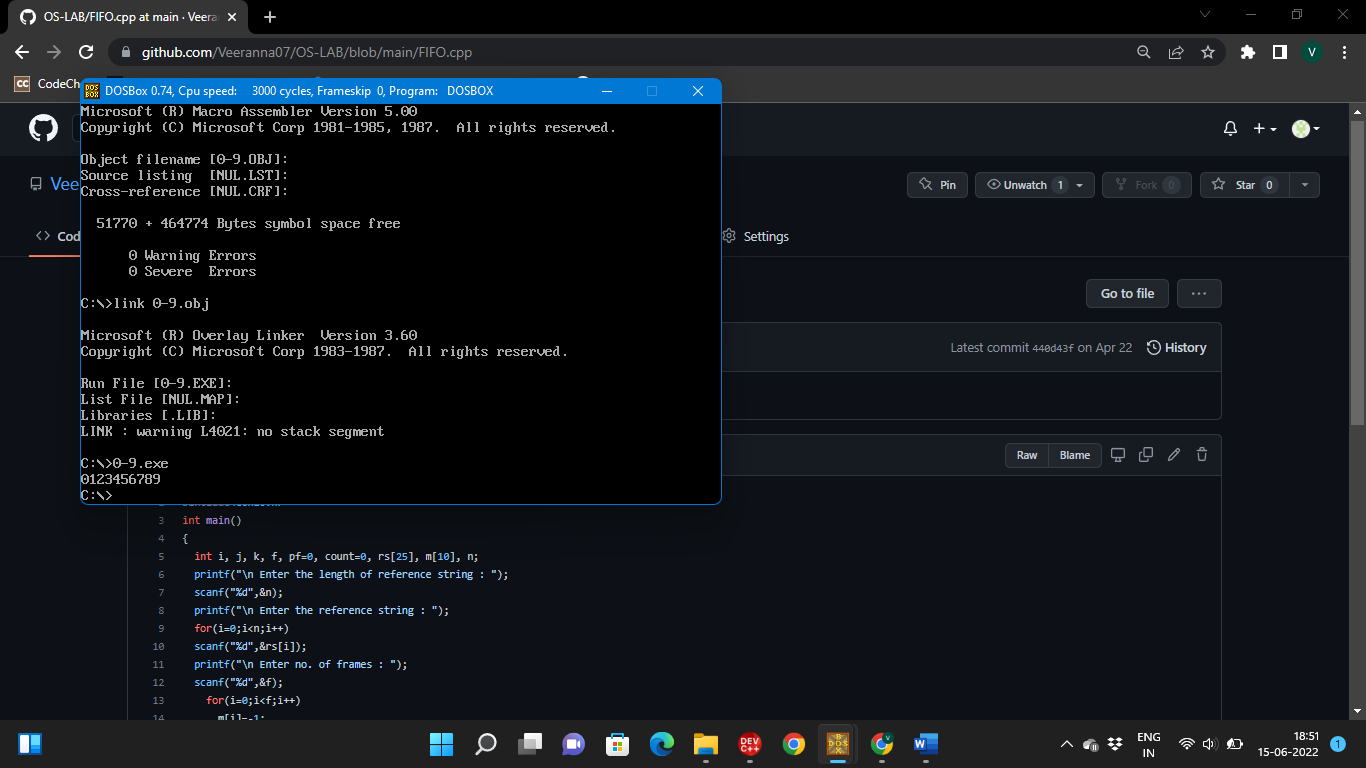
MOV AH,4CH

INT 21H

CODE ENDS

END BEGIN

**OUTPUT**

****

**SET-2**

1. Write a program to simulate the Optimal page replacement algorithm.

#include<stdio.h>

#include<conio.h>

int fr[3], n, m;

void display();

int main()

{

int i,j,page[20],fs[10];

int max,found=0,lg[3],index,k,l,flag1=0,flag2=0,pf=0;

float pr;

printf("\nEnter length of the reference string : ");

scanf("%d",&n);

printf("\nEnter the reference string : ");

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

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

printf("\nEnter no of frames : ");

scanf("%d",&m);

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

fr[i]=-1; pf=m;

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

{

flag1=0; flag2=0;

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

{

if(fr[i]==page[j])

{

flag1=1; flag2=1;

break;

}

}

if(flag1==0)

{

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

{

if(fr[i]==-1)

{

fr[i]=page[j]; flag2=1;

break;

}

}

}

if(flag2==0)

{

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

lg[i]=0;

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

{

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

{

if(fr[i]==page[k])

{

lg[i]=k-j;

break;

}

}

}

found=0;

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

{

if(lg[i]==0)

{

index=i;

found = 1;

break;

}

}

if(found==0)

{

max=lg[0]; index=0;

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

{

if(max<lg[i])

{

max=lg[i];

index=i;

}

}

}

fr[index]=page[j];

pf++;

}

display();

}

printf("\nNumber of page faults : %d\n", pf);

pr=(float)pf/n\*100;

printf("Page fault rate = %f \n", pr);

}

void display()

{

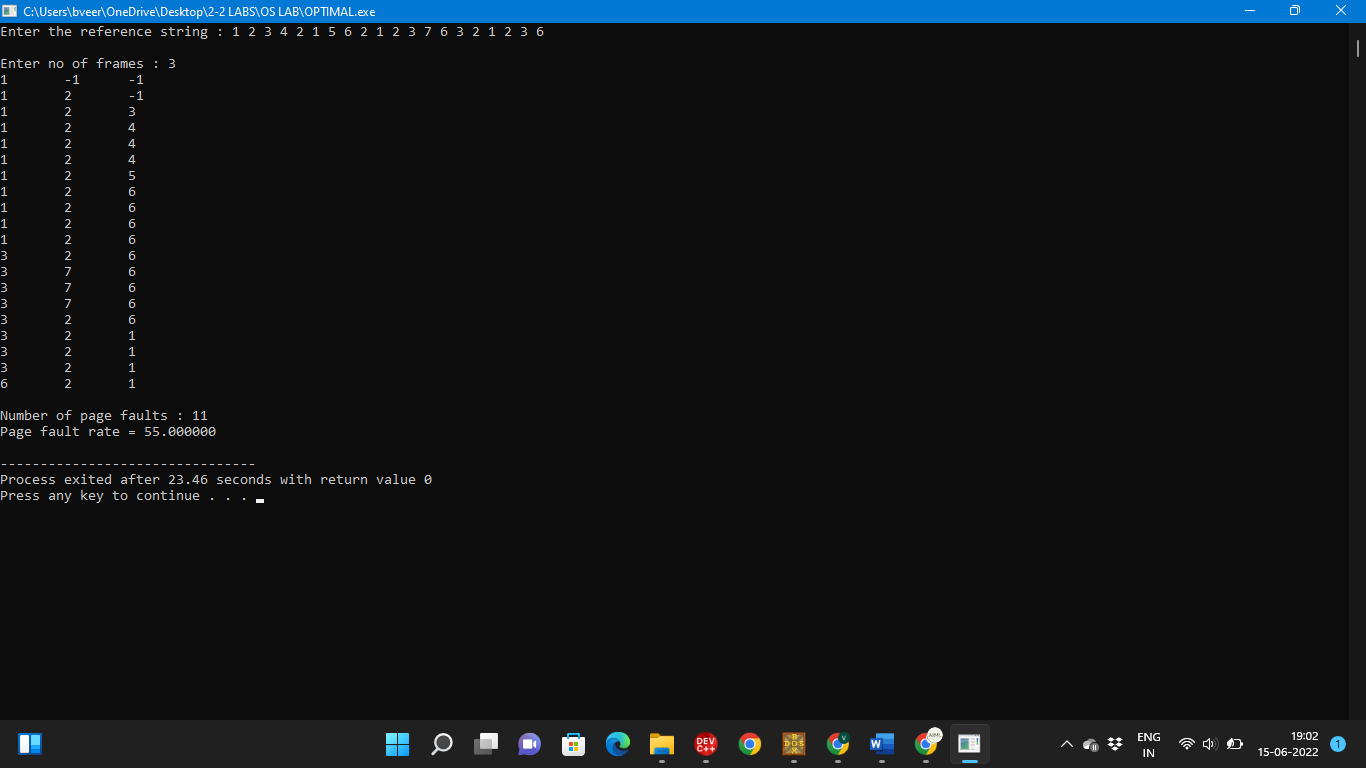
int i; for(i=0;i<m;i++)

printf("%d\t",fr[i]);

printf("\n");

}

**OUTPUT**

****

1. Write an Assembly Language Program (ALP) to check whether a given number is even or odd.

ASSUME CS:CODE,DS:DATA

DATA SEGMENT

NL1 DB 10,'ENTER NUMBER:$'

NL2 DB 10,'ODD$'

NL3 DB 10,'EVEN$'

SMLST DB ?

DATA ENDS

CODE SEGMENT

START:

MOV AX,DATA

MOV DS,AX

LEA DX,NL1

MOV AH,09H

INT 21H

MOV AH,01H

INT 21H

SUB AL,30H

TEST AX,01H

JE SKIP1

LEA DX,NL2

MOV AH,09H

INT 21H

JMP SKIP2

SKIP1:

LEA DX,NL3

MOV AH,09H

INT 21H

SKIP2:

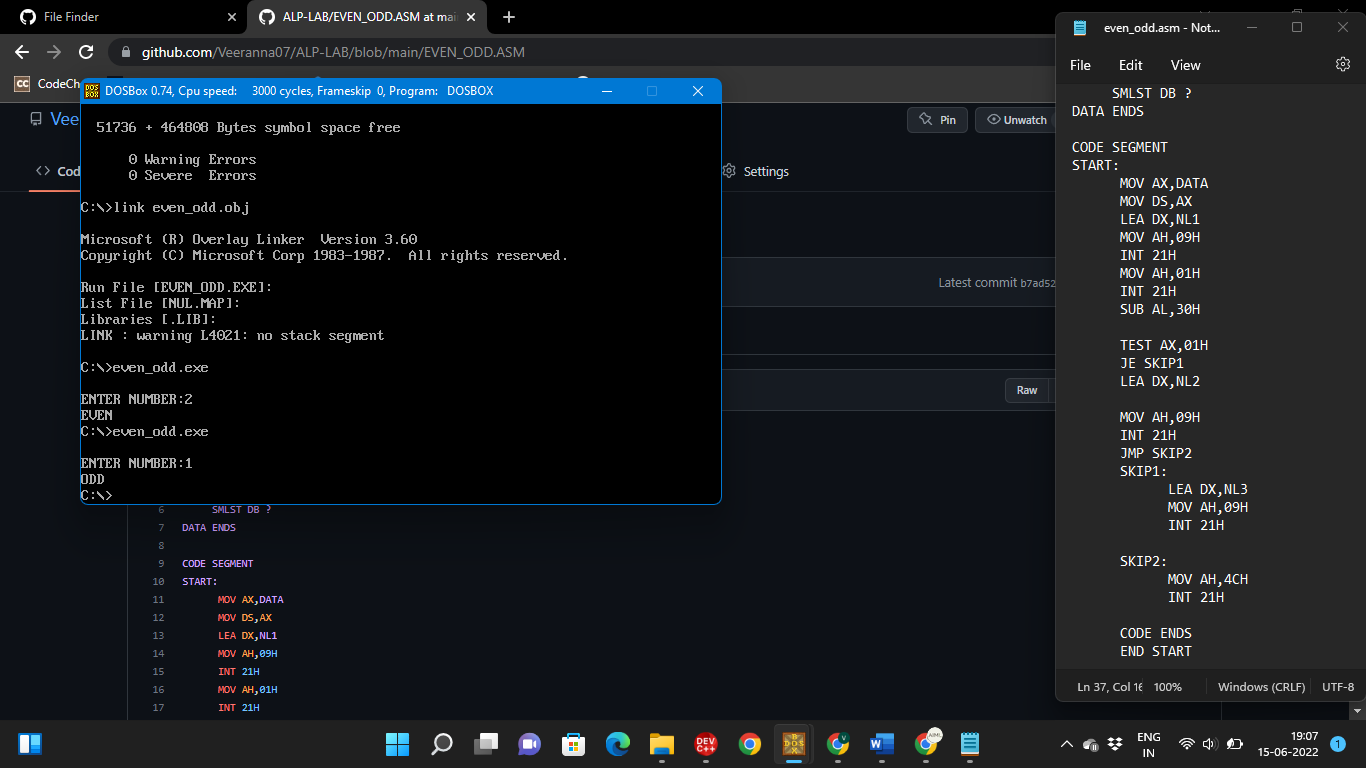
MOV AH,4CH

INT 21H

CODE ENDS

END START

**OUTPUT**



**SET-3**

1. Write a program to simulate the page replacement algorithm Least Recently Used (LRU)

#include<stdio.h>

#include<conio.h>

int main()

{

int i, j , k, min, rs[25], m[10], count[10], flag[25], n, f, pf=0, next=1;

printf("\nEnter the length of reference string : ");

scanf("%d",&n);

printf("\nEnter the reference string : ");

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

{

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

flag[i]=0;

}

printf("\nEnter the number of frames : ");

scanf("%d",&f);

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

{

count[i]=0;

m[i]=-1;

}

printf("\nThe Page Replacement process is \n");

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

{

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

{

if(m[j]==rs[i])

{

flag[i]=1;

count[j]=next;

next++;

}

}

if(flag[i]==0)

{

if(i<f)

{

m[i]=rs[i];

count[i]=next;

next++;

}

else

{

min=0;

for(j=1;j<f;j++)

if(count[min] > count[j]) min=j;

m[min]=rs[i];

count[min]=next;

next++;

}

pf++;

}

for(j=0;j<f;j++) printf("%d\t", m[j]);

if(flag[i]==0)

printf("PF No. --> %d" , pf);

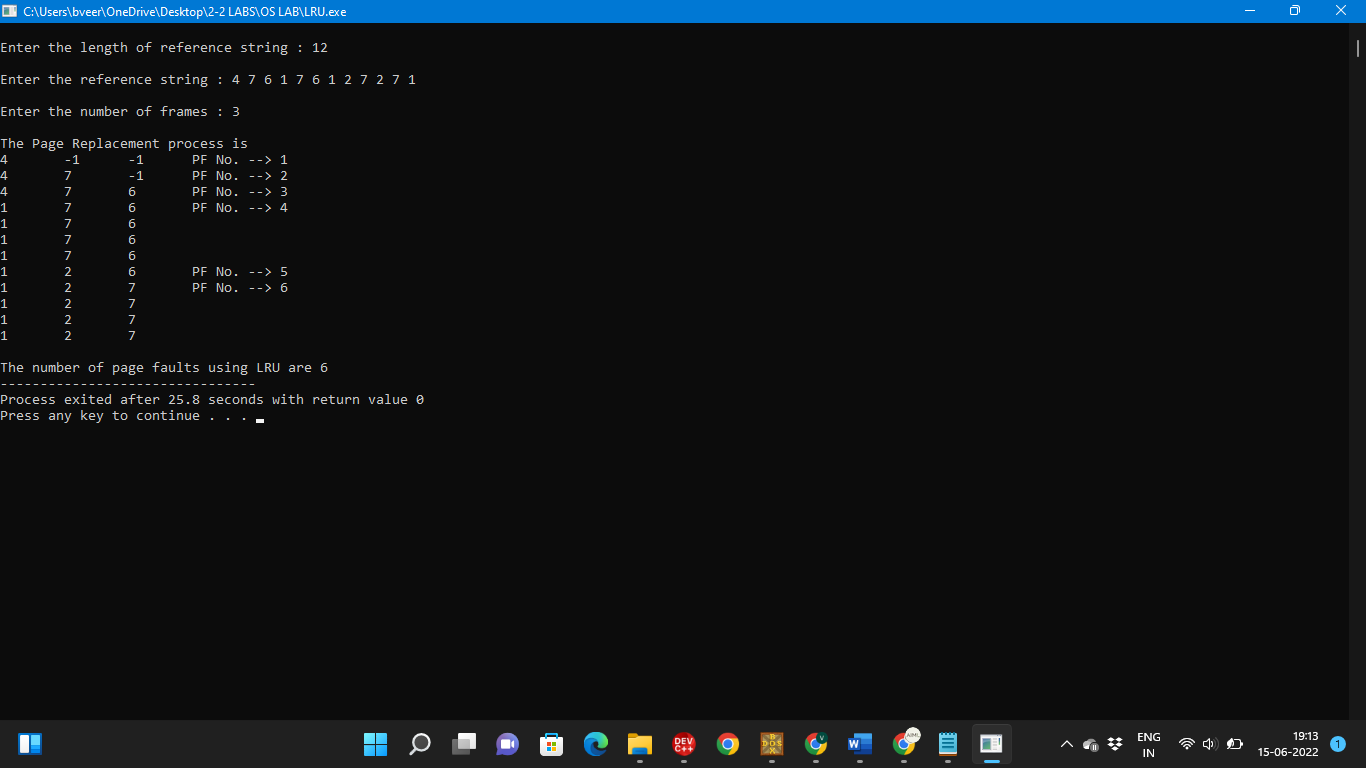
printf("\n");

}

printf("\nThe number of page faults using LRU are %d",pf);

}

OUTPUT



1. Write an Assembly Language Program (ALP) to find Factorial of a number.

ASSUME CS:CODE

CODE SEGMENT

START:

MOV CX,05H

MOV AX,01H

NEXT:

MUL CX

DEC CX

CMP CX,01H

JNZ NEXT

MOV AH,4CH

MOV BL,AL

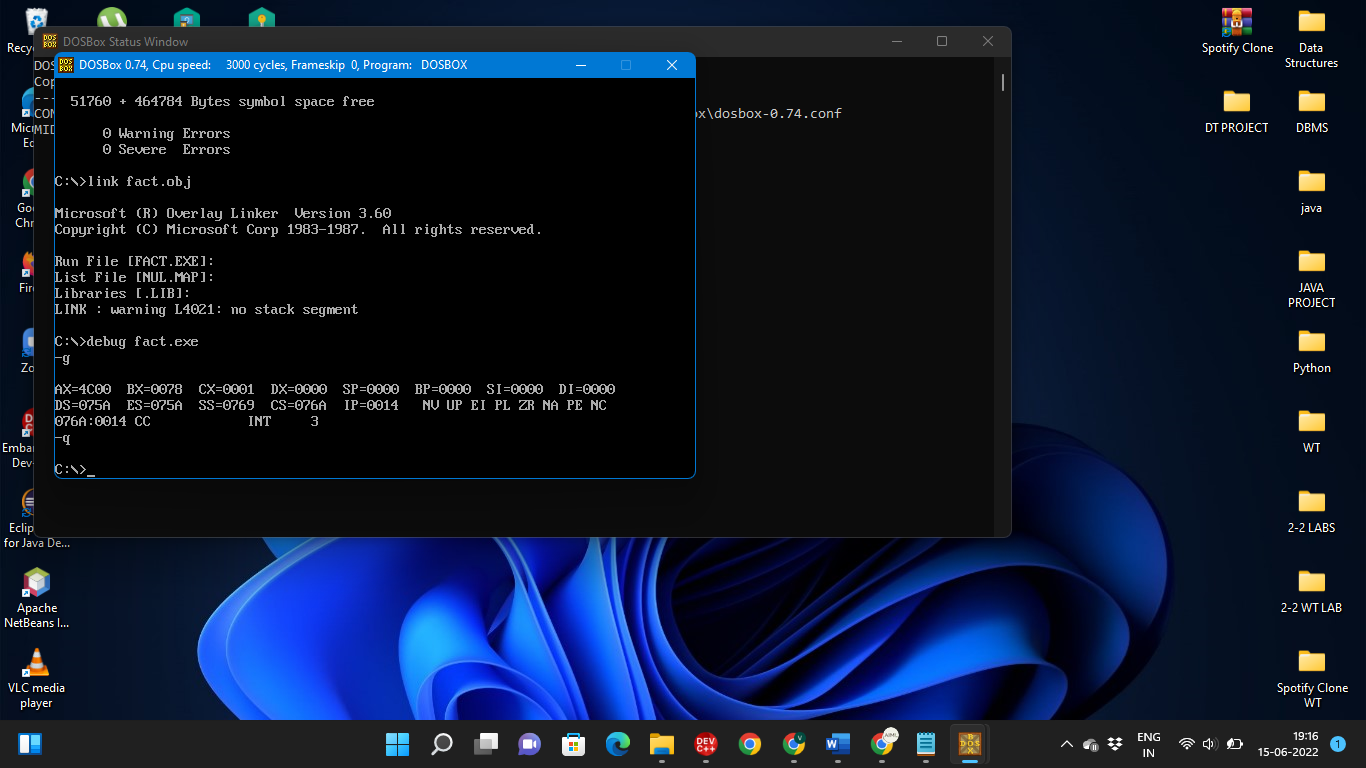
MOV AL,0H

INT 03H

CODE ENDS

END START

**OUTPUT**



**SET-4**

1. Write a program to simulate the File Organization Technique Single level directory.

#include<stdio.h>

#include<string.h>

int main()

{

int nf=0,i=0,j=0,ch;

char mdname[10],fname[10][10],name[10];

printf("Enter the directory name:");

scanf("%s",mdname);

printf("\nEnter the number of files:");

scanf("%d",&nf);

do

{

printf("Enter file name to be created:");

scanf("%s",name);

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

{

if(!strcmp(name,fname[i]))

break;

}

if(i==nf)

{

strcpy(fname[j++],name);

nf++;

}

else

printf("\nThere is already %s\n",name);

printf("\nDo you want to enter another file(yes - 1 or no - 0):");

scanf("%d",&ch);

}

while(ch==1);

printf("\nDirectory name is:%s\n",mdname);

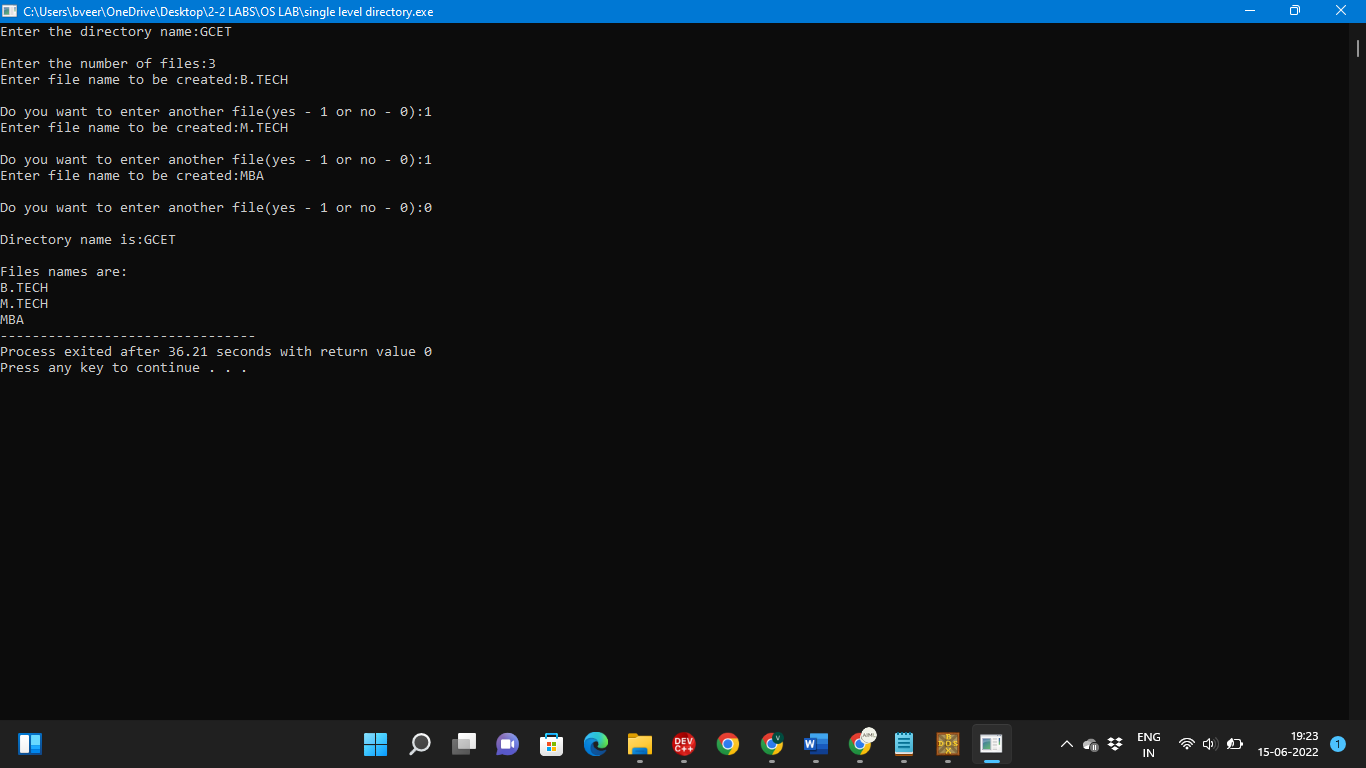
printf("\nFiles names are:");

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

printf("\n%s",fname[i]);

}

**OUTPUT**



1. Write an Assembly Language Program (ALP) to print Fibonacci series up to 5 numbers.

ASSUME CS:CODE,DS:DATA

DATA SEGMENT

NL1 DB 10,'ENTER NUMBER'

NL2 DB,'$'

TEMP DB ?

DATA ENDS

CODE SEGMENT

START:

MOV AX,DATA

MOV DS,AX

LEA DX,NL1

MOV AH,09H

INT 21H

MOV AH,01H

INT 21H

SUB AL,30H

MOV CL,AL

MOV AL,0

MOV BL,1

LBL1:

MOV TEMP,AL

LEA DX,NL2

MOV AH,09H

INT 21H

ADD AX,3030H

MOV DX,AX

MOV AH,02H

INT 21H

MOV AL,TEMP

ADD AL,BL

MOV BL,TEMP

LOOP LBL1

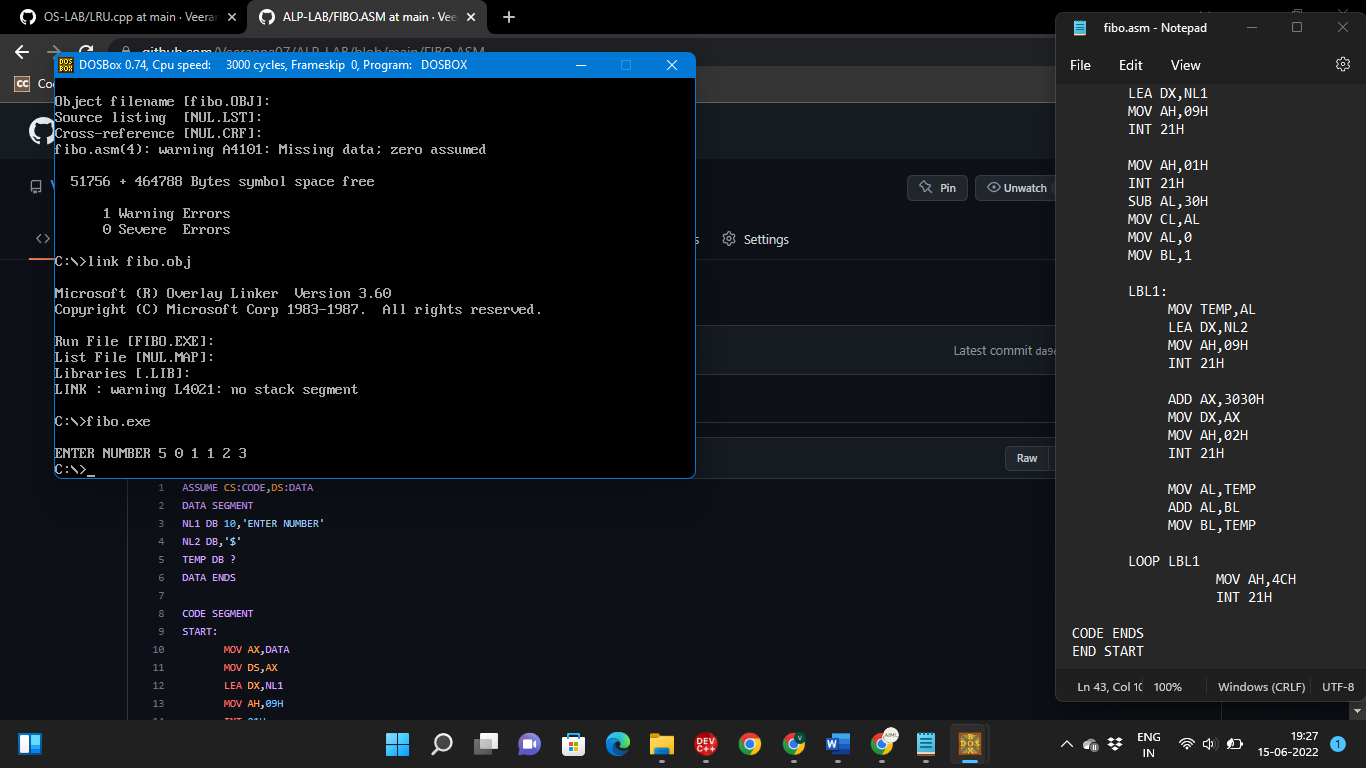
MOV AH,4CH

INT 21H

CODE ENDS

END START

**OUTPUT**



**SET-5**

1. Write a program to simulate the File Organization Technique Two level directory.

#include<stdio.h>

#include<conio.h>

struct st

{

char dname[10];

char sdname[10][10];

char fname[10][10][10];

int ds,sds[10];

}dir[10];

int main()

{

int i,j,k,n;

printf("\nEnter number of directories:");

scanf("%d",&n);

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

{

printf("\nEnter directory %d names:",i+1);

scanf("%s",&dir[i].dname);

printf("enter size of directories:");

scanf("%d",&dir[i].ds);

for(j=0;j<dir[i].ds;j++)

{

printf("\nEnter subdirectory name and size:");

scanf("%s",&dir[i].sdname[j]);

scanf("%d",&dir[i].sds[j]);

for(k=0;k<dir[i].sds[j];k++)

{

printf("\nEnter file name:");

scanf("%s",&dir[i].fname[j][k]);

}

}

}

printf("\ndirname\t\tsize\tsubdirname\tsize\tfiles");

printf("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

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

{

printf("%s\t\t%d",dir[i].dname,dir[i].ds);

for(j=0;j<dir[i].ds;j++)

{

printf("\t%s\t\t%d\t",dir[i].sdname[j],dir[i].sds[j]);

for(k=0;k<dir[i].sds[j];k++)

printf("%s\t",dir[i].fname[j][k]);

printf("\n\t\t");

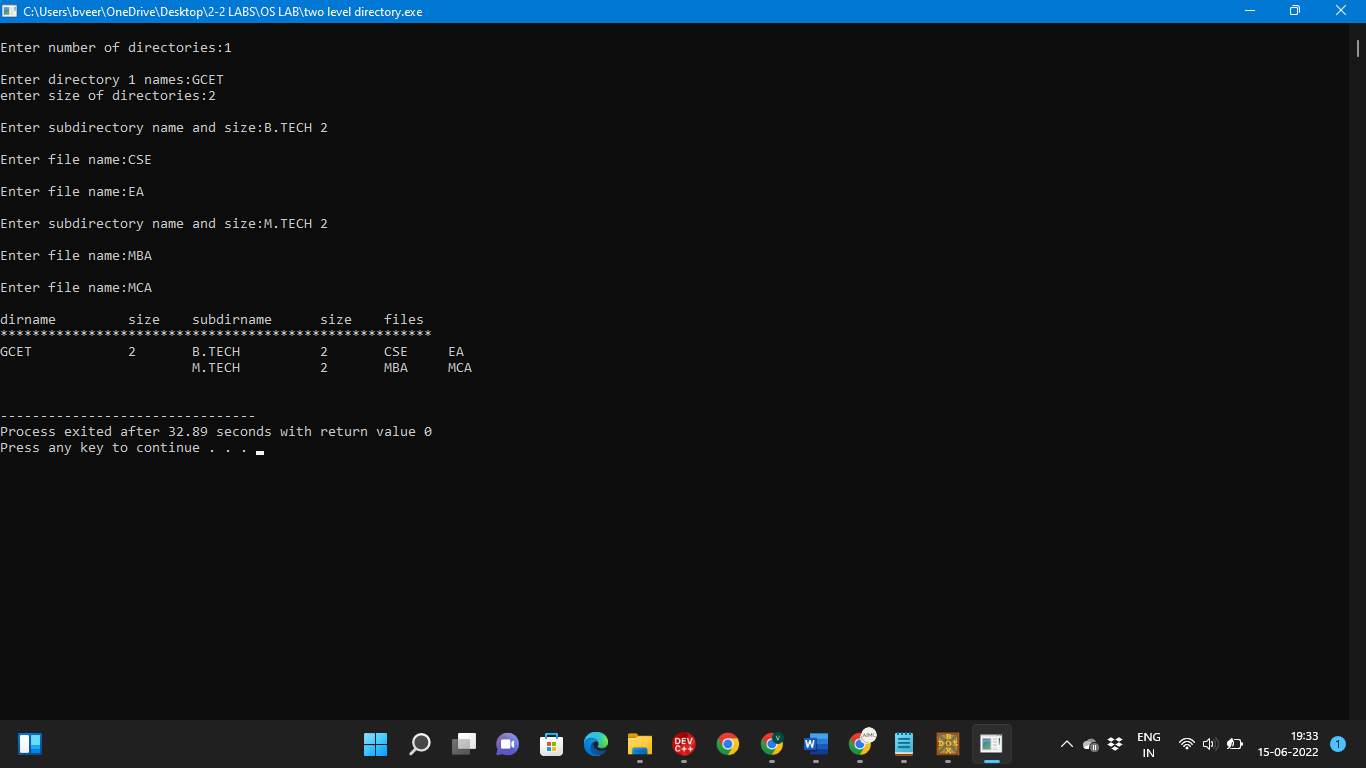
}

printf("\n");

}

}

**OUTPUT**



1. Write an Assembly Language Program (ALP) to take n values from user and calculate their sum for 8086.

ASSUME CS:CODE, DS:DATA

DATA SEGMENT

VAL1 DB ?

NL1 DB 0AH,0DH,'ENTER HOW MANY NO U WANT:','$'

NL2 DB 0AH,0DH,'ENTER NO:','$'

DATA ENDS

CODE SEGMENT

MAIN PROC

MOV AX,DATA

MOV DS,AX

LEA DX,NL1

MOV AH,09H

INT 21H

MOV AH,01H

INT 21H

SUB AL,30H

MOV CL,AL

MOV BL,AL

MOV AL,00

MOV VAL1,AL

LBL1:

LEA DX,NL2

MOV AH,09H

INT 21H

MOV AH,01H

INT 21H

SUB AL,30H

ADD AL,VAL1

MOV VAL1,AL

LOOP LBL1

MOV AX,00

MOV BL,VAL1

MOV AH,4CH

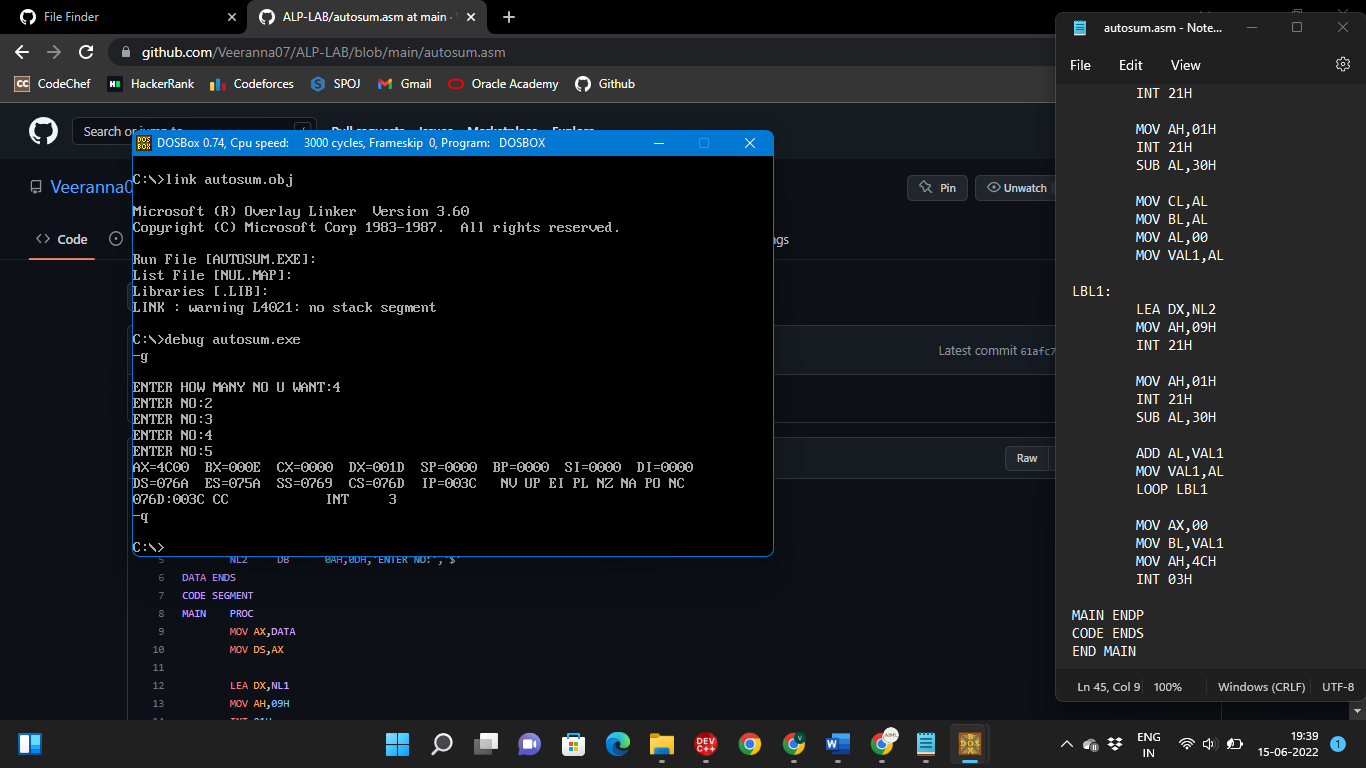
INT 03H

MAIN ENDP

CODE ENDS

END MAIN

**OUTPUT**



**SET-6**

1. Write a program to simulate the File Organization Technique Hierarchical directory.

#include<stdio.h>

struct st

{

char dname[10];

char sdname[10][10];

char fname[10][10][10];

int ds,sds;

}

dir[3];

main()

{ int i,j,k,n;

printf("enter number of directories : ");

scanf("%d",&n);

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

{

printf("enter directory %d name : ",i);

scanf("%s",&dir[i].dname);

printf("enter size of directory : ");

scanf("%d",&dir[i].ds);

for(j=0;j<dir[i].ds;j++)

{

printf("enter subdirectory name : ");

scanf("%s",dir[i].sdname[j]);

printf("enter subdirectory size : ");

scanf("%d",&dir[i].sds);

for(k=0;k<dir[i].sds;k++)

{

printf("enter filename : ");

scanf("%s",&dir[i].fname[j][k]);

} } }

printf("\n DIRNAME\t SIZE\t SUBDIRNAME \t SIZE\t FILES\n");

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

{

printf("%s\t\t%d",dir[i].dname,dir[i].ds);

for(j=0;j<dir[i].ds;j++)

{

printf("\t%s\t%d\t",dir[i].sdname[j],dir[i].sds);

for(k=0;k<dir[i].sds;k++)

printf("%s",dir[i].fname[j][k]);

printf("\n\t");

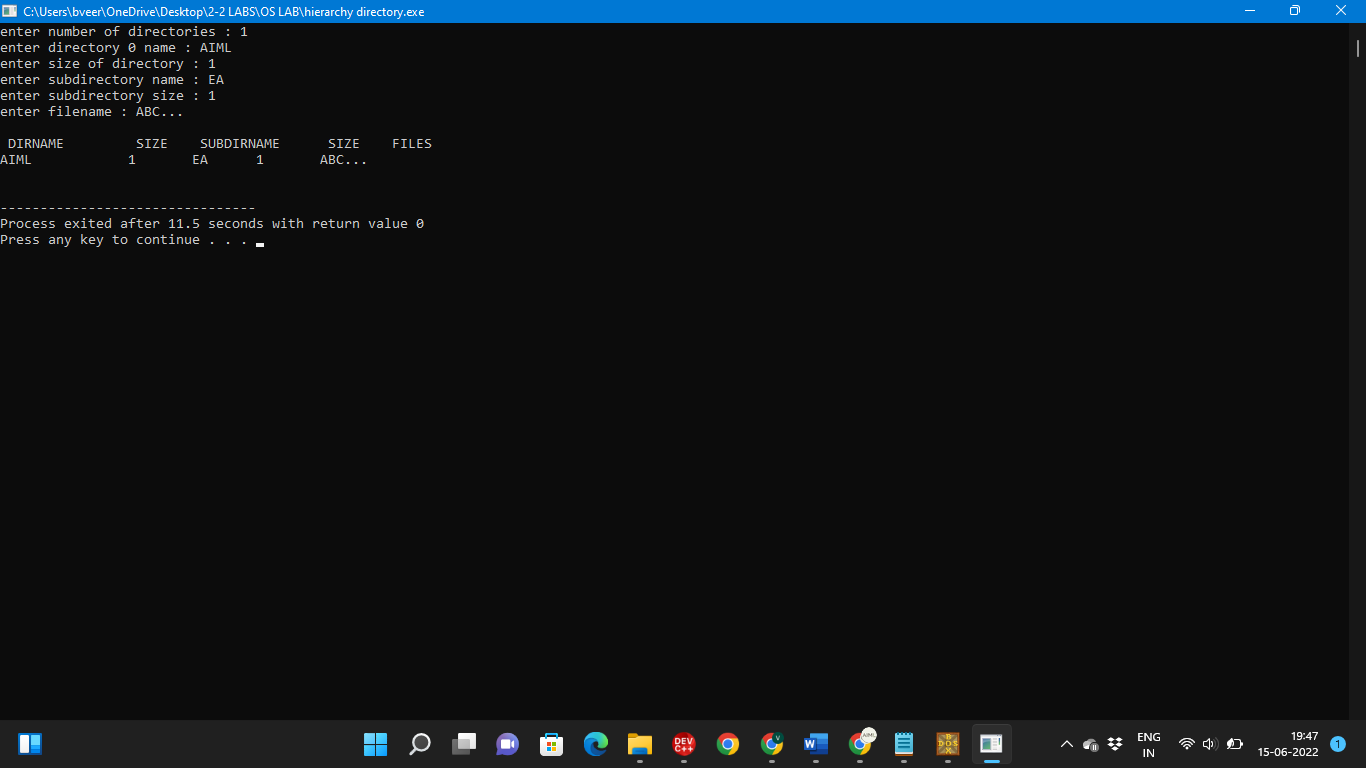
}

printf("\n");

}

}

**OUTPUT**



1. Write an Assembly Language Program(ALP) to take n values from user and calculate maximum and minimum values.

ASSUME DS:DATA, CS:CODE

DATA SEGMENT

ARR DB 5,3,7,1,9,2,6,8,4

LEN DW $-ARR

MIN DB ?

MAX DB ?

DATA ENDS

CODE SEGMENT

START:

MOV AX,DATA

MOV DS,AX

LEA SI,ARR

MOV AL,ARR[SI]

MOV MIN,AL

MOV MAX,AL

MOV CX,LEN

REPEAT:

MOV AL,ARR[SI]

CMP MIN,AL

JL CHECKMAX

MOV MIN,AL

CHECKMAX:

CMP MAX,AL

JG DONE

MOV MAX,AL

DONE:

INC SI

LOOP REPEAT

MOV BL,MIN

MOV CL,MAX

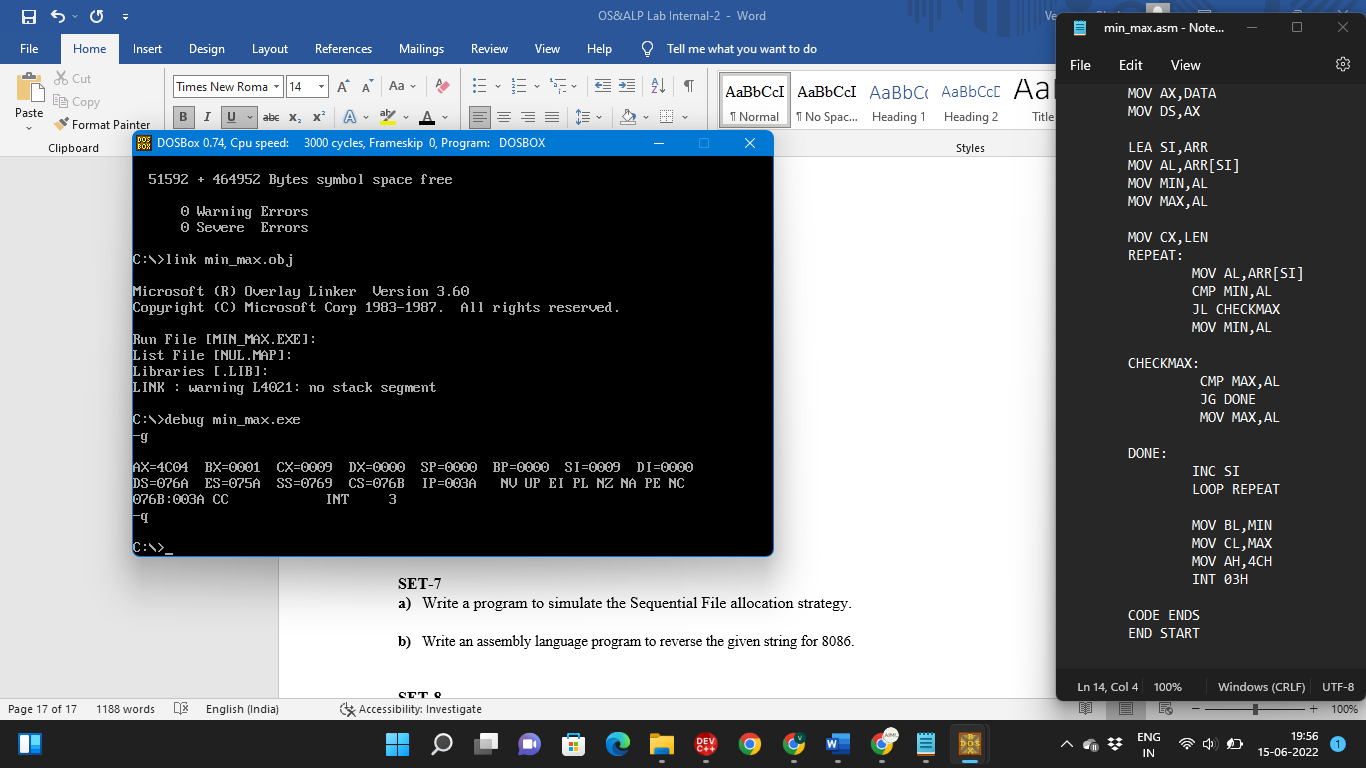
MOV AH,4CH

INT 03H

CODE ENDS

END START

**OUTPUT**



**SET-7**

1. Write a program to simulate the Sequential File allocation strategy.

#include<stdio.h>

#include<conio.h>

int main()

{

int f[50], i, st, len, j, c, k, count = 0;

for(i=0;i<50;i++)

f[i]=0;

x: count=0;

printf("\nEnter starting block and length of files: ");

scanf("%d%d", &st,&len);

for(k=st;k<(st+len);k++)

if(f[k]==0)

count++;

if(len==count)

{

for(j=st;j<(st+len);j++)

if(f[j]==0)

{

f[j]=1;

printf("%d--> %d\n",j,f[j]);

}

if(j!=(st+len-1))

printf("\n\tThe file is allocated to disk\n");

}

else

printf("\n\tThe file is not allocated \n");

printf("\nDo you want to enter more file(Yes - 1/No - 0)");

scanf("%d", &c);

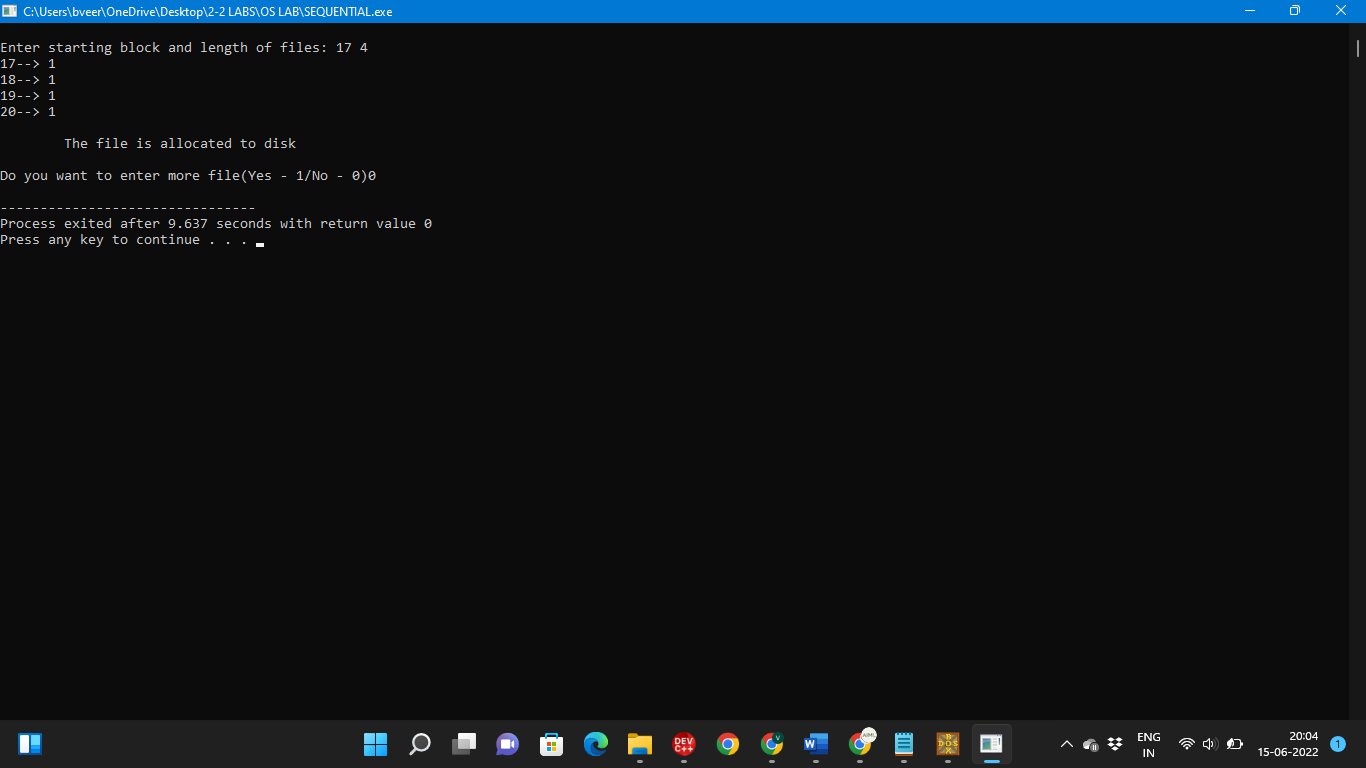
if(c==1)

goto x;

getch();

}

**OUTPUT**



1. Write an assembly language program to reverse the given string for 8086.

DATA SEGMENT

OSTR DB 'COMPUTER','$'

SLEN DW $-OSTR

RSTR DB 20 DUP('COMPUTER')

DATA ENDS

CODE SEGMENT

ASSUME CS:CODE,DS:DATA

BEGIN:

MOV AX,DATA

MOV DS,AX

MOV ES,AX

MOV CX,SLEN

ADD CX,-2

LEA SI,OSTR

LEA DI,RSTR

ADD SI,SLEN

ADD SI,-2

L1:

MOV AL,[SI]

MOV [DI],AL

DEC SI

INC DI

LOOP L1

MOV AL,[SI]

MOV [DI],AL

INC DI

MOV DL,'$'

MOV [DI],DL

PRINT:

MOV AH,09H

LEA DX,RSTR

INT 21H

EXIT:

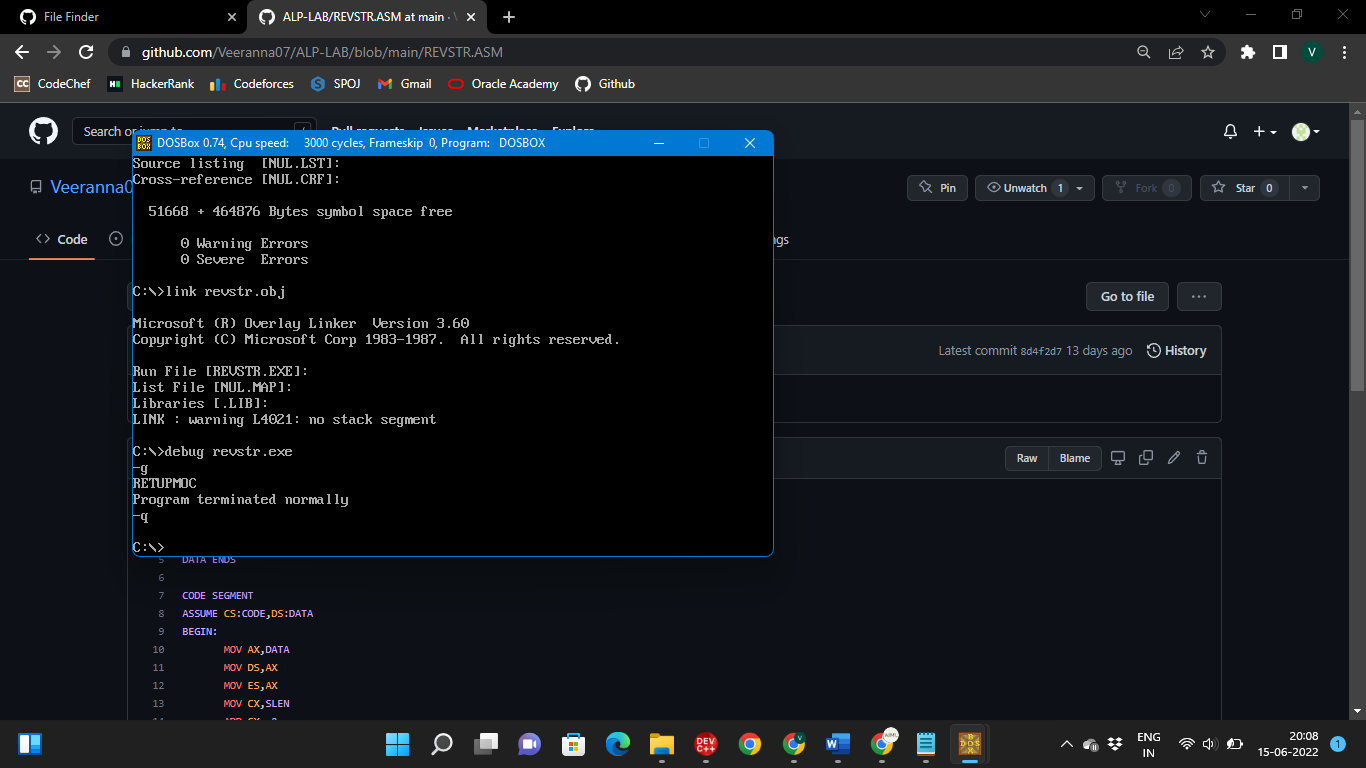
MOV AX,4C00H

INT 21H

CODE ENDS

END BEGIN

**OUTPUT**



**SET-8**

1. Write a program to simulate the indexed File allocation strategy.

#include<stdio.h>

#include<conio.h>

#include<stdlib.h>

int main()

{

int f[50], index[50],i, n, st, len, j, c, k, ind,count=0;

for(i=0;i<50;i++)

f[i]=0;

x:printf("\nEnter the index block: ");

scanf("%d",&ind);

if(f[ind]!=1)

{

printf("\nEnter no of blocks needed and no of files for the index %d on the disk : \n", ind);

scanf("%d",&n);

}

else

{

printf("\t%d index is already allocated \n",ind);

goto x;

}

y: count=0;

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

{

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

if(f[index[i]]==0)

count++;

}

if(count==n)

{

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

f[index[j]]=1;

printf("\tAllocated\n");

printf("File Indexed\n");

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

printf("%d ------->%d : %d\n",ind,index[k],f[index[k]]);

}

else

{

printf("\tFile in the index is already allocated \n");

printf("\nEnter another file indexed : ");

goto y;

}

printf("\nDo you want to enter more file(Yes - 1/No - 0)");

scanf("%d", &c);

if(c==1)

goto x;

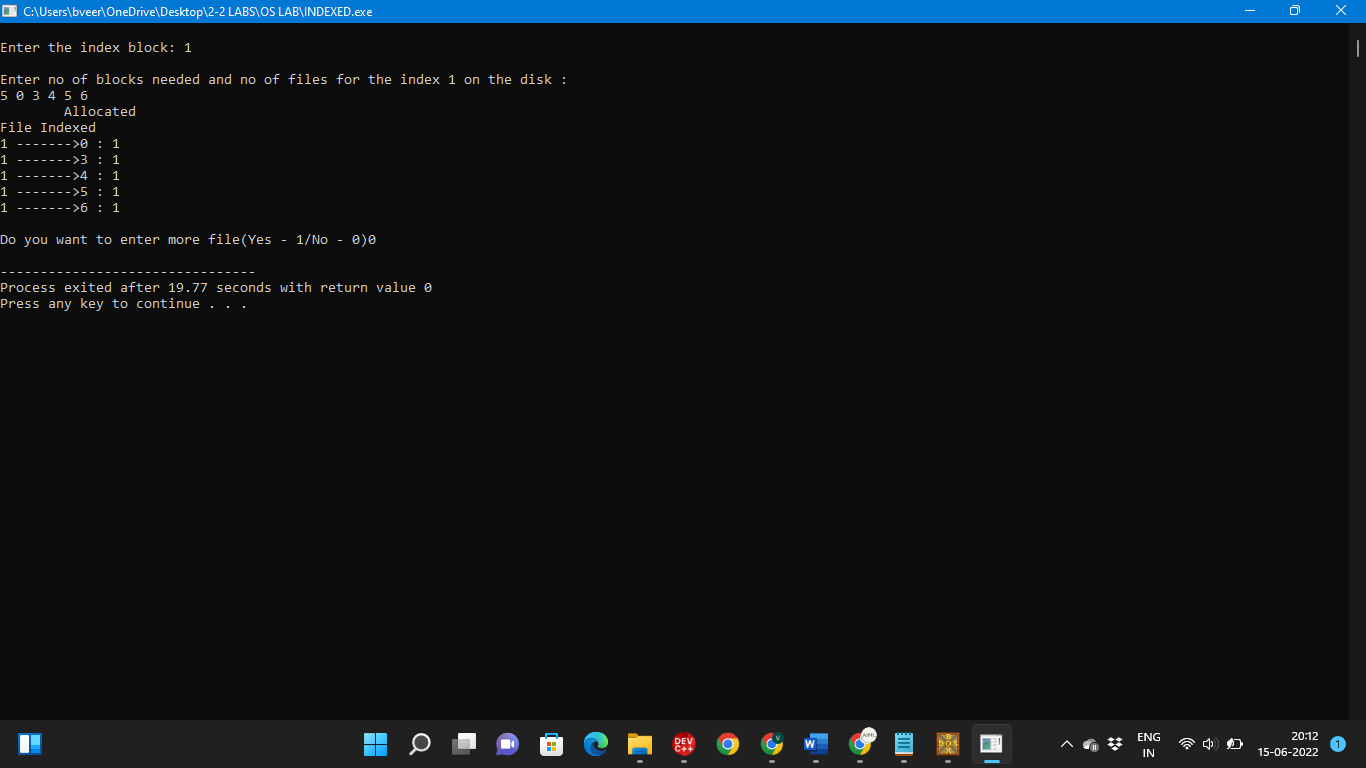
else

exit(0);

getch();

}

**OUTPUT**



1. Write 8086 assembly language program to transfer a block of data from one location to another.

ORG 100H

ASSUME CS:CODE

CODE SEGMENT

START:

MOV CX,08H

MOV BX,10H

MOV DX,21H

MOV SI,BX

MOV DI,DX

MOV AL,CL

STORE:

MOV BYTE PTR[SI],AL

DEC AL

CMP AL,0H

JNE STORE

AGAIN:

MOV AL,BYTE PTR[SI]

MOV BYTE PTR[DI],AL

LOOP AGAIN

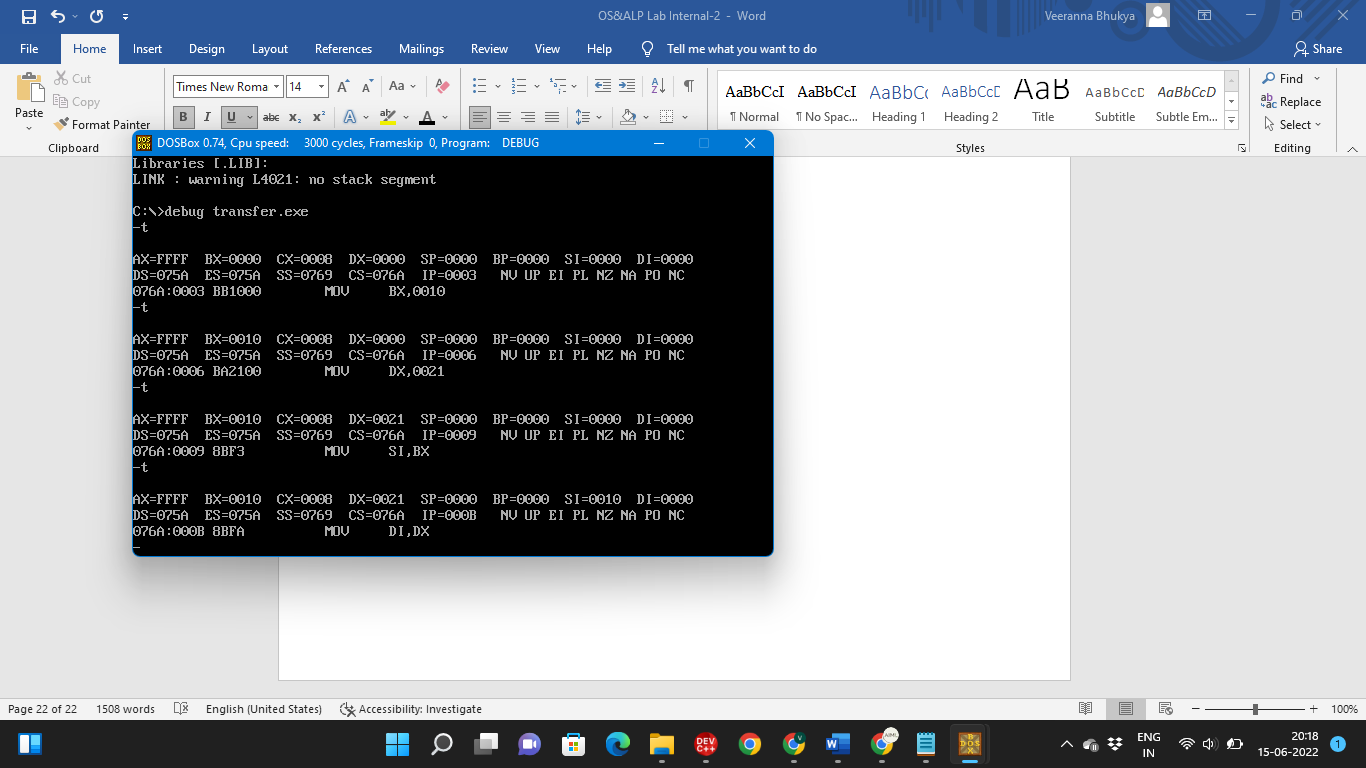
MOV AH,4CH

INT 21H

CODE ENDS

END START

**OUTPUT**



**SET-9**

1. Write a program to simulate the Linked List File allocation strategy.

#include<stdio.h>

#include<conio.h>

#include<stdlib.h>

int main()

{

int f[50], p,i, st, len, j, c, k, a;

for(i=0;i<50;i++)

f[i]=0;

printf("\nEnter how many blocks already allocated: ");

scanf("%d",&p);

printf("\nEnter blocks already allocated: ");

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

{

scanf("%d",&a);

f[a]=1;

}

x: printf("\nEnter index starting block and length: ");

scanf("%d%d", &st,&len);

k=len;

if(f[st]==0)

{

for(j=st;j<(st+k);j++)

{

if(f[j]==0)

{

f[j]=1;

printf("\t%d ------->%d\n",j,f[j]);

}

else

{

printf("\t%d Block is already allocated \n",j);

k++;

}

}

}

else

printf("\n%d starting block is already allocated \n",st);

printf("\nDo you want to enter more file(Yes - 1/No - 0)");

scanf("%d", &c);

if(c==1)

goto x;

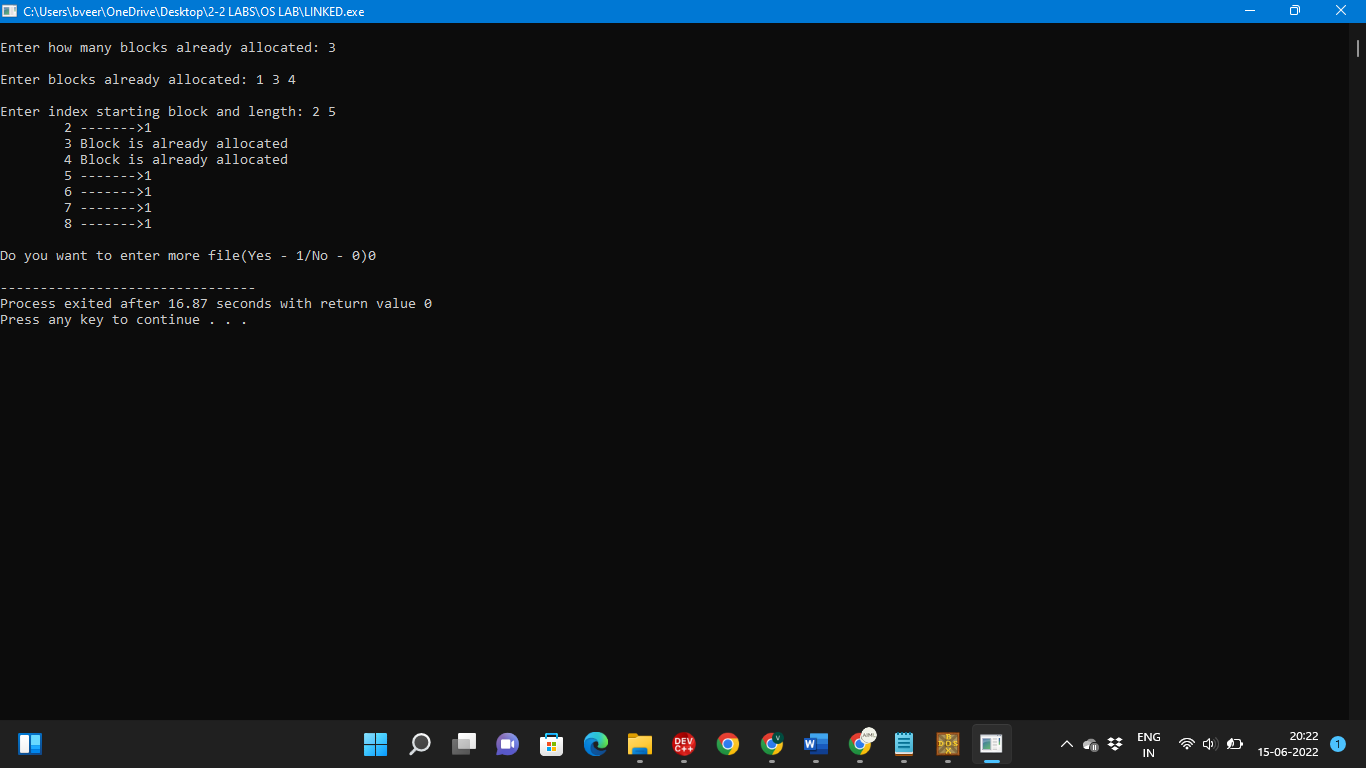
else

exit(0);

getch();

}

**OUTPUT**



1. Write an Assembly Language Program (ALP) to find Factorial of a number.

ASSUME CS:CODE

CODE SEGMENT

START:

MOV CX,05H

MOV AX,01H

NEXT:

MUL CX

DEC CX

CMP CX,01H

JNZ NEXT

MOV AH,4CH

MOV BL,AL

MOV AL,0H

INT 03H

CODE ENDS

END START

**OUTPUT**

