Tarea Porticos

Por: Cesar Fernando Gamba Tiusaba

Código: 215524

Problema.

18 Kn/m

E = 20 MPa.

5

2

40 kN

3

4

1

6

60 kN

2

4 m

30 kN

20 kN

3

20 kN

8

11

1

9

7

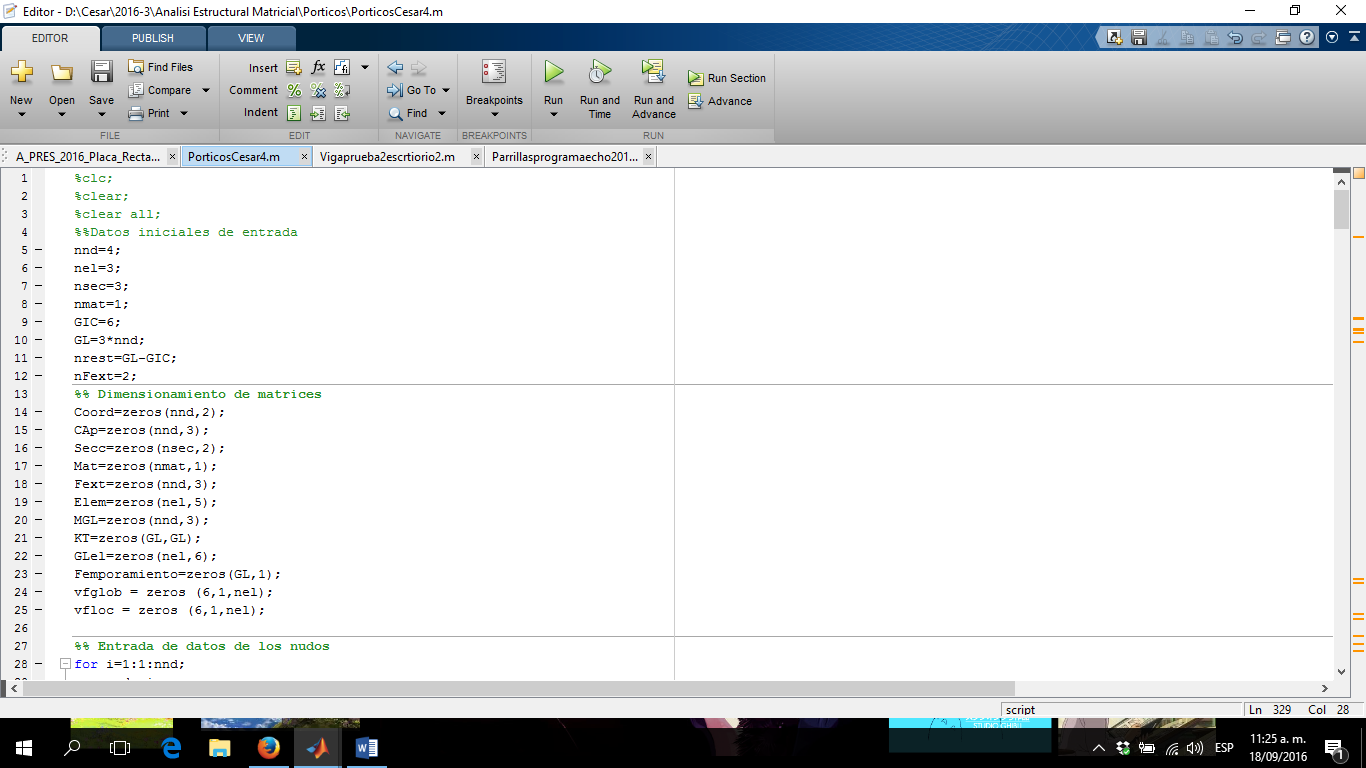
10

12

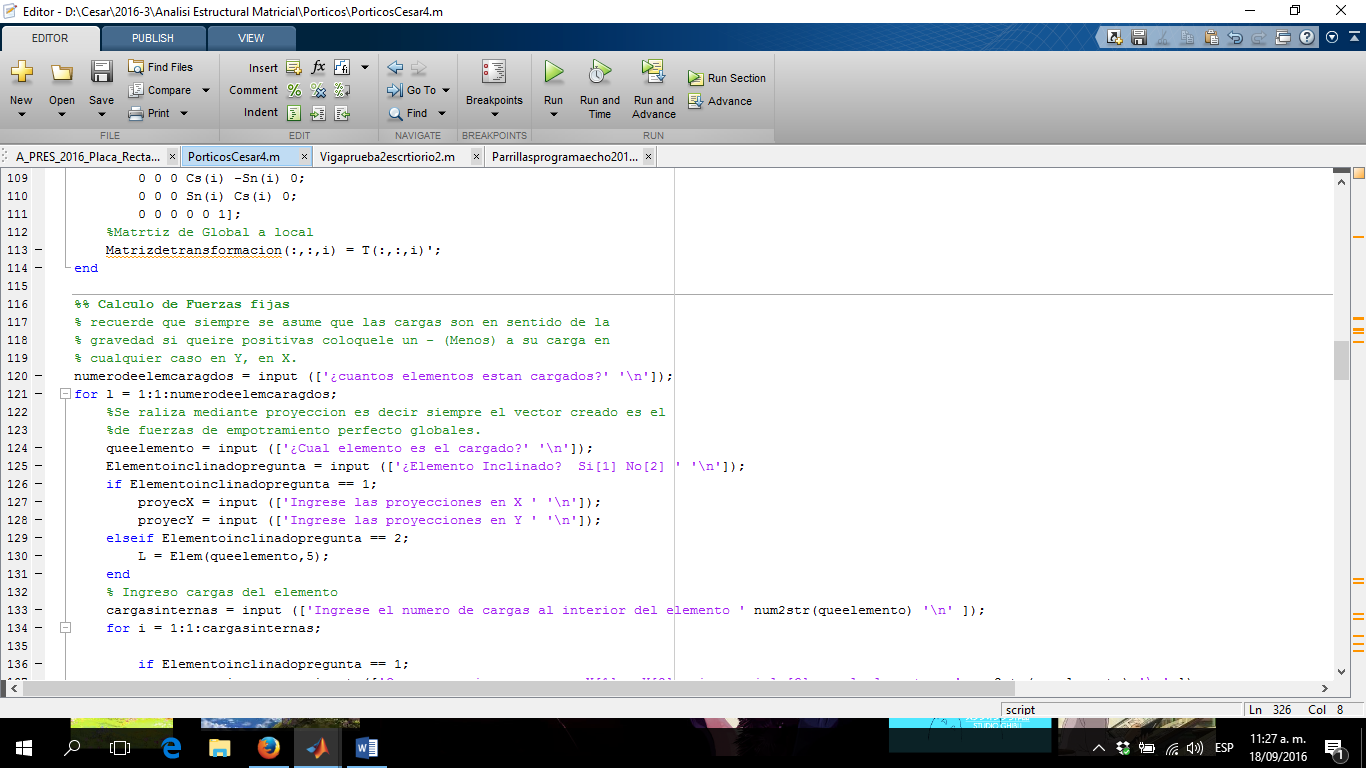
3 m

6 m

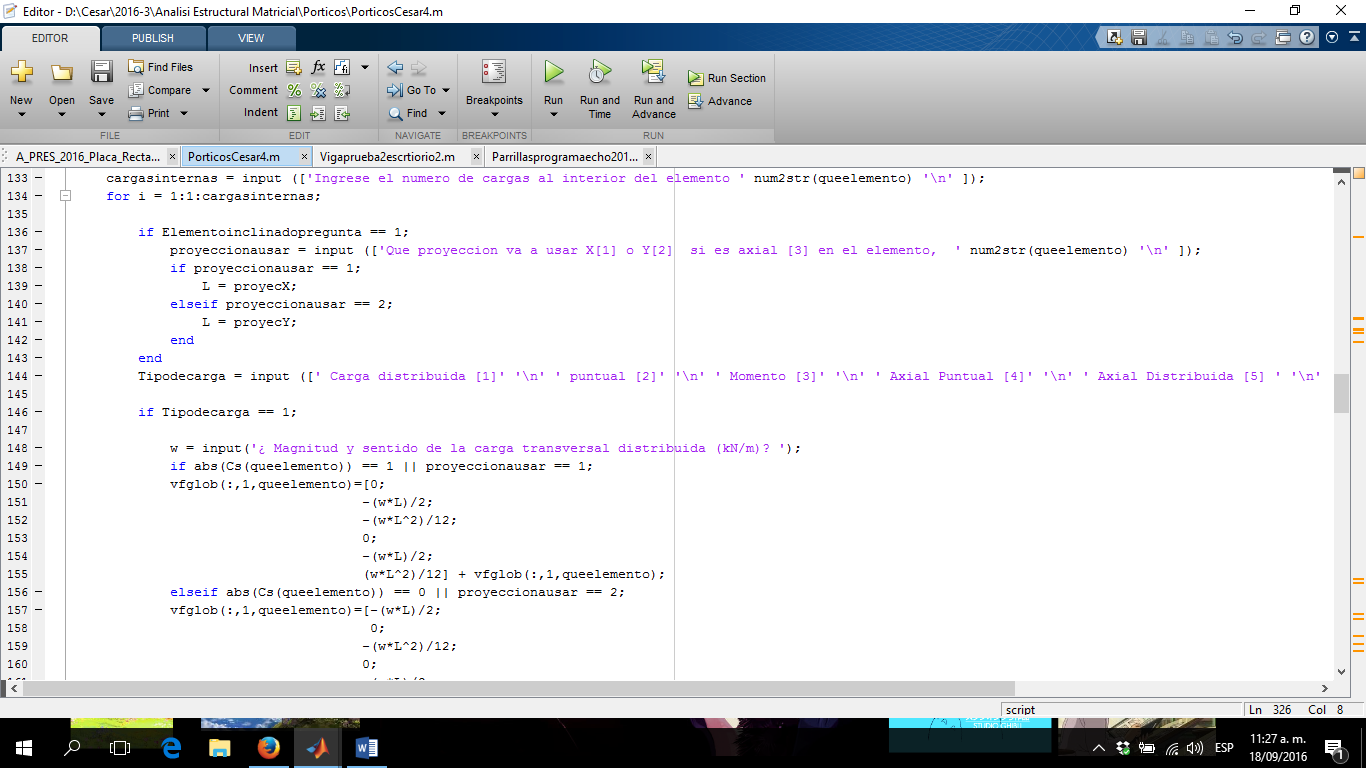
Ingreso de datos:



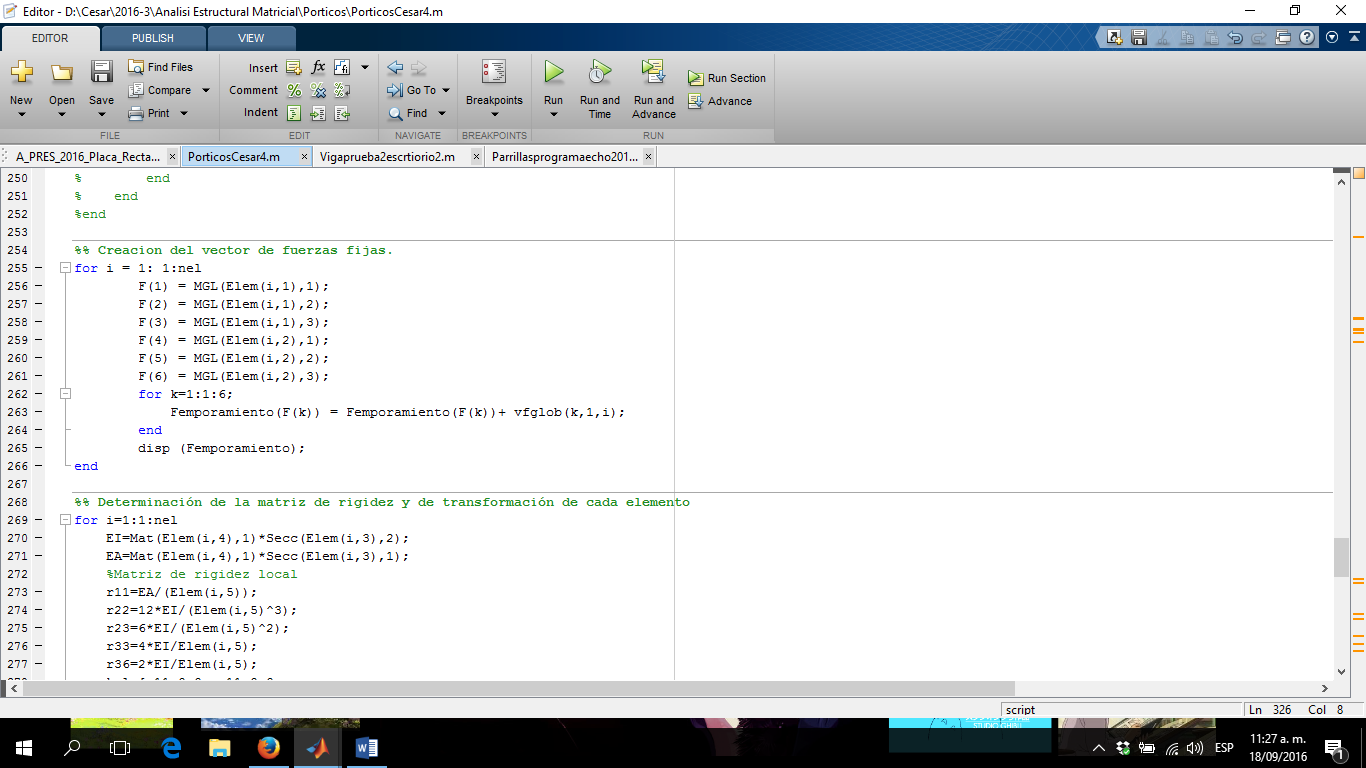
Se piden el numero de elementos cargados y seguidamente cual elemento es el cargado, tambien se le pregunta el numero de cargas que hay al interior de cada elemento, asi el usuario puede ingresar todas las acciones que quiera.

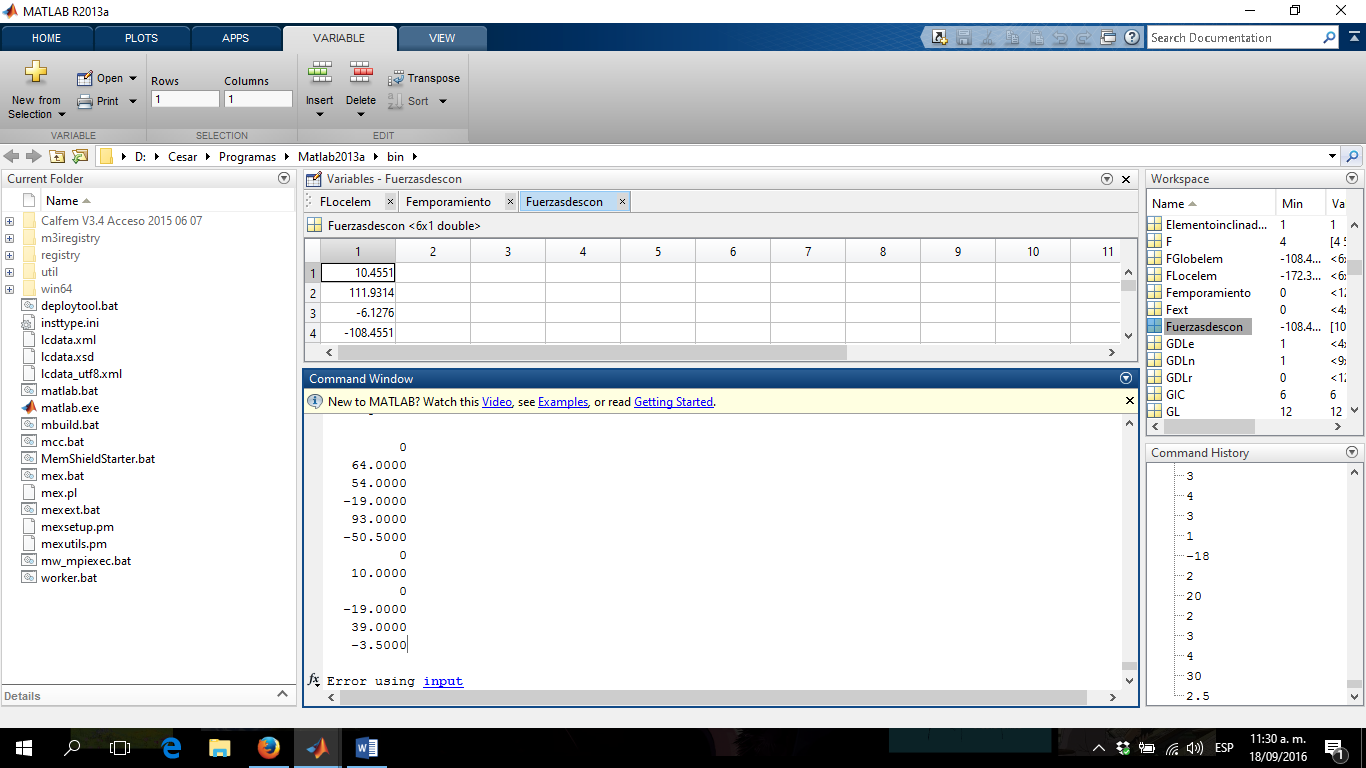


Luego dependiendo de si es una carga distribuida o una carga puntual ubicada a cualquier distancia del elemento, se realizaran las operacuiones correspondientes.

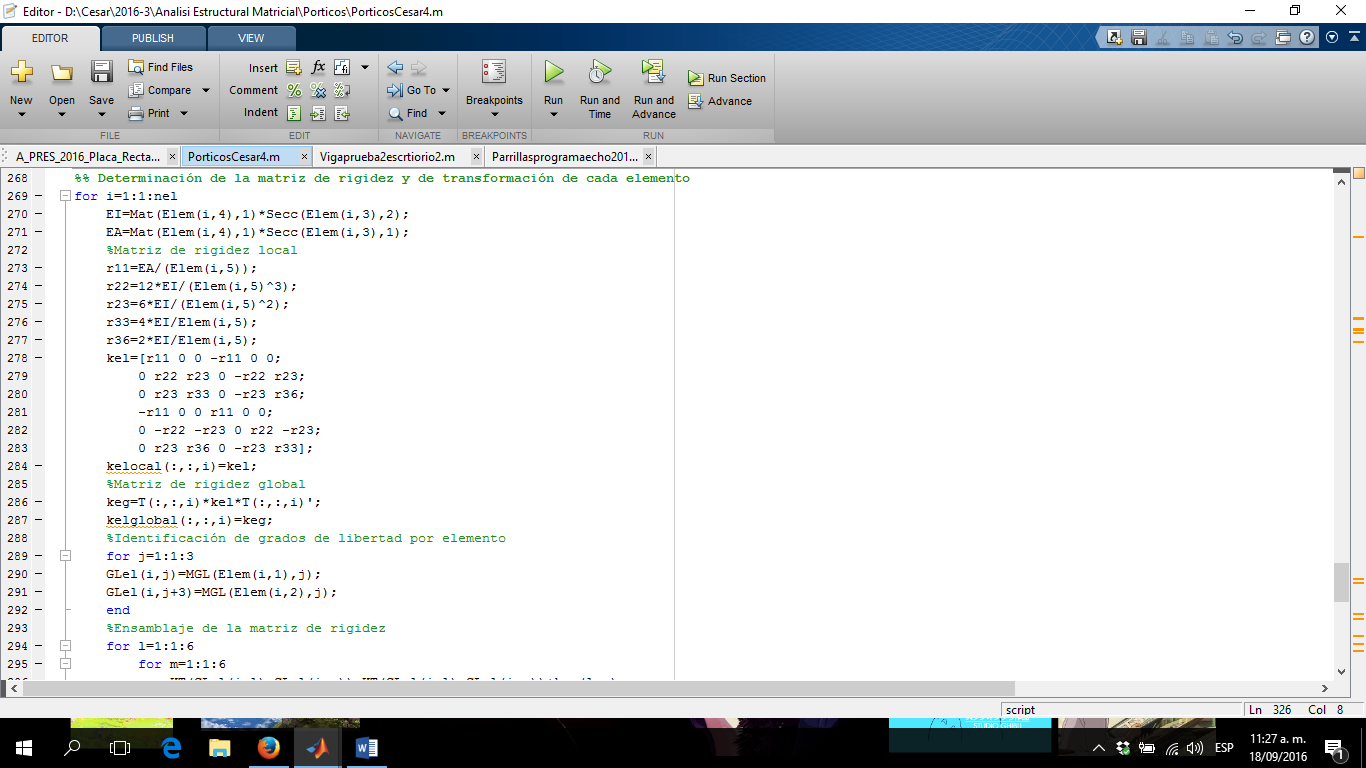


Finalmente se construye el vector de acciones fijas para nuestro caso.

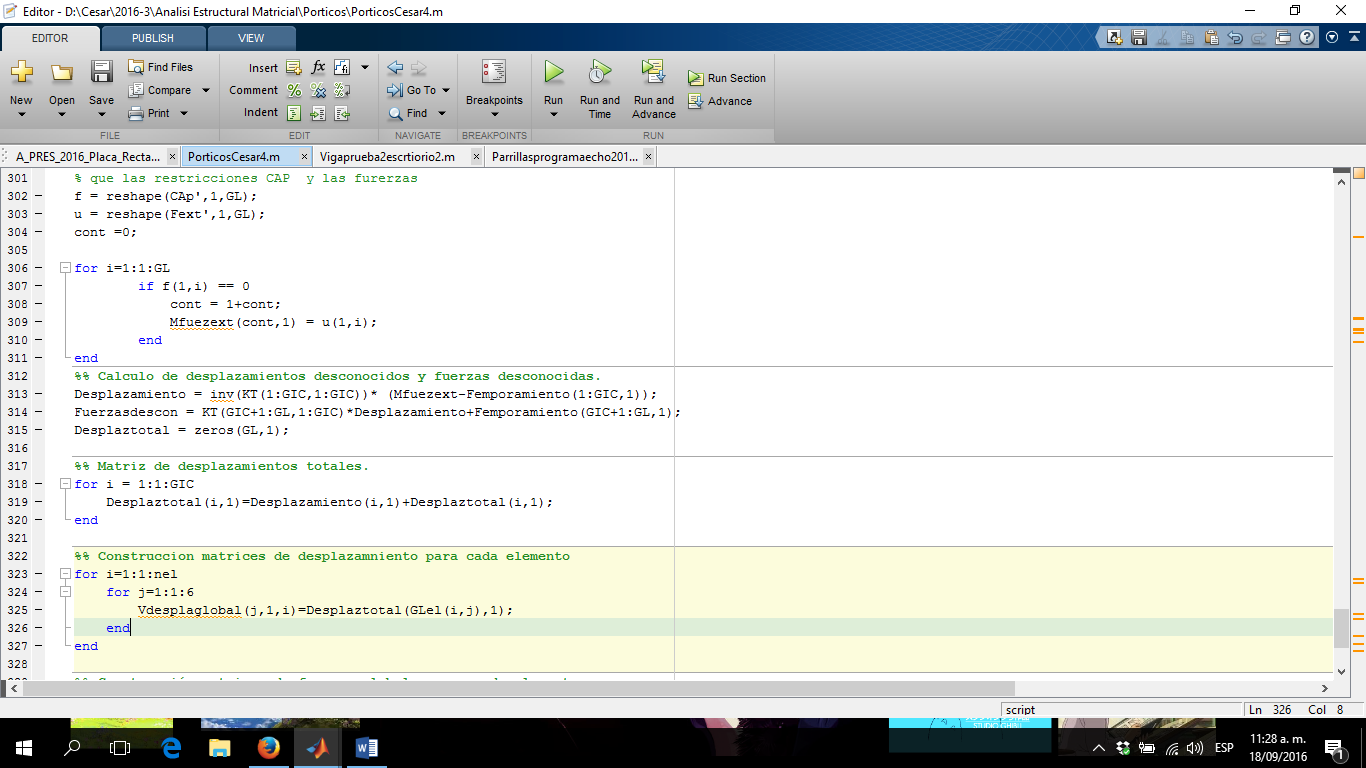


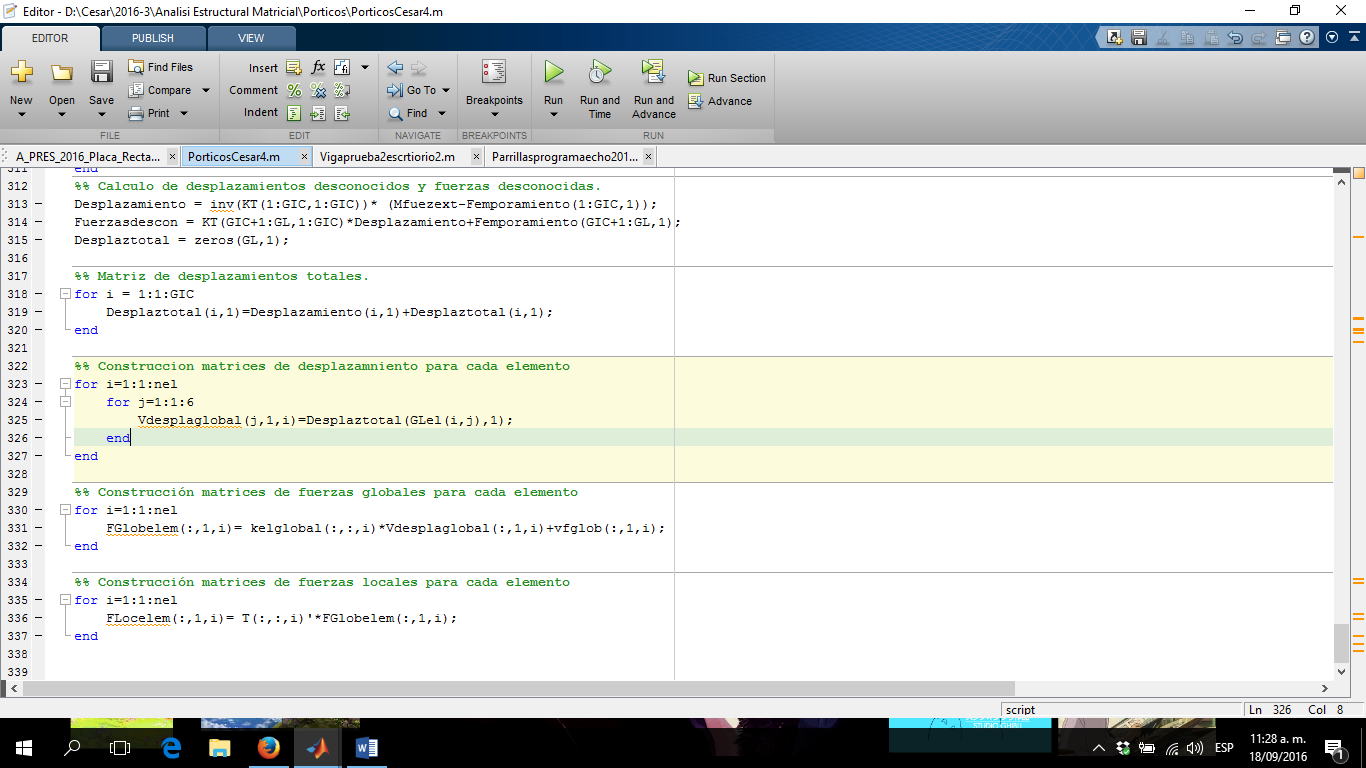


Se construye la matriz de rigidez de cada elemento y se ensambla el sistema.



Se resuelve el sistema y se muestran los resultados.



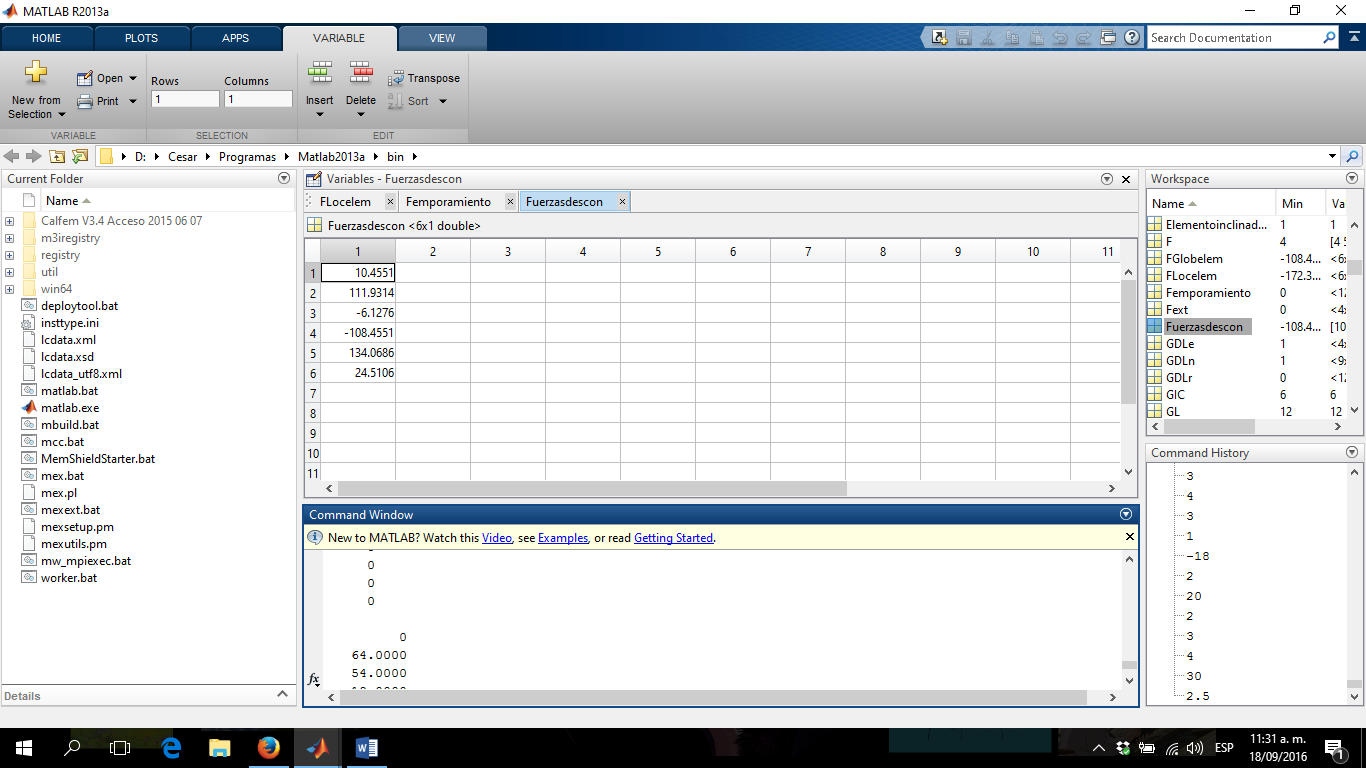


Resultados.

Fuerzas locales

|  |  |  |
| --- | --- | --- |
|  |  |  |

Reacciones apoyos.



Conclusiones.

* Es mucho mas rapido realizar analizis de esta manera que usando excel.
* Se tiene la posibilidad de tener un programa muy versatil que puede ser usado en otros temas de ingenieria civil, como la idealizasion de un lecho elastico mediante un sistema de porticos.

Codigo:

%clc;

%clear;

%clear all;

%%Datos iniciales de entrada

nnd=4;

nel=3;

nsec=3;

nmat=1;

GIC=6;

GL=3\*nnd;

nrest=GL-GIC;

nFext=2;

%% Dimensionamiento de matrices

Coord=zeros(nnd,2);

CAp=zeros(nnd,3);

Secc=zeros(nsec,2);

Mat=zeros(nmat,1);

Fext=zeros(nnd,3);

Elem=zeros(nel,5);

MGL=zeros(nnd,3);

KT=zeros(GL,GL);

GLel=zeros(nel,6);

Femporamiento=zeros(GL,1);

vfglob = zeros (6,1,nel);

vfloc = zeros (6,1,nel);

%% Entrada de datos de los nudos

for i=1:1:nnd;

nudo=i;

Coord(i,1)=input(['Coordenada x del nudo (m) ' num2str(i) '\n']);

Coord(i,2)=input(['Coordenada y del nudo (m) ' num2str(i) '\n']);

end

%% Entrada de datos de las restricciones

for ires=1:1:nrest;

i=input(['Nudo restringido:' '\n']);

j=input(['restringido en x=1, en y=2 o en giro=3?? ']);

CAp(i,j)=1;

disp (CAp);

end

%% Matriz de grados de libertad

% 1 = apoyo restringido, 0 = apoyo sin restricción.

cont1=1;

cont2=GIC+1;

for i=1:1:nnd

for j=1:1:3

if CAp(i,j)== 1;

MGL(i,j)=cont2;

cont2=cont2+1;

else

MGL(i,j)=cont1;

cont1=cont1+1;

end

end

end

disp (MGL);

%% Fuerzas externas

for cont1=1:1:nFext

i=input(['Nudo con carga' '\n']);

j=input(['Dirección de la carga en x=1, y=2, Giro=3' '\n']);

if j<3

Fext(i,j)=input(['Magnitud y sentido de la carga en el nudo (KN,KN-m' num2str(i) '\n']);

else

F=input(['Magnitud y sentido de la carga en el nudo ' num2str(i) '\n']);

Ang=input(['Ángulo de inclinación de la carga con respecto a x positivo ' num2str(i) '\n']);

Fext(i,1)=F\*cosd(Ang);

Fext(i,2)=F\*sind(Ang);

end

end

%% Secciones y materiales

for i=1:1:nsec;

Base = input(['Base del elemento (m) ' num2str(i) '\n']);

Altura = input(['Altura del elemento (m) ' num2str(i) '\n']);

Secc(i,1)= Base\*Altura;

Secc(i,2)=Base\*Altura^3/12;

end

%% Modulo de elasticidad

for i=1:1:nmat;

Mat(i,1)=input(['Módulo de elasticidad del material? (Mpa)' num2str(i) '\n'])\*10^6;

end

%% Identificación de los elementos

for i=1:1:nel

Elem(i,1)=input(['Nudo inicial del elemento ' num2str(i) '\n']);

Elem(i,2)=input(['Nudo final del elemento ' num2str(i) '\n']);

Elem(i,3)=input(['Tipo de sección del elemento ' num2str(i) '\n']);

Elem(i,4)=input(['Tipo de material del elemento ' num2str(i) '\n']);

xi=Coord(Elem(i,1),1);

yi=Coord(Elem(i,1),2);

xf=Coord(Elem(i,2),1);

yf=Coord(Elem(i,2),2);

Delx(i)=xf-xi;

Dely(i)=yf-yi;

Elem(i,5)=sqrt(Delx(i)^2+Dely(i)^2);

disp (Elem);

end

%% Matriz de Transformación

for i=1:1:nel

Cs(i)=Delx(i)/Elem(i,5);

Sn(i)=Dely(i)/Elem(i,5);

%Matriz de local a global

T(:,:,i)=[Cs(i) -Sn(i) 0 0 0 0;

Sn(i) Cs(i) 0 0 0 0;

0 0 1 0 0 0;

0 0 0 Cs(i) -Sn(i) 0;

0 0 0 Sn(i) Cs(i) 0;

0 0 0 0 0 1];

%Matrtiz de Global a local

Matrizdetransformacion(:,:,i) = T(:,:,i)';

end

%% Calculo de Fuerzas fijas

% recuerde que siempre se asume que las cargas son en sentido de la

% gravedad si queire positivas coloquele un - (Menos) a su carga en

% cualquier caso en Y, en X.

numerodeelemcaragdos = input (['¿cuantos elementos estan cargados?' '\n']);

for l = 1:1:numerodeelemcaragdos;

%Se raliza mediante proyeccion es decir siempre el vector creado es el

%de fuerzas de empotramiento perfecto globales.

queelemento = input (['¿Cual elemento es el cargado?' '\n']);

Elementoinclinadopregunta = input (['¿Elemento Inclinado? Si[1] No[2] ' '\n']);

if Elementoinclinadopregunta == 1;

proyecX = input (['Ingrese las proyecciones en X ' '\n']);

proyecY = input (['Ingrese las proyecciones en Y ' '\n']);

elseif Elementoinclinadopregunta == 2;

L = Elem(queelemento,5);

end

% Ingreso cargas del elemento

cargasinternas = input (['Ingrese el numero de cargas al interior del elemento ' num2str(queelemento) '\n' ]);

for i = 1:1:cargasinternas;

if Elementoinclinadopregunta == 1;

proyeccionausar = input (['Que proyeccion va a usar X[1] o Y[2] si es axial [3] en el elemento, ' num2str(queelemento) '\n' ]);

if proyeccionausar == 1;

L = proyecX;

elseif proyeccionausar == 2;

L = proyecY;

end

end

Tipodecarga = input ([' Carga distribuida [1]' '\n' ' puntual [2]' '\n' ' Momento [3]' '\n' ' Axial Puntual [4]' '\n' ' Axial Distribuida [5] ' '\n' ' En el elemento ' num2str(queelemento) '\n']);

if Tipodecarga == 1;

w = input('¿ Magnitud y sentido de la carga transversal distribuida (kN/m)? ');

if abs(Cs(queelemento)) == 1 || proyeccionausar == 1;

vfglob(:,1,queelemento)=[0;

-(w\*L)/2;

-(w\*L^2)/12;

0;

-(w\*L)/2;

(w\*L^2)/12] + vfglob(:,1,queelemento);

elseif abs(Cs(queelemento)) == 0 || proyeccionausar == 2;

vfglob(:,1,queelemento)=[-(w\*L)/2;

0;

-(w\*L^2)/12;

0;

-(w\*L)/2;

(w\*L^2)/12] + vfglob(:,1,queelemento);

end

vfloc(:,1,queelemento)= Matrizdetransformacion(:,:,queelemento)\*vfglob(:,1,queelemento);

elseif Tipodecarga == 2;

Puntual = input (['¿ Magnitud y sentido de la carga Puntual (kN) en el elemento ' num2str(queelemento) ' ? ''\n']);

Distanciaizq = input (['¿ Distancia de izquierda a derecha ? ' '\n']);

Distanciader = L - Distanciaizq ;

if abs(Cs(queelemento)) == 1 || proyeccionausar == 1;

vfglob(:,1,queelemento)=[0;

-Puntual\*(Distanciader^2/L^2)\*(3-2\*(Distanciader/L));

-(Puntual\*Distanciaizq\*Distanciader^2)/(L)^2;

0;

-Puntual\*(Distanciaizq^2/L^2)\*(3-2\*(Distanciaizq/L));

(Puntual\*Distanciaizq^2\*Distanciader)/(L)^2] + vfglob(:,1,queelemento);

elseif abs(Cs(queelemento)) == 0 || proyeccionausar == 2;

vfglob(:,1,queelemento)= [-Puntual\*(Distanciader^2/L^2)\*(3-2\*(Distanciader/L));

0;

-(Puntual\*Distanciaizq\*Distanciader^2)/(L)^2;

-Puntual\*(Distanciaizq^2/L^2)\*(3-2\*(Distanciaizq/L));

0;

(Puntual\*Distanciaizq^2\*Distanciader)/(L)^2] + vfglob(:,1,queelemento);

end

vfloc(:,1,queelemento) = Matrizdetransformacion(:,:,queelemento)\*vfglob(:,1,queelemento);

elseif Tipodecarga == 3;

Momentointerno = input (['¿ Magnitud y sentido del Momento en (KN-m) en el elemento ' num2str(queelemento) '? ''\n']);

Distanciaizq = input (['¿ Distancia de izquierda a derecha ? ' '\n']);

Distanciader = L - Distanciaizq ;

if abs(Cs(queelemento)) == 1 || proyeccionausar == 1;

vfglob(:,1,queelemento)=[0;

-Momentointerno\*6\*Distanciaizq\*Distanciader/L^3;

-Momentointerno\*(Distanciader/L)\*(2-3\*(Distanciader/L));

0;

Momentointerno\*6\*Distanciaizq\*Distanciader/L^3;

-Momentointerno\*(Distanciaizq/L)\*(2-3\*(Distanciaizq/L))] + vfglob(:,1,queelemento);

elseif abs(Cs(queelemento)) == 0 || proyeccionausar == 2;

vfglob(:,1,queelemento)=[-Momentointerno\*6\*Distanciaizq\*Distanciader/L^3;

0;

-Momentointerno\*(Distanciader/L)\*(2-3\*(Distanciader/L));

Momentointerno\*6\*Distanciaizq\*Distanciader/L^3;

0;

-Momentointerno\*(Distanciaizq/L)\*(2-3\*(Distanciaizq/L))] + vfglob(:,1,queelemento);

end

vfloc(:,1,queelemento) = Matrizdetransformacion(:,:,queelemento)\*vfglob(:,1,queelemento);

elseif Tipodecarga == 4;

% La carag simepre se asume que va a la derecha es decir ejel

% local X positivo.

Axialpunt= input (['¿ Magnitud de la carga axial Puntual (KN) en el elemento ' num2str(queelemento) '? ''\n']);

Distanciaizq = input (['¿ Distancia de izquierda a derecha ?' '\n']);

Distanciader = Elem(queelemento,5) - Distanciaizq ;

vfloc(:,1,queelemento)=[-Axialpunt\*Distanciader/Elem(queelemento,5);

0;

0;

-Axialpunt\*Distanciaizq/Elem(queelemento,5);

0;

0] + vfloc(:,1,queelemento);

vfglob(:,1,queelemento) = Matrizdetransformacion(:,:,queelemento)'\*vfloc(:,1,queelemento);

elseif Tipodecarga == 5;

% La carag simepre se asume que va a la derecha es decir ejel

% local X positivo.

Axidist = input (['¿ Magnitud de la carga axial distribuida (KN/m) en el elemento ' num2str(queelemento) '? ''\n']);

vfloc(:,1,queelemento)=[(-Axidist \*Elem(queelemento,5)/2);

0;

0;

(Axidist \*Elem(queelemento,5)/2);

0;

0] + vfloc(:,1,queelemento);

vfglob(:,1,queelemento) = Matrizdetransformacion(:,:,queelemento)'\*vfloc(:,1,queelemento);

end

end

end

%Creacion de vector de posiciones

%for p=1:1:nel

% for i=1:1:2;

% for l=1:1:6;

% for j=1:1:3;

% prueba(l) = MGL(Elem(p,i),j);

% end

% end

% end

%end

%% Creacion del vector de fuerzas fijas.

for i = 1: 1:nel

F(1) = MGL(Elem(i,1),1);

F(2) = MGL(Elem(i,1),2);

F(3) = MGL(Elem(i,1),3);

F(4) = MGL(Elem(i,2),1);

F(5) = MGL(Elem(i,2),2);

F(6) = MGL(Elem(i,2),3);

for k=1:1:6;

Femporamiento(F(k)) = Femporamiento(F(k))+ vfglob(k,1,i);

end

disp (Femporamiento);

end

%% Determinación de la matriz de rigidez y de transformación de cada elemento

for i=1:1:nel

EI=Mat(Elem(i,4),1)\*Secc(Elem(i,3),2);

EA=Mat(Elem(i,4),1)\*Secc(Elem(i,3),1);

%Matriz de rigidez local

r11=EA/(Elem(i,5));

r22=12\*EI/(Elem(i,5)^3);

r23=6\*EI/(Elem(i,5)^2);

r33=4\*EI/Elem(i,5);

r36=2\*EI/Elem(i,5);

kel=[r11 0 0 -r11 0 0;

0 r22 r23 0 -r22 r23;

0 r23 r33 0 -r23 r36;

-r11 0 0 r11 0 0;

0 -r22 -r23 0 r22 -r23;

0 r23 r36 0 -r23 r33];

kelocal(:,:,i)=kel;

%Matriz de rigidez global

keg=T(:,:,i)\*kel\*T(:,:,i)';

kelglobal(:,:,i)=keg;

%Identificación de grados de libertad por elemento

for j=1:1:3

GLel(i,j)=MGL(Elem(i,1),j);

GLel(i,j+3)=MGL(Elem(i,2),j);

end

%Ensamblaje de la matriz de rigidez

for l=1:1:6

for m=1:1:6

KT(GLel(i,l),GLel(i,m))=KT(GLel(i,l),GLel(i,m))+keg(l,m);

end

end

end

%% Esto se hace para poder usar la matriz de desplazamientos del elemento ya

% que las restricciones CAP y las furerzas

f = reshape(CAp',1,GL);

u = reshape(Fext',1,GL);

cont =0;

for i=1:1:GL

if f(1,i) == 0

cont = 1+cont;

Mfuezext(cont,1) = u(1,i);

end

end

%% Calculo de desplazamientos desconocidos y fuerzas desconocidas.

Desplazamiento = inv(KT(1:GIC,1:GIC))\* (Mfuezext-Femporamiento(1:GIC,1));

Fuerzasdescon = KT(GIC+1:GL,1:GIC)\*Desplazamiento+Femporamiento(GIC+1:GL,1);

Desplaztotal = zeros(GL,1);

%% Matriz de desplazamientos totales.

for i = 1:1:GIC

Desplaztotal(i,1)=Desplazamiento(i,1)+Desplaztotal(i,1);

end

%% Construccion matrices de desplazamniento para cada elemento

for i=1:1:nel

for j=1:1:6

Vdesplaglobal(j,1,i)=Desplaztotal(GLel(i,j),1);

end

end

%% Construcción matrices de fuerzas globales para cada elemento

for i=1:1:nel

FGlobelem(:,1,i)= kelglobal(:,:,i)\*Vdesplaglobal(:,1,i)+vfglob(:,1,i);

end

%% Construcción matrices de fuerzas locales para cada elemento

for i=1:1:nel

FLocelem(:,1,i)= T(:,:,i)'\*FGlobelem(:,1,i);

end