# ADVANCED MATHS PROCEDURES



Abdul Basit : 20K-0333

# “Computer Organizaion And Assembly Language”

National Universiy Of Computer And

Emerging Sciences – Fast

INTRODUCTION :

We have designed advanced maths procedures in our project. As Mathematics is the most important area of Computer Sciences and technology and also play a vital role in building logics etc

LITERATURE REVIEW :

For the implementation of our project we have reviewed many research links on internet also studied book regarding this and after a deep analysis and close study ,we are able to achieve our goal.

PROBLEM DEFINITION :

In Assembly Language, pupils usually deal with integer values and arithmetic operations on it but in our project along with integer data , we also made use of float data and the working of our procedures were mainly based on it.

METHODOLOGY :

We have made use of FPU registers along with general purpose registers in the implementation of our project as they can easily handle floating poing values and arithmetic operations on it by providing a lot of set of instructions.

DETAILED DESIGN AND ARCHITECTURE :

In our project , there are two main sections of mathematics implemented .

1. Coordinate Geometry.
2. Vector Algebra.

In the main procedure there were only two procedures called ,one is for showing slight animation as intro and other is used to show menu for the user to operate the system as per desire.

Furthermore ,in menu procedure there were two choices for the users shown as mentioned above which will switch the control over the desired section .

We have implemented 16 functions .10 for Coordinate Geometry and 6 for Vector Algebra.

IMPLEMENTATION TESTING AND PROGRAMMING CODE :

• Coordinate Geometry :

1. Quadratic\_Equation

It will calculate the roots of the quadratic equation given the values of a , b and c and also showing nature of root.

1. Point Of Intersection :

It will show coordinates of point of intersection of two lines and will also handle the case of inconsistency.

1. Distance :

Showing distance b/w two points in the rectangular cartesian coordinate system .

1. Show\_Line\_Detail :

Displaying x-intercept , y-intercept ,slope and angle of line from x-axis.

1. Angle\_B/W\_Lines :

Giving angle between two intersecting lines in degrees.

1. Trigonometric\_Ratios :

Calculating all six trigonometric ratios upon entering input of angle in degrees.

1. Area\_Of\_Triangle :

Finding area of triangle upon input of three sides of triangles .

1. Centroid\_Of\_Triangle :

Giving coordinates of centroid of triangle ,taking coordinates of vertices of triangle.

1. Incentre\_Of\_Triangle :

Finding coordinates of incentre of triangle,by taking coordinates of vertices of triangle.

1. Circum\_Radius\_Of\_Triangle :

Evaluating circum radius of triangle upon input of sides of triangle.

• Vector Algebra :

1. Magnitude\_Of\_A\_Vector

Calculating magnitude of a vector given its 3 coordinates.

1. Unit\_Vector :

Finding Unit vector of a given vector.

1. Angle\_BW\_Vectors:

Calculating angle b/w two vectors.

1. Dot\_Product :

Displaying Dot Product of two given vectors .

1. Vector Product / Perpendicual Vector :

Finding vector product using logic of determinant .

1. Linear Combination :

Express third vector in terms of first and second vector for only unique solution.

CODE:

include irvine32.inc include macros.inc

.data

heading1 BYTE ">>>>>>>>>>>>>> COAL PROJECT

<<<<<<<<<<<<<<",0

heading2 BYTE "TITLE: ( ADVANCED MATHS PROCEDURES )" .code main proc

call clrscr

mov eax,black+(11 \*16) call setTextColor call clrscr call ProjectIntro call MainMenu

call clrscr exit

main endp

ProjectIntro PROC

LOCAL vp:BYTE

LOCAL save:DWORD

LOCAL i :BYTE

LOCAL j :BYTE LOCAL k :BYTE mov eax,0

mov edx,0 mov ecx,37 mov ebx,0 mov vp,0 mov i,0 mov j,0 mov k,0 mov si,0 LB1: mov dh,10 mov dl,39 sub dl,vp call gotoxy mov al,22 call writechar mov dl,39 add dl,vp call gotoxy call writechar inc vp

mov eax,1 call delay Loop LB1 mov ecx,9 LB2: mov save,ecx mov ecx,36 LB3: mov dl,39 sub dl,j

mov dh,9 sub dh,i call gotoxy mov al,24 call writechar mov dl,39 add dl,j mov dh,9 sub dh,i call gotoxy call writechar mov dl,39 sub dl,j

mov dh,9 add dh,i call gotoxy mov al,25 call writechar mov dl,39 add dl,j mov dh,9 add dh,i call gotoxy call writechar inc j

mov eax,1 call delay Loop LB3 mov ecx,36

L4:

mov dl,39 sub dl,k mov dh,9 sub dh,i call gotoxy mwrite<" "> mov dl,39 add dl,k mov dh,9 sub dh,i call gotoxy mwrite<" "> mov dl,39 sub dl,k mov dh,9 add dh,i call gotoxy mwrite<" "> mov dl,39 add dl,k mov dh,9 add dh,i call gotoxy mwrite<" "> inc k Loop L4 mov ecx,save

inc i

dec ecx jnz LB2 mov k,0 mov ecx,38

L5: mov dl,39 sub dl,k mov dh,1 call gotoxy mov al,22 call writechar mov dl,39 add dl,k mov dh,1 call gotoxy call writechar mov dl,39 sub dl,k mov dh,17 call gotoxy call writechar mov dl,39 add dl,k mov dh,17 call gotoxy call writechar inc k dec ecx jnz L5 mov k,0 mov ecx,9 L6:

mov dl,39 mov dh,10 add dh,k dec dh call writechar inc k Loop L6 mov al,dl mov ah,dh mov dl,15 mov dh,7 call gotoxy mov edx,OFFSET heading1 call writestring

mwrite<0ah,0ah,0ah,0ah,0ah,0ah,0ah," "> mov edx,OFFSET heading2 call writestring mov dl,al mov dh,ah call gotoxy mwrite<0ah,0ah,0ah,0ah,0ah,0ah,0ah," "> call waitmsg ret

ProjectIntro Endp

MainMenu PROC STMenu: call clrscr mov dl,0 mov dh,0 call gotoxy

mwrite<0ah,0ah,0ah,0ah,0ah,0ah," !!!!!!!!!!! PROJECT SECTIONS !!!!!!!!!!!!:",0ah,0ah,0ah,0ah>

mwrite<" (1) ( Coordinate Geometry ) :",0ah,0ah,0ah> mwrite<" (2) ( Vector Algebra ) : ",0ah,0ah,0ah,0ah,0ah,0ah> mwrite<" Enter your choice : "> call readdec cmp eax,1 JE CG cmp eax,2

JE VL

jmp MNQ

CG: call CoordinateGeometry call waitmsg jmp STMenu

VL: call VectorAlgebra call waitmsg jmp STMenu MNQ:

ret

MainMenu Endp CoordinateGeometry PROC call clrscr mov dl,0 mov dh,0 call gotoxy mwrite<0ah,0ah,0ah>

mwrite<" >>>>>>>>>>>> COORDINATE GEOMETRY

>>>>>>>>>>>>",0ah,0ah,0ah>

mwrite<" (1) Quadratic\_Equation ",0ah,0ah> mwrite<" (2) Point Of Intersection ",0ah,0ah> mwrite<" (3) Distance ",0ah,0ah> mwrite<" (4) Show\_Line\_Detail",0ah,0ah> mwrite<" (5) Angle\_B/W\_Lines ",0ah,0ah> mwrite<" (6) Trigonometric\_Ratios ",0ah,0ah> mwrite<" (7) Area\_Of\_Triangle ",0ah,0ah> mwrite<" (8) Centroid\_Of\_Triangle ",0ah,0ah> mwrite<" (9) Incentre\_Of\_Triangle",0ah,0ah> mwrite<" (10) Circum\_Radius\_Of\_Triangle",0ah,0ah,0ah> mwrite<" Enter Your Choice : "> call readdec cmp eax,1 JE CG1 cmp eax,2 JE CG2 cmp eax,3 JE CG3 cmp eax,4 JE CG4

cmp eax,5 JE CG5 cmp eax,6 JE CG6 cmp eax,7 JE CG7 cmp eax,8 JE CG8 cmp eax,9 JE CG9 cmp eax,10 JE CG10 jmp CGQ CG1:

call clrscr mov dl,0 mov dh,0 call gotoxy call QuadraticEquation jmp CGQ

CG2:

call clrscr mov dl,0 mov dh,0 call gotoxy call pointOfIntersection jmp CGQ call clrscr mov dl,0 mov dh,0 call gotoxy CG3:

call clrscr mov dl,0 mov dh,0 call gotoxy call Distance jmp CGQ CG4:

call clrscr

mov dl,0 mov dh,0 call gotoxy call ShowLineDetail jmp CGQ CG5:

call clrscr mov dl,0 mov dh,0 call gotoxy call AngleBWLines jmp CGQ CG6:

call clrscr mov dl,0 mov dh,0 call gotoxy call trigonometricRatios jmp CGQ CG7:

call clrscr mov dl,0 mov dh,0 call gotoxy call AreaOfTriangle jmp CGQ CG8:

call clrscr mov dl,0 mov dh,0 call gotoxy call CentroidOfTriangle jmp CGQ CG9:

call clrscr mov dl,0 mov dh,0 call gotoxy call IncentreOfTriangle jmp CGQ

CG10:

call clrscr mov dl,0 mov dh,0 call gotoxy call CircumRadius CGQ:

ret

CoordinateGeometry Endp VectorAlgebra PROC

call clrscr mov dl,0 mov dh,0 call gotoxy mwrite<0ah,0ah,0ah>

mwrite<" >>>>>>>>>>>> VECTOR ALGEBRA

>>>>>>>>>>>>",0ah,0ah,0ah>

mwrite<" (1) Magnitude\_Of\_A\_Vector ",0ah,0ah> mwrite<" (2) Unit\_Vector ",0ah,0ah> mwrite<" (3) Angle\_BW\_Vectors ",0ah,0ah> mwrite<" (4) Dot\_Product",0ah,0ah> mwrite<" (5) Vector\_Product ",0ah,0ah>

mwrite<" (6)

Linear\_Combination\_Of\_Vectors",0ah,0ah,0ah> mwrite<" Enter Your Choice : "> call readdec cmp eax,1 JE VL1 cmp eax,2 JE VL2 cmp eax,3 JE VL3 cmp eax,4 JE VL4 cmp eax,5 JE VL5 cmp eax,6 JE VL6 jmp VLQ VL1:

call clrscr mov dl,0 mov dh,0 call gotoxy call Magnitude jmp VLQ VL2: call clrscr mov dl,0 mov dh,0 call gotoxy call UnitVector jmp VLQ VL3: call clrscr mov dl,0 mov dh,0 call gotoxy call Angle\_BW\_Vectors jmp VLQ

VL4: call clrscr mov dl,0 mov dh,0 call gotoxy call DotProduct jmp VLQ VL5: call clrscr mov dl,0 mov dh,0 call gotoxy call VectorProduct jmp VLQ VL6: call clrscr mov dl,0 mov dh,0 call gotoxy call LinearCombination VLQ: ret

VectorAlgebra Endp

QuadraticEquation PROC

LOCAL a:real4

LOCAL b:real4

LOCAL cz:real4

LOCAL const:DWORD mov eax,4 mov const,eax

finit

mwrite<0ah," ( Equation Of Form ax^2 + bx + c = 0

)",0ah,0ah>

mwrite<" Enter value of a : "> call readfloat fstp a

mwrite<" Enter value of b : "> call readfloat fstp b

mwrite<" Enter value of c : ">

call readfloat fstp cz fld b fmul b fild const fmul a fmul cz fsub mov eax,0 mov const,0 fild const fcomi st(0),st(1)

JA L1 JE L2 jmp L3 L1:

mwrite<0ah,0dh,0ah," Since , Discriminant b^2 -4ac < 0 , roots are complex"> call crlf finit

jmp quit L2:

mwrite<0ah," Discriminant b^2 -4ac = 0 , roots are real and same">

finit

mov const,2 fild const fmul a fld b fchs fdiv st(0),st(1)

mwrite<0dh,0ah,0ah," Your Answer : "> call writefloat fstp st(1) call crlf

jmp quit L3:

mwrite<0ah," Discriminant b^2 -4ac > 0 , roots are real and distinct"> fstp st(0) fsqrt

fst cz fld b fchs fadd st(1),st(0) fstp st(0) mov const,2 fild const fmul a fdiv st(1),st(0) fstp st(0) mwrite<0dh,0ah,0ah," X1 : "> call writefloat fld cz fld b fchs fsub st(1),st(0) fstp st(0) fchs fild const fmul a

fdiv st(1),st(0) fstp st(0) mwrite<0dh,0ah,0ah," X2 : "> call writefloat call crlf quit: ret

QuadraticEquation Endp pointOfIntersection PROC

LOCAL pt1:real4

LOCAL pt2:real4

LOCAL a:SDWORD

LOCAL b:SDWORD

LOCAL cz:SDWORD

LOCAL d:SDWORD

LOCAL e:SDWORD

LOCAL f:SDWORD

LOCAL Deno:SDWORD

LOCAL Num:SDWORD

LOCAL const:DWORD

finit

mov const,0

mwrite<0ah," ( Straight Line Of Form : ax + by = c )",0ah,0ah> mwrite<0ah," For Line 1:",0ah,0ah,0ah> mwrite<" Enter a :"> call readint mov a,eax

mwrite<0ah," Enter b :"> call readint mov b,eax

mwrite<0ah," Enter c :"> call readint mov cz,eax

mwrite<0ah,0ah," For Line 2:",0ah,0ah> mwrite<0ah," Enter a :"> call readint mov d,eax

mwrite<0ah," Enter b :"> call readint

mov e,eax mwrite<0ah," Enter c :"> call readint mov f,eax mov eax,b imul d mov deno,eax mov eax,a imul e sub eax,Deno mov Deno,eax mov eax,e imul cz

mov Num,eax mov eax,b imul f

sub Num,eax fild Deno fild const fcomi st(0),st(1) JE L1 jmp l2 L1: mwrite< 0ah, " Solution Not Possible",0ah> finit

jmp quit l2: finit

fild Num fild Deno fdiv st(1),st(0) fstp st(0) fstp pt1

mwrite<0ah,0ah," Point Of Intersection ( "> mov eax,Num

call writeint mwrite<"/"> mov eax,Deno

call writeint mov eax,a imul f

mov Num,eax mov eax,cz imul d sub Num,eax mov eax,Num fild Num fild Deno fdiv st(1),st(0) fstp st(0) fstp pt2 mwrite<" , "> call writeint

mov eax,Deno mwrite<"/"> call writeint mwrite<" ) OR ",0ah> call crlf fld pt1 mwrite<" ("> call writefloat mwrite<" ,"> fld pt2 call writefloat mwrite<" )">

call crlf quit: ret

pointOfIntersection Endp Distance PROC

LOCAL x1:SDWORD LOCAL x2:SDWORD

LOCAL y1:SDWORD

LOCAL y2:SDWORD

finit

mwrite<0ah,0ah," ( Distance B/W Points )"> mwrite<0ah,0ah," Enter x1 : "> call readint mov x1,eax

mwrite<0ah," Enter x2 : ">

call readint mov x2,eax mwrite<0ah," Enter y1 : "> call readint mov y1,eax mwrite<0ah," Enter y2 : "> call readint mov y2,eax mov eax,x1 sub eax,x2 imul eax mov x1,eax mov eax,y1 sub eax,y2 imul eax add eax,x1

mwrite<0ah,0ah," Distance : sqrt("> call writedec mov x1,eax

fild x1

fsqrt mwrite<") = "> call writefloat mwrite<" approx",0ah>

ret

Distance Endp

ShowLineDetail PROC

LOCAL a : SDWORD

LOCAL b : SDWORD

LOCAL cz : SDWORD

LOCAL x : real4

LOCAL y : real4

LOCAL m : real4 LOCAL const:real4

finit

mwrite<0ah,0ah," ( Line of the form ax + by = c

)",0ah,0ah>

mwrite<" Enter a : "> call readint mov a,eax mwrite<0ah," Enter b : "> call readint mov b,eax mwrite<0ah," Enter c : "> call readint mov cz,eax fild b fild a fdiv st(0),st(1) fchs

mwrite<0ah," slope : "> call writefloat fstp m fild cz fdiv st(0),st(1)

mwrite<0ah,0ah," y-intercept : "> call writefloat fstp y finit fild a

fild cz fdiv st(0),st(1)

mwrite<0ah,0ah," x-intercept : "> call writefloat fstp x finit fild a fchs fild b fpatan fldpi

mov const,180 fild const fdiv st(0),st(1) fmul st(2),st(0) fstp st(0) fstp st(0)

mwrite<0ah,0ah," Angle from x-axis: "> call writefloat mwrite<" degrees">

fld m fld y fld x call crlf ret

ShowLineDetail Endp

AreaOfTriangle PROC

LOCAL a: real4

LOCAL b: real4

LOCAL cz: real4

LOCAL s:real4 LOCAL const:real4

finit

mov const,2

mwrite<0ah,0ah," ( Area Of Triangle ): "> mwrite<0ah,0ah,0ah," Enter lenght of side 1 : "> call readfloat fstp a

mwrite<0ah," Enter lenght of side 2 : "> call readfloat

fstp b

mwrite<0ah," Enter lenght of side 3 : "> call readfloat fst cz fld b fld a fadd st(0),st(1) fstp st(1) fadd st(0),st(1) fstp st(1) fild const fdiv st(1),st(0) fstp st(0) fstp s fld a fld s fsub st(0),st(1) fstp st(1) fld b fld s

fsub st(0),st(1) fstp st(1) fld cz fld s fsub st(0),st(1) fstp st(1) fld s fmul st(0),st(1) fstp st(1) fmul st(0),st(1) fstp st(1) fmul st(0),st(1) fstp st(1) mov const,0 fcomi st(0),st(1) JAE L1 fstp st(1) fsqrt

mwrite<0ah,0ah," Area of triangle is : "> call writefloat

mwrite<" sq.units">

call crlf jmp quit L1:

mwrite<0ah,0ah," Area of triangle is not possible"> finit quit: ret

AreaOfTriangle Endp

AngleBWLines PROC

LOCAL m1 : real4

LOCAL m2 : real4

LOCAL const:DWORD

finit

mov const,1

mwrite<0ah," ( Angle B/W Lines )",0ah,0ah> mwrite<" Enter slop of line 1 : "> call readfloat fst m1

mwrite<" Enter slop of line 2 : ">

call readfloat fst m2 fsub st(0),st(1) fstp st(1) fld m1 fld m2 fmul st(0),st(1) fstp st(1) fild const fadd fdiv st(1),st(0) fstp st(0) fabs fild const fpatan mov const,180 fild const fldpi fdiv st(1),st(0) fstp st(0)

fmul st(1),st(0) fstp st(0)

mwrite<0ah,0ah," Angle : "> call writefloat mwrite<" degrees",0ah,0ah> ret

AngleBWLines Endp trigonometricRatios PROC

LOCAL sinQ : real4

LOCAL cosQ : real4 LOCAL tanQ : real4

LOCAL cotQ : real4

LOCAL secQ : real4

LOCAL cosecQ : real4

LOCAL angle : real4

LOCAL const : DWORD

finit

mwrite<0ah,0ah," >>>>>>>> Trigonometric Ratios

<<<<<<<<<",0ah,0ah,0ah> mwrite<" Enter angle in degrees : "> call readfloat fstp angle mov const,180 fild const fldpi fdiv st(0),st(1) fstp st(1) fld angle fmul st(0),st(1) fstp st(1) fst angle fsin fst sinQ mov const,1 fild const fdiv st(0),st(1) fstp st(1) fstp cosecQ fld angle fcos

fst cosQ fild const fdiv st(0),st(1) fstp st(1) fstp secQ fld sinQ mwrite<0ah,0ah," sinQ : "> call writefloat fld cosecQ mwrite<" cosecQ : "> call writefloat fld cosQ mwrite<0ah,0ah," cos : "> call writefloat fld secQ mwrite<" sec : "> call writefloat fld angle fptan fstp angle

fst tanQ mov const,1 fild const fdiv st(0),st(1) fstp st(1) fstp cotQ fld tanQ mwrite<0ah,0ah," tanQ : "> call writefloat fld cotQ mwrite<" cotQ : "> call writefloat call crlf ret trigonometricRatios Endp

CentroidOfTriangle PROC

LOCAL x1:real4

LOCAL x2:real4

LOCAL x3:real4 LOCAL y1:real4 LOCAL y2:real4

LOCAL y3:real4

LOCAL const:DWORD mov const,3

finit

mwrite<0ah,0ah," >>>>>>>> Centroid Of Triangle

<<<<<<<",0ah,0ah> mwrite<" For vertex A: ",0ah> mwrite<" Enter x1 : "> call readfloat fstp x1

mwrite<" Enter y1 : "> call readfloat fstp y1

mwrite<0ah," For vertex B: ",0ah> mwrite<" Enter x2 : "> call readfloat fstp x2

mwrite<" Enter y2 : "> call readfloat

fstp y2

mwrite<0ah," For vertex C: ",0ah> mwrite<" Enter x3 : "> call readfloat fstp x3 mwrite<" Enter y3 : "> call readfloat fstp y3 fld x1 fld x2 fld x3 fadd st(0),st(1) fstp st(1) fadd st(0),st(1) fstp st(1) fild const fdiv st(1),st(0) fstp st(0)

mwrite<0ah,0ah," Coordinates : ( "> call writefloat

fld y1 fld y2 fld y3 fadd st(0),st(1) fstp st(1) fadd st(0),st(1) fstp st(1) fild const fdiv st(1),st(0) fstp st(0) mwrite<" , "> call writefloat mwrite<"),0ah"> ret

CentroidOfTriangle Endp

IncentreOfTriangle PROC

LOCAL x1:real4

LOCAL x2:real4

LOCAL x3:real4 LOCAL y1:real4 LOCAL y2:real4

LOCAL y3:real4

LOCAL const:DWORD

LOCAL a:real4

LOCAL b:real4

LOCAL cz:real4

LOCAL X: real4 LOCAL Y:real4

finit

mov const,3

mwrite<0ah,0ah," >>>>>>>> Incentre Of Triangle

<<<<<<<",0ah,0ah> mwrite<" For vertex A: ",0ah> mwrite<" Enter x1 : "> call readfloat fstp x1

mwrite<" Enter y1 : "> call readfloat fstp y1

mwrite<0ah," For vertex B: ",0ah>

mwrite<" Enter x2 : "> call readfloat fstp x2 mwrite<" Enter y2 : "> call readfloat fstp y2

mwrite<0ah," For vertex C: ",0ah> mwrite<" Enter x3 : "> call readfloat fstp x3 mwrite<" Enter y3 : "> call readfloat fstp y3 fld x2 fld x3 fsub st(0),st(1) fstp st(1) fmul st(0),st(0) fld y2 fld y3

fsub st(0),st(1) fstp st(1) fmul st(0),st(0) fadd st(0),st(1) fstp st(1) fsqrt fstp a fld x3 fld x1 fsub st(0),st(1) fstp st(1) fmul st(0),st(0) fld y3 fld y1 fsub st(0),st(1) fstp st(1) fmul st(0),st(0) fadd st(0),st(1) fstp st(1) fsqrt

fstp b fld x1 fld x2 fsub st(0),st(1) fstp st(1) fmul st(0),st(0) fld y1 fld y2 fsub st(0),st(1) fstp st(1) fmul st(0),st(0) fadd st(0),st(1) fstp st(1) fsqrt fst cz fld a fld b fld cz fadd st(0),st(1) fstp st(1)

fadd st(0),st(1) fstp st(1) fld a fld x1 fmul st(0),st(1) fstp st(1) fld b fld x2 fmul st(0),st(1) fstp st(1) fld cz fld x3 fmul st(0),st(1) fstp st(1) fadd st(0),st(1) fstp st(1) fadd st(0),st(1) fstp st(1) fdiv st(0),st(1) fstp X

fld a fld y1 fmul st(0),st(1) fstp st(1) fld b fld y2 fmul st(0),st(1) fstp st(1) fld cz fld y3 fmul st(0),st(1) fstp st(1) fadd st(0),st(1) fstp st(1) fadd st(0),st(1) fstp st(1) fdiv st(0),st(1) fstp Y finit fld X mwrite<0ah,0ah," Coordinates : ( "> call writefloat mwrite<" , "> fld Y call writefloat mwrite<" )">

call crlf ret

IncentreOfTriangle Endp

CircumRadius PROC

LOCAL a: real4

LOCAL b: real4

LOCAL cz: real4

LOCAL s:real4 LOCAL const:real4

finit

mov const,2

mwrite<0ah,0ah," ( Circum-Radius Of Triangle ): "> mwrite<0ah,0ah,0ah," Enter lenght of side 1 : "> call readfloat

fstp a

mwrite<0ah," Enter lenght of side 2 : "> call readfloat fstp b

mwrite<0ah," Enter lenght of side 3 : "> call readfloat fst cz fld b fld a fadd st(0),st(1) fstp st(1) fadd st(0),st(1) fstp st(1) fild const fdiv st(1),st(0) fstp st(0) fstp s fld a fld s fsub st(0),st(1)

fstp st(1) fld b fld s fsub st(0),st(1) fstp st(1) fld cz fld s fsub st(0),st(1) fstp st(1) fld s fmul st(0),st(1) fstp st(1) fmul st(0),st(1) fstp st(1) fmul st(0),st(1) fstp st(1) fsqrt mov const,4 fild const fmul st(0),st(1) fstp st(1) fld a fld b fld cz fmul st(0),st(1) fstp st(1) fmul st(0),st(1) fstp st(1) fdiv st(0),st(1) fstp st(1)

mwrite<0ah,0ah," Circum-Radius : "> call writefloat call crlf ret

CircumRadius Endp Magnitude PROC

mwrite<0ah,0ah," ( Magnitude Of Vector

)",0ah,0ah,0ah>

finit

mwrite<" Enter x-coordinate : ">

call readfloat fmul st(0),st(0)

mwrite<0ah," Enter y-coordinate : "> call readfloat fmul st(0),st(0)

mwrite<0ah," Enter z-coordinate : "> call readfloat fmul st(0),st(0) fadd st(0),st(1) fstp st(1) fadd st(0),st(1) fstp st(1)

mwrite<0ah,0ah," Magnitude : sqrt( "> call writefloat mwrite<" ) = "> fsqrt call writefloat mwrite<" approx"> call crlf ret

Magnitude Endp

UnitVector PROC USES eax

LOCAL a:SDWORD

LOCAL b:SDWORD

LOCAL cz:SDWORD

LOCAL m:real4 LOCAL n:SDWORD

mwrite<0ah,0ah," >>>>>>>> Unit Vector

<<<<<<",0ah,0ah,0ah> mwrite<" Enter x-coordinate : "> call readint mov a,eax

fild a fmul st(0),st(0)

mwrite<0ah," Enter y-coordinate : "> call readint mov b,eax

fild b fmul st(0),st(0)

mwrite<0ah," Enter z-coordinate : ">

call readint mov cz,eax

fild cz fmul st(0),st(0) fadd st(0),st(1) fstp st(1) fadd st(0),st(1) fstp st(1) fist n

mwrite<0ah,0ah," Unit Vector : ( "> mov eax,a

call writeint mwrite<"i "> mov eax,b

call writeint mwrite<"j "> mov eax,cz call writeint mwrite<"k ) / sqrt "> mov eax,n

call writedec

mwrite<0ah,0ah," : ("> fsqrt fstp m fild a fdiv m call writefloat mwrite<")i ("> fild b fdiv m call writefloat mwrite<")j (">

fild cz fdiv m call writefloat mwrite<")k ">

call crlf ret

UnitVector Endp

DotProduct PROC

LOCAL x1:SDWORD LOCAL y1:SDWORD

LOCAL z1:SDWORD

LOCAL x2:SDWORD LOCAL y2:SDWORD LOCAL z2:SDWORD

mwrite<0ah,0ah," >>>>>>>> Dot Product

<<<<<<",0ah,0ah,0ah> mwrite<" For Vector 1 : ",0ah,0ah> mwrite<" Enter x-coordinate : "> call readint mov x1,eax

mwrite<0ah," Enter y-coordinate : "> call readint mov y1,eax

mwrite<0ah," Enter z-coordinate : "> call readint mov z1,eax

mwrite<0ah,0ah," For Vector 2 : ",0ah,0ah> mwrite<" Enter x-coordinate : ">

call readint mov x2,eax

mwrite<0ah," Enter y-coordinate : "> call readint mov y2,eax

mwrite<0ah," Enter z-coordinate : "> call readint mov z2,eax mov eax,x1 imul x2 mov ebx,eax mov eax,y1 imul y2 add ebx,eax mov eax,z1 imul z2 add ebx,eax mov eax,ebx

mwrite<0ah,0ah," Dot Product : "> call writeint

mov ebx,0 cmp eax,ebx

JE L1 jmp quit L1:

mwrite<" ( Perpendicual Vectors )"> quit: call crlf ret

DotProduct Endp

Angle\_BW\_Vectors PROC

LOCAL x1:SDWORD LOCAL y1:SDWORD

LOCAL z1:SDWORD

LOCAL x2:SDWORD LOCAL y2:SDWORD

LOCAL z2:SDWORD LOCAL save:real4

mwrite<0ah,0ah," >>>>>>>> Angle B/W Vectors

<<<<<<",0ah,0ah,0ah>

finit

mwrite<" For Vector 1 : ",0ah,0ah> mwrite<" Enter x-coordinate : "> call readint mov x1,eax

fild x1 fmul st(0),st(0)

mwrite<0ah," Enter y-coordinate : "> call readint mov y1,eax

fild y1 fmul st(0),st(0)

mwrite<0ah," Enter z-coordinate : "> call readint mov z1,eax

fild z1 fmul st(0),st(0) fadd st(0),st(1) fstp st(1) fadd st(0),st(1) fstp st(1) fsqrt

mwrite<0ah,0ah," For Vector 2 : ",0ah,0ah> mwrite<" Enter x-coordinate : "> call readint mov x2,eax

fild x2 fmul st(0),st(0)

mwrite<0ah," Enter y-coordinate : "> call readint mov y2,eax

fild y2 fmul st(0),st(0)

mwrite<0ah," Enter z-coordinate : "> call readint mov z2,eax

fild z2 fmul st(0),st(0) fadd st(0),st(1) fstp st(1)

fadd st(0),st(1) fstp st(1) fsqrt fmul st(0),st(1) fstp st(1) mov eax,x1 imul x2 mov ebx,eax mov eax,y1 imul y2 add ebx,eax mov eax,z1 imul z2 add ebx,eax mov z1,ebx fild z1 fdiv st(0),st(1) fstp st(1) fst save fmul st(0),st(0) fchs mov x1,1

fild x1 fadd st(0),st(1) fstp st(1) fsqrt fld save fpatan mwrite<0ah,0ah," Angle :( "> call writefloat mwrite<" ) radians",0ah,0ah> mov x1,180

fldpi fild x1 fdiv st(0),st(1) fstp st(1) fmul st(0),st(1) mwrite<" :( "> call writefloat mwrite<" ) degrees"> call crlf ret

Angle\_BW\_Vectors Endp

VectorProduct PROC

LOCAL x1:SDWORD LOCAL y1:SDWORD

LOCAL z1:SDWORD

LOCAL x2:SDWORD LOCAL y2:SDWORD

LOCAL z2:SDWORD LOCAL const:DWORD mov const,-1

mwrite<0ah,0ah," >>>>>>>> Vector Product / Perpendicular Product <<<<<<",0ah,0ah,0ah> finit

mwrite<" For Vector 1 : ",0ah,0ah> mwrite<" Enter x-coordinate : "> call readint mov x1,eax

mwrite<0ah," Enter y-coordinate : ">

call readint mov y1,eax

mwrite<0ah," Enter z-coordinate : "> call readint mov z1,eax

mwrite<0ah,0ah," For Vector 2 : ",0ah,0ah> mwrite<" Enter x-coordinate : "> call readint mov x2,eax

mwrite<0ah," Enter y-coordinate : "> call readint mov y2,eax

mwrite<0ah," Enter z-coordinate : "> call readint mov z2,eax mov eax,y1 imul z2 mov ebx,eax mov eax,y2 imul z1

sub ebx,eax mov ecx,ebx mov eax,x1 imul z2 mov ebx,eax mov eax,x2 imul z1 sub ebx,eax mov eax,ebx imul const mov ebx,eax mov eax,x1 imul y2 mov z1,eax mov eax,x2 imul y1 sub z1,eax mov eax,z1

mwrite<0ah,0ah," Vector Product / Perpendicular Vector : "> mov z1,eax

mov eax,ecx call writeint mwrite<"i "> mov eax,ebx call writeint mwrite<"j "> mov eax,z1 call writeint mwrite<"k ">

call crlf ret

VectorProduct Endp

LinearCombination PROC

LOCAL pt1:real4

LOCAL pt2:real4

LOCAL a:SDWORD

LOCAL b:SDWORD

LOCAL cz:SDWORD

LOCAL d:SDWORD

LOCAL e:SDWORD

LOCAL f:SDWORD

LOCAL Deno:SDWORD

LOCAL Num:SDWORD LOCAL const:DWORD

finit

mov const,0

mwrite<0ah," ( Linear Combination )",0ah,0ah> mwrite<0ah," For vector v1:",0ah,0ah,0ah> mwrite<" Enter x-coordinate :"> call readint mov a,eax

mwrite<0ah," Enter y-coordinate :"> call readint mov d,eax

mwrite<0ah,0ah," For vector v2 :",0ah,0ah> mwrite<0ah," Enter x-coordinate :"> call readint mov b,eax

mwrite<0ah," Enter y-coordinate :">

call readint mov e,eax

mwrite<0ah,0ah," For vector v3 (To be expressed) :",0ah,0ah> mwrite<0ah," Enter x-coordinate :"> call readint mov cz,eax

mwrite<0ah," Enter y-coordinate :"> call readint mov f,eax mov eax,b imul d mov deno,eax mov eax,a imul e sub eax,Deno mov Deno,eax mov eax,e imul cz

mov Num,eax mov eax,b imul f

sub Num,eax fild Deno fild const fcomi st(0),st(1) JE L1 jmp l2 L1:

mwrite< 0ah, " Multiple Solution Or May Be No Solution exists",0ah>

finit

jmp quit l2: finit

fild Num fild Deno fdiv st(1),st(0) fstp st(0) fstp pt1

mwrite<0ah,0ah," Solution : V3 = ( ">

mov eax,Num

call writeint mwrite<"/"> mov eax,Deno

call writeint

mov eax,a imul f

mov Num,eax mov eax,cz imul d sub Num,eax mov eax,Num fild Num fild Deno fdiv st(1),st(0) fstp st(0) fstp pt2 mwrite<" ) v1 + ( "> call writeint

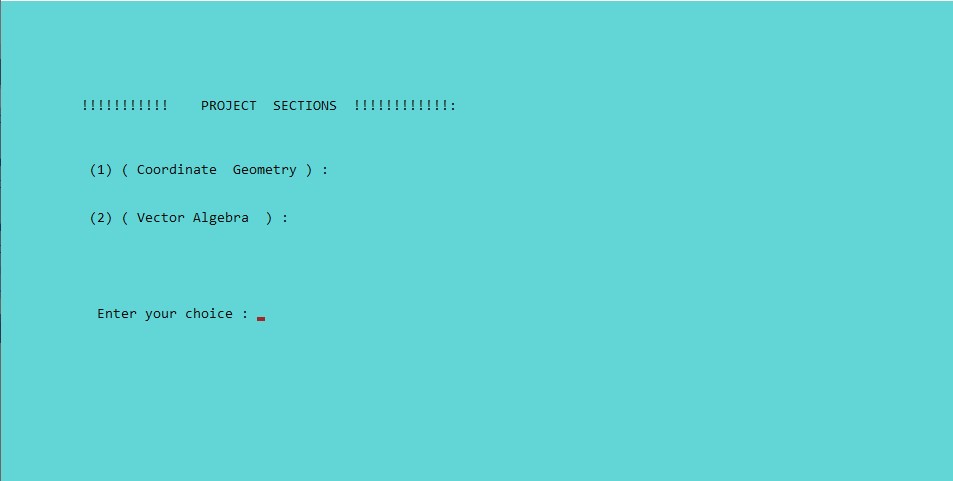
mov eax,Deno mwrite<"/"> call writeint mwrite<" ) V2",0ah>

call crlf fld pt1 fld pt2 call crlf quit: ret

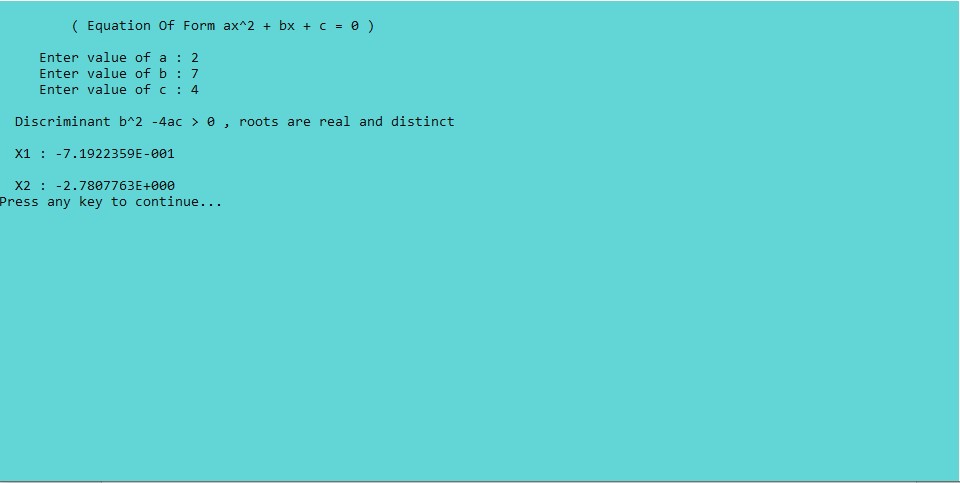
LinearCombination Endp end main

RESULTS / SOFTWARE SIMULATION AND DISCUSSION :

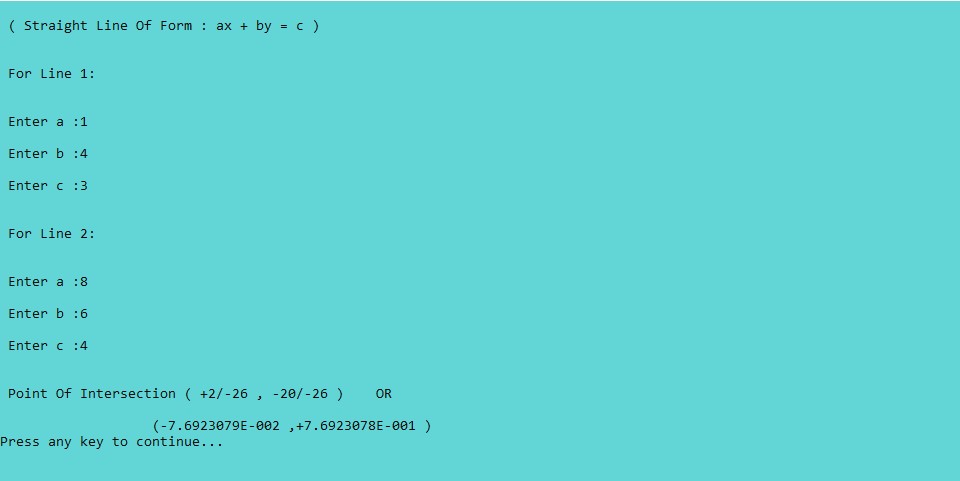
# 1)



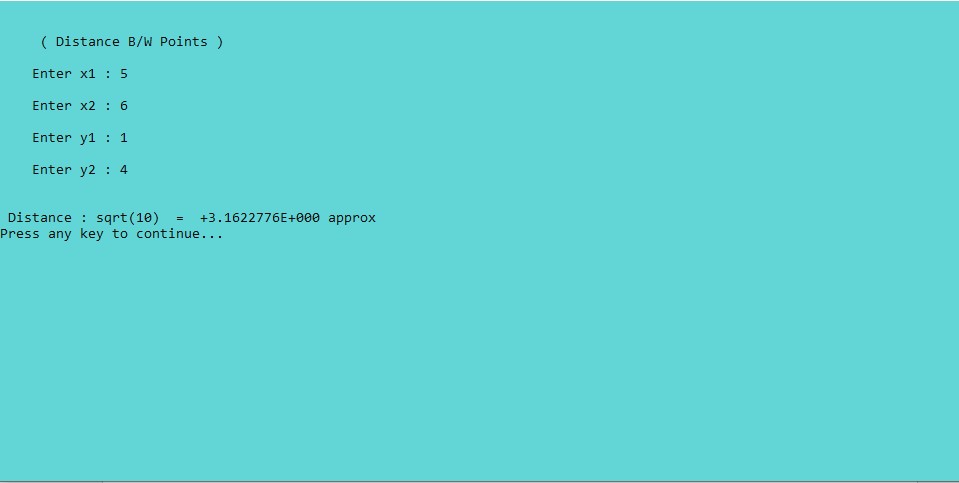
# 2)



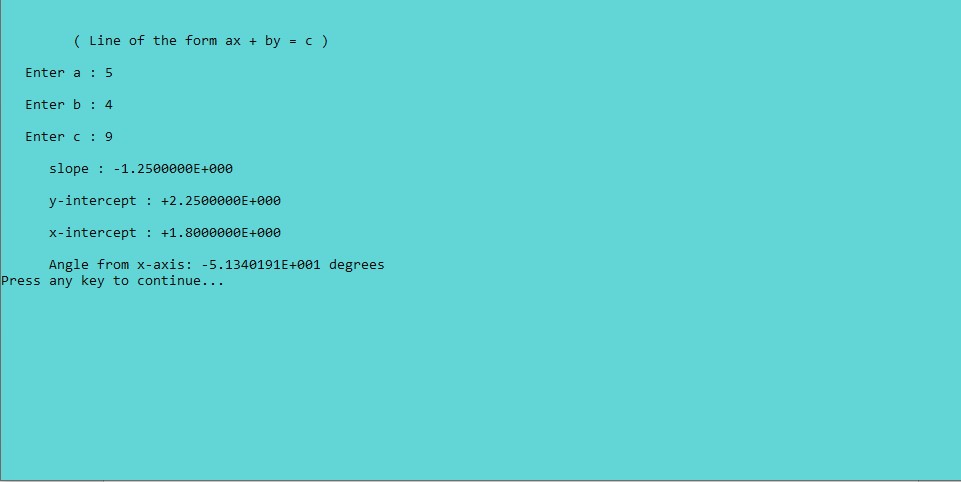
# 3)



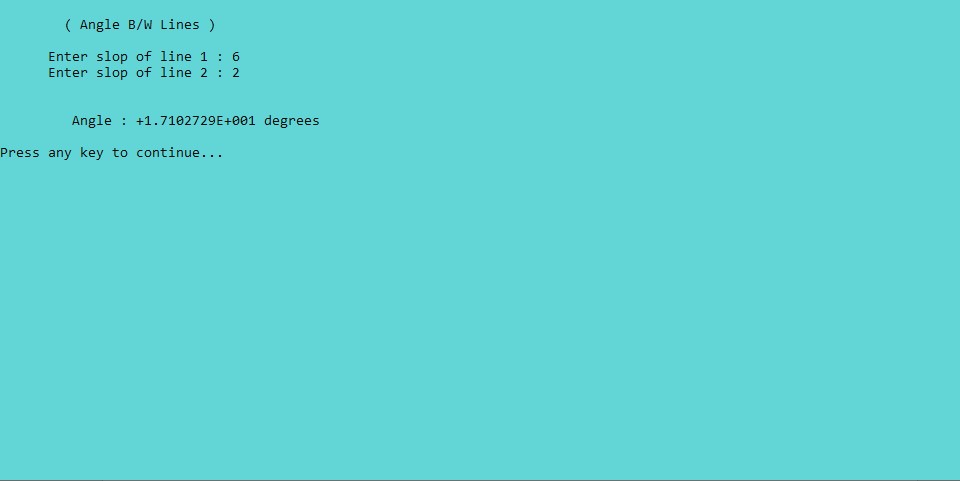
# 4)



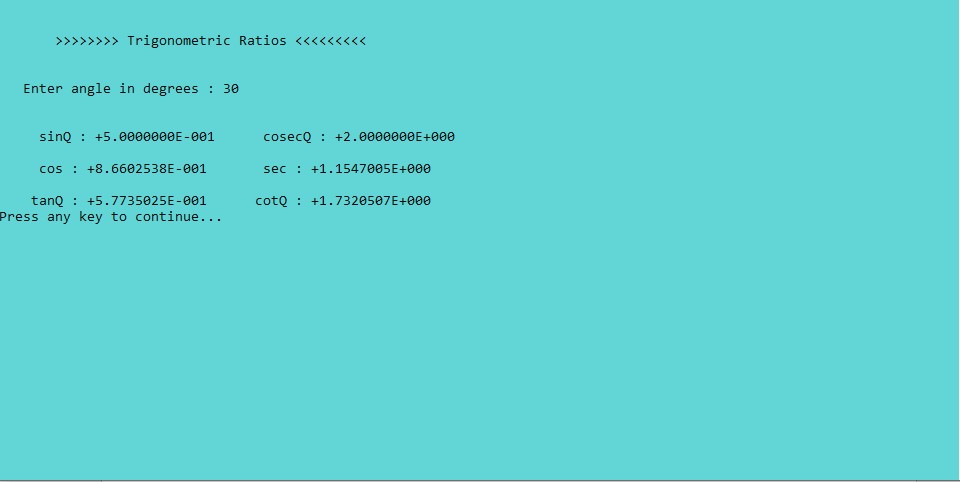
# 5)



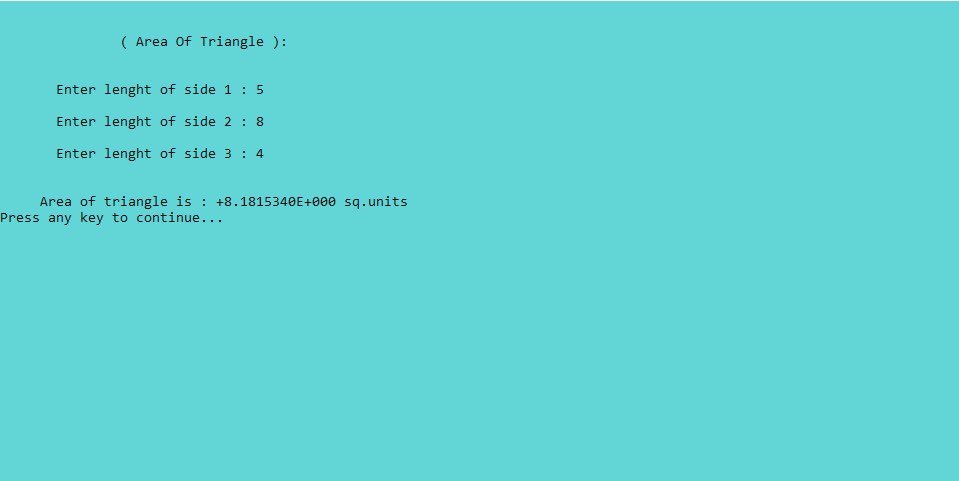
# 6)



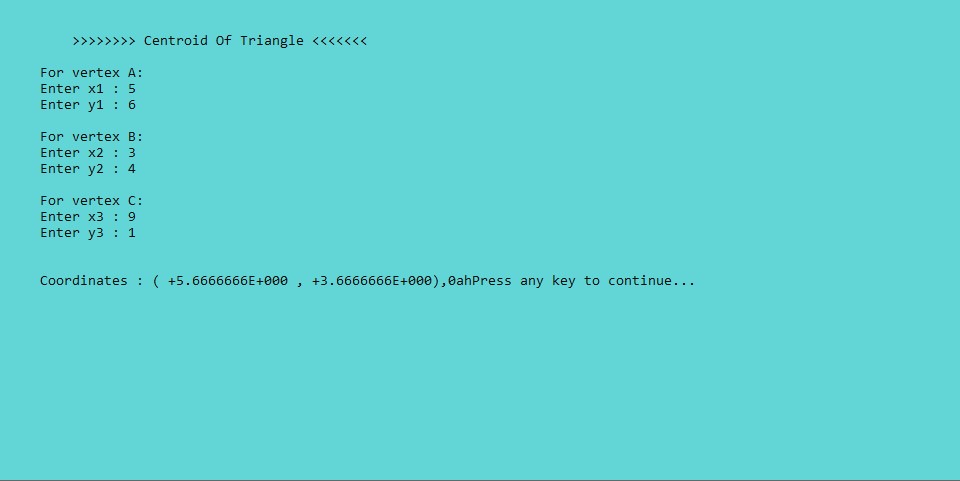
# 7)



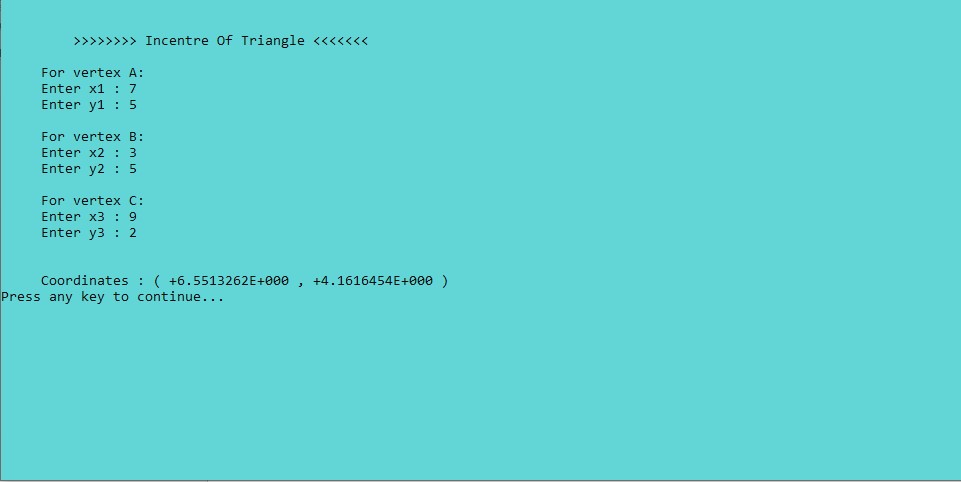
# 8)



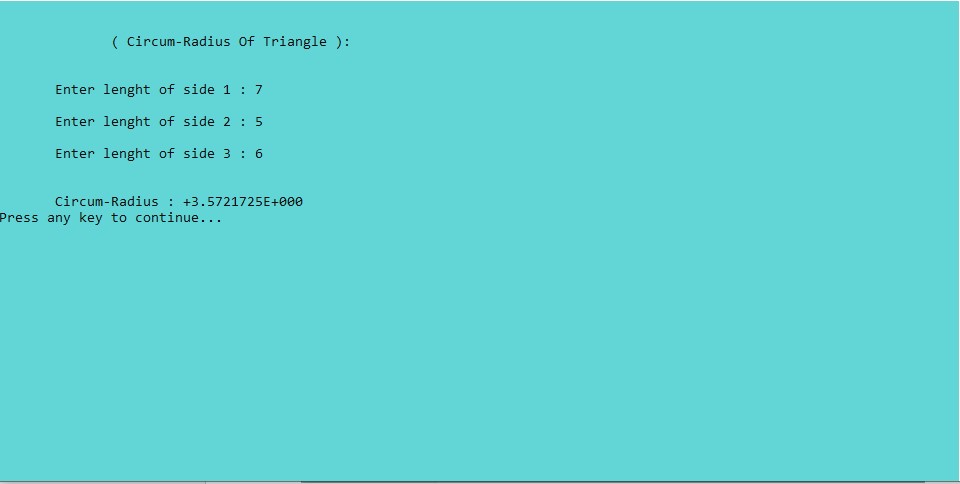
# 9)



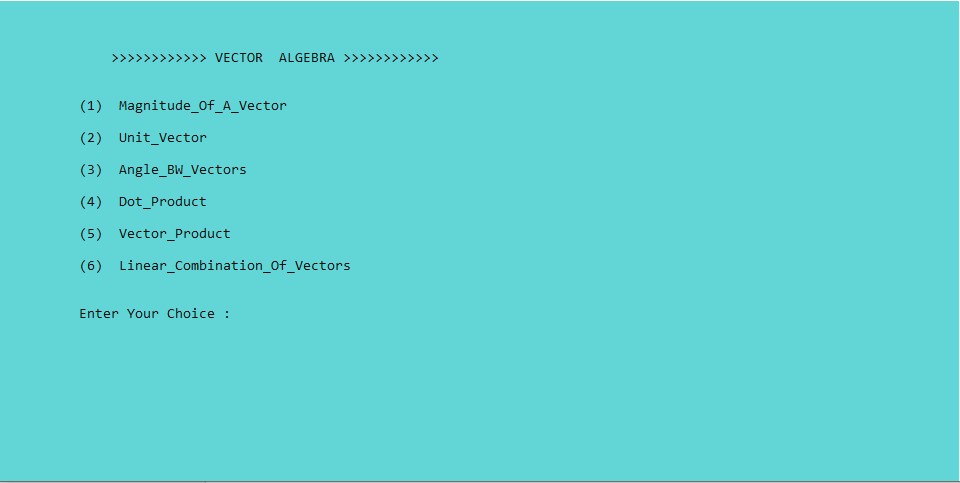
# 10)



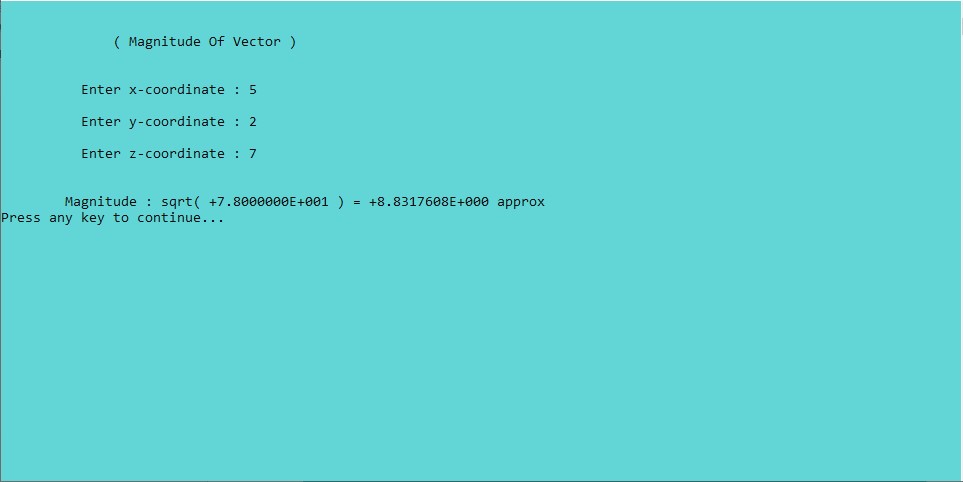
# 11)



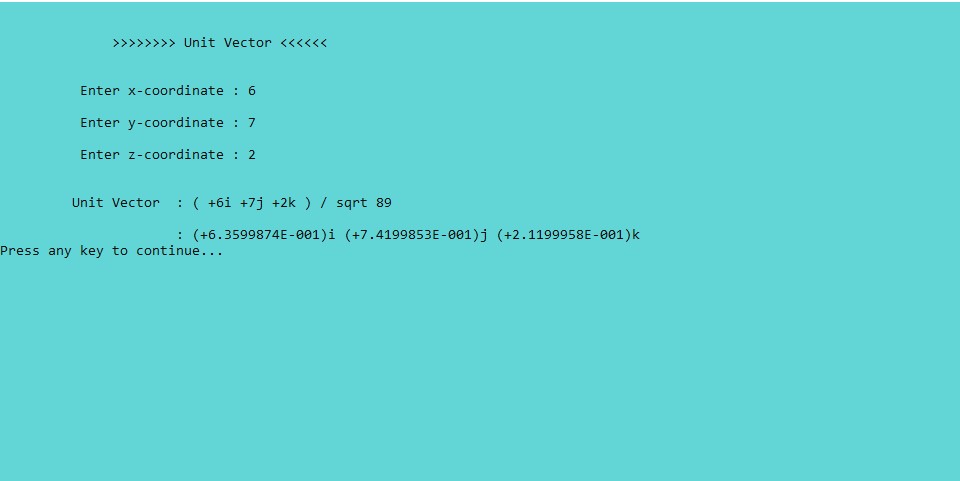
# 12)



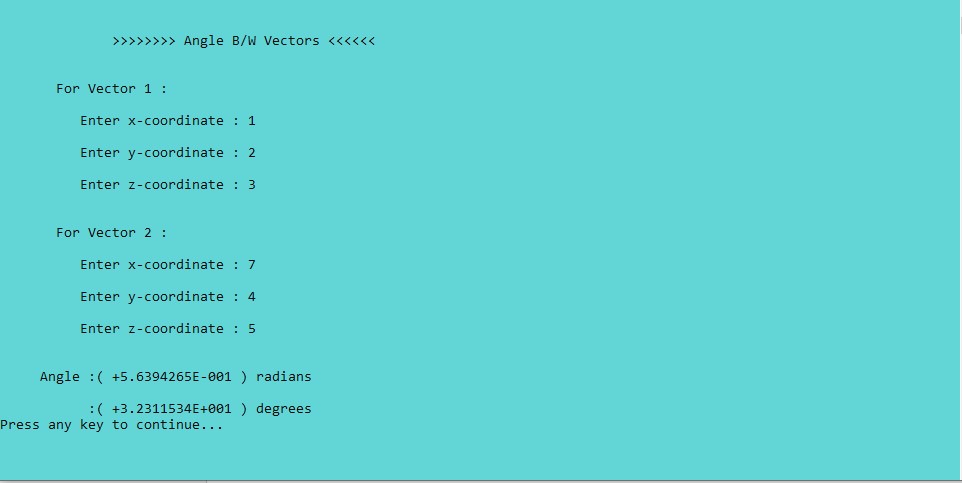
# 13)



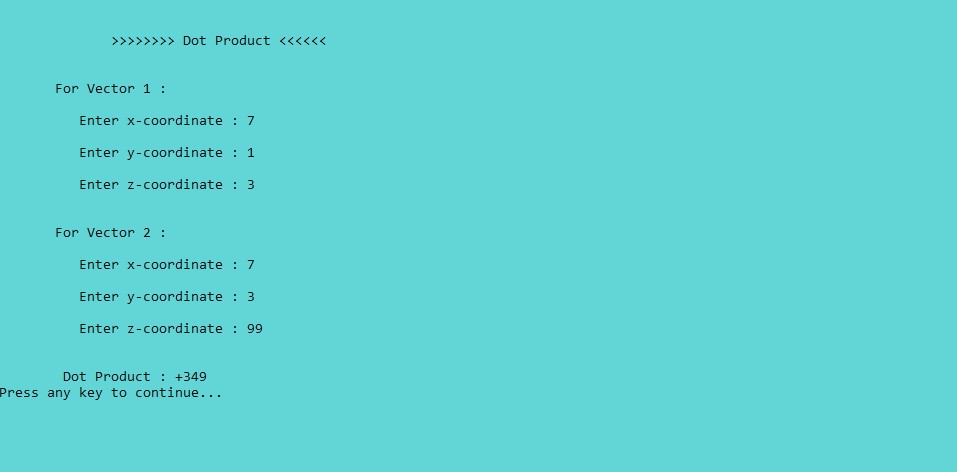
# 14)



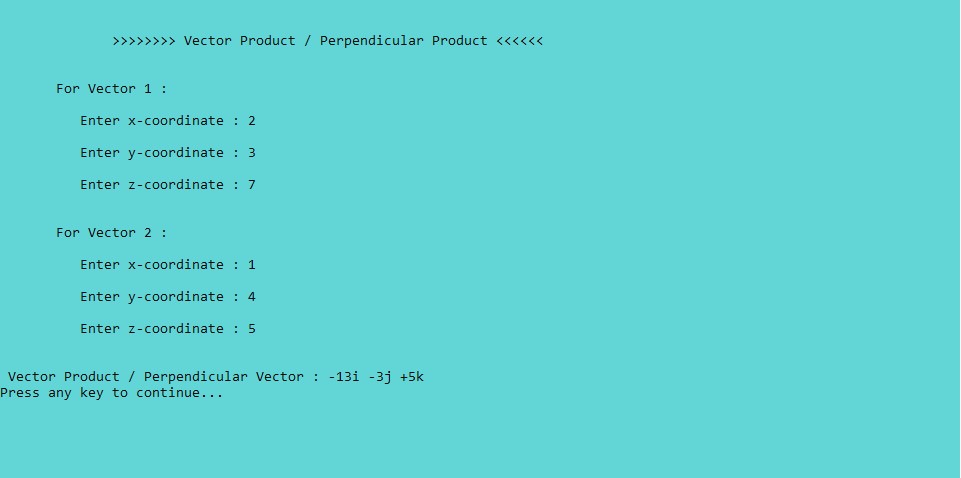
# 15)



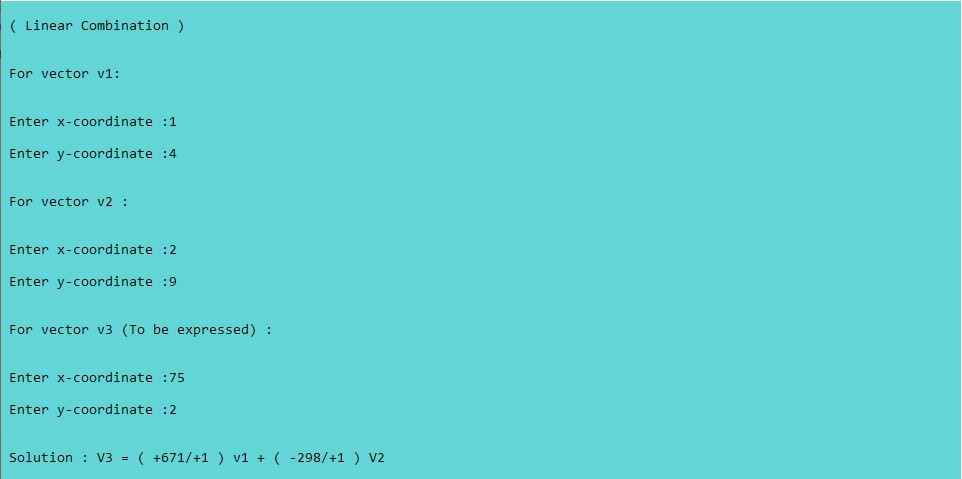
# 16)



# 17)



# 18)



CONCLUSION COST AND FUTURE WORK:

As mathematics is applicable in almost every field nowadays so keeping this in mind we designed this and will be helpful in solving mathematics problems.

REFERENCES :

By taking help of internet ,we achieved our goal in designing this project

>>>>>>>>THANK YOU<<<<<<<<