Apéndice 2: Código de programación del programa realizado en MATLAB.

El código de programación que se ha empleado para realizar el programa de los índices es el siguiente:

```
function A=indice(b)
Z=menu('¿Que indicadores de fiabilidad desea comprobar?:  ',...
   'Diseño de la red',...
   'Plan de líneas',...
   'Elaboración de horarios',...
   'Secuenciación de personal, materiales y actividades',...
   'Salir');
if Z==1
   S=menu('¿Que índice desea comprobar?:  ',...
   'Diámetro de una red',...
   'Incremento de diámetro al eliminar arcos o nodos',...
   'Perjuicio por incremento de diámetro',...
   'Resistencia de la red',...
   'Valor medio de la  resistencia de la línea',...
   'Distancia media entre pares de vértices',...
   'Incremento de la distancia entre pares de vértices',...
   'Incremento mínimo de la distancia entre pares de vértices',...
   'Incremento máximo de la distancia entre pares de vértices',...
   'Perjuicio por el incremento de la distancia entre pares de vértices',...
   'Conectividad 1',...
   'Conectividad 2',...
   'Conectividad 3',...
   'Grado de una red',...
   'Salir');
   if S==1
      % Ver si hay conexión entre nodos
      clear
      clc
      u=input ('Mete el número de nodos   ');
      disp('     ')
      disp('     ')
      b=zeros(u,u);
      disp('Empieza por el nodo más exterior de la red y acaba por el más alejado del primero')
      disp('     ')
      disp('     ')
      for i=1:u
   for j=1:u
       if i==j
           b(i,j)=0;
       else if i>j
```

```
            b(i,j)=b(j,i);
          else disp ('el nodo i es:   ')
              i
              disp('     ')
              disp ('el nodo j es:   ')
              j
              disp('     ')
              b(i,j)=input ('Si hay conexión del nodo i al j, teclea 1, si no teclea inf      ');
              clc
           end
       end
    end
end
disp('     ')
disp('Las conexiones entre los nodos son:     ')
b
% Ver las distancias entre los nodos
disp('     ')
disp('     ')
for i=1:u
    for j=1:u
       if b(i,j)>0 & b(i,j)~=inf
          if i>j
              b(i,j)=b(j,i);
           else disp ('el nodo i es:   ')
               i
               disp('     ')
               disp ('el nodo j es:   ')
               j
               disp('     ')
               b(i,j)=input ('Introduce la distancia del nodo i al nodo j      ');
               clc
            end
        end
     end
 end
W=b;
disp('     ')
disp('     ')
[m,n]=size(W);
if m~=n
   disp('La matriz de distancias debe ser cuadrada')
   return
end
L=1;  % starting vertex for shortest path
for j=1:n
   P(j)=0;
   D(j)=W(j,1);
   if D(j) < Inf
```

```
      P(j)=1;
    end
end
Lnew=L;
searchset=1:n;
while length(L) < n
    % Find search set for next vertex
    searchset=strrep(num2str(searchset),num2str(Lnew),'');
    searchset=str2num(searchset);
    [Dmin,kind]=min(D(searchset));
    k=searchset(kind);  % vertex k is next closest to vertex 1
    Lnew=k;
    L=[L Lnew];
    for j=2:n
      if all(j~=L)
        % For all j not in L find
        % D(j) = current shortest distance from vertex 1 to vertex j
        if D(j) > D(k)+W(k,j)
          D(j)=D(k)+W(k,j);
          P(j)=k;
        end
      end
    end
end
E=[];
for j=L
    if j > 1 & P(j) ~= 0
      E=[E; [P(j) j]];
    end
end
 disp('     ')
disp('El diámetro de red es:  ')
 disp('     ')
 A=max(D(L));
elseif S==2
    % Ver si hay conexión entre nodos
clear
clc
u=input ('Mete el número de nodos de la red inicial  ');
disp('     ')
disp('     ')
b=zeros(u,u);
disp('Empieza por el nodo más exterior de la red y acaba por el más alejado del
primero')
disp('     ')
disp('     ')
for i=1:u
    for j=1:u
        if i==j
```

```
          b(i,j)=0;
       else if i>j
             b(i,j)=b(j,i);
          else disp ('el nodo i es:   ')
              i
              disp('     ')
              disp ('el nodo j es:   ')
              j
              disp('     ')
              b(i,j)=input ('Si hay conexión del nodo i al j, teclea 1, si no teclea inf      ');
              clc
           end
       end
   end
end
disp('     ')
disp('Las conexiones entre los nodos son:     ')
b
% Ver las distancias entre los nodos
disp('     ')
disp('     ')
for i=1:u
   for j=1:u
      if b(i,j)>0 & b(i,j)~=inf
         if i>j
            b(i,j)=b(j,i);
          else disp ('el nodo i es:   ')
              i
              disp('     ')
              disp ('el nodo j es:   ')
              j
              disp('     ')
              b(i,j)=input ('Introduce la distancia del nodo i al nodo j      ');
              clc
           end
       end
    end
 end
W=b;
disp('     ')
disp('     ')
[m,n]=size(W);
if m~=n
  disp('La matriz de distancias debe ser cuadrada')
  return
end
L=1;  % starting vertex for shortest path
for j=1:n
  P(j)=0;
```

```
  D(j)=W(j,1);
  if D(j) < Inf
    P(j)=1;
  end
end
Lnew=L;
searchset=1:n;
while length(L) < n
  % Find search set for next vertex
  searchset=strrep(num2str(searchset),num2str(Lnew),'');
  searchset=str2num(searchset);
  [Dmin,kind]=min(D(searchset));
  k=searchset(kind);  % vertex k is next closest to vertex 1
  Lnew=k;
  L=[L Lnew];
  for j=2:n
    if all(j~=L)
      % For all j not in L find
      % D(j) = current shortest distance from vertex 1 to vertex j
      if D(j) > D(k)+W(k,j)
        D(j)=D(k)+W(k,j);
        P(j)=k;
      end
    end
  end
end
E=[];
for j=L
  if j > 1 & P(j) ~= 0
    E=[E; [P(j) j]];
  end
end
 T=max(D(L));
u=input ('Mete el número de nodos de la red final  ');
disp('     ')
disp('     ')
b=zeros(u,u);
disp('Empieza por el nodo más exterior de la red y acaba por el más alejado del
primero')
disp('     ')
disp('     ')
for i=1:u
   for j=1:u
      if i==j
         b(i,j)=0;
      else if i>j
            b(i,j)=b(j,i);
         else disp ('el nodo i es:   ')
              i
```

```
            disp('     ')
            disp ('el nodo j es:   ')
            j
            disp('     ')
            b(i,j)=input ('Si hay conexión del nodo i al j, teclea 1, si no teclea inf      ');
            clc
        end
    end
  end
end
disp('     ')
disp('Las conexiones entre los nodos son:     ')
b
% Ver las distancias entre los nodos
disp('     ')
disp('     ')
for i=1:u
   for j=1:u
      if b(i,j)>0 & b(i,j)~=inf
         if i>j
            b(i,j)=b(j,i);
         else disp ('el nodo i es:   ')
            i
            disp('     ')
            disp ('el nodo j es:   ')
            j
            disp('     ')
            b(i,j)=input ('Introduce la distancia del nodo i al nodo j      ');
            clc
         end
      end
   end
end
W=b;
disp('     ')
disp('     ')
[m,n]=size(W);
if m~=n
  disp('La matriz de distancias debe ser cuadrada')
  return
end
L=1;  % starting vertex for shortest path
for j=1:n
  P(j)=0;
  D(j)=W(j,1);
  if D(j) < Inf
    P(j)=1;
  end
end
```

```
Lnew=L;
searchset=1:n;
while length(L) < n
  % Find search set for next vertex
  searchset=strrep(num2str(searchset),num2str(Lnew),'');
  searchset=str2num(searchset);
  [Dmin,kind]=min(D(searchset));
  k=searchset(kind);  % vertex k is next closest to vertex 1
  Lnew=k;
  L=[L Lnew];
  for j=2:n
    if all(j~=L)
      % For all j not in L find
      % D(j) = current shortest distance from vertex 1 to vertex j
      if D(j) > D(k)+W(k,j)
        D(j)=D(k)+W(k,j);
        P(j)=k;
      end
    end
  end
end
E=[];
for j=L
  if j > 1 & P(j) ~= 0
    E=[E; [P(j) j]];
  end
end
 disp('     ')
 V=max(D(L));
 A=(V/T);
 disp ('El incremento del diámetro de la red en tanto por uno es:    ')
 disp('      ')
elseif S==3
   % Ver si hay conexión entre nodos
clear
clc
u=input ('Mete el número de nodos de la red inicial  ');
disp('     ')
disp('     ')
b=zeros(u,u);
disp('Empieza por el nodo más exterior de la red y acaba por el más alejado del
primero')
disp('     ')
disp('     ')
for i=1:u
   for j=1:u
      if i==j
         b(i,j)=0;
      else if i>j
```

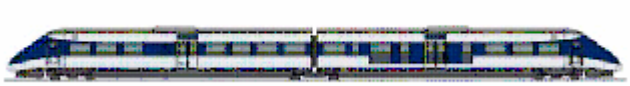
```
            b(i,j)=b(j,i);
          else disp ('el nodo i es:   ')
              i
              disp('     ')
              disp ('el nodo j es:   ')
              j
              disp('     ')
              b(i,j)=input ('Si hay conexión del nodo i al j, teclea 1, si no teclea inf      ');
              clc
           end
       end
    end
end
disp('     ')
disp('Las conexiones entre los nodos son:     ')
b
% Ver las distancias entre los nodos
disp('     ')
disp('     ')
for i=1:u
    for j=1:u
       if b(i,j)>0 & b(i,j)~=inf
          if i>j
              b(i,j)=b(j,i);
           else disp ('el nodo i es:   ')
               i
               disp('     ')
               disp ('el nodo j es:   ')
               j
               disp('     ')
               b(i,j)=input ('Introduce la distancia del nodo i al nodo j      ');
               clc
            end
        end
     end
 end
W=b;
disp('     ')
disp('     ')
[m,n]=size(W);
if m~=n
   disp('La matriz de distancias debe ser cuadrada')
   return
end
L=1;  % starting vertex for shortest path
for j=1:n
   P(j)=0;
   D(j)=W(j,1);
   if D(j) < Inf
```

```
      P(j)=1;
    end
end
Lnew=L;
searchset=1:n;
while length(L) < n
    % Find search set for next vertex
    searchset=strrep(num2str(searchset),num2str(Lnew),'');
    searchset=str2num(searchset);
    [Dmin,kind]=min(D(searchset));
    k=searchset(kind);  % vertex k is next closest to vertex 1
    Lnew=k;
    L=[L Lnew];
    for j=2:n
      if all(j~=L)
        % For all j not in L find
        % D(j) = current shortest distance from vertex 1 to vertex j
        if D(j) > D(k)+W(k,j)
          D(j)=D(k)+W(k,j);
          P(j)=k;
        end
      end
    end
end
E=[];
for j=L
    if j > 1 & P(j) ~= 0
      E=[E; [P(j) j]];
    end
end
 T=max(D(L));
u=input ('Mete el número de nodos de la red final  ');
disp('     ')
disp('     ')
b=zeros(u,u);
disp('Empieza por el nodo más exterior de la red y acaba por el más alejado del
primero')
disp('     ')
disp('     ')
for i=1:u
    for j=1:u
        if i==j
            b(i,j)=0;
        else if i>j
                b(i,j)=b(j,i);
            else disp ('el nodo i es:   ')
                i
                disp('     ')
                disp ('el nodo j es:   ')
```

```
            j
            disp('     ')
            b(i,j)=input ('Si hay conexión del nodo i al j, teclea 1, si no teclea inf      ');
            clc
          end
       end
    end
end
disp('     ')
disp('Las conexiones entre los nodos son:     ')
b
% Ver las distancias entre los nodos
disp('     ')
disp('     ')
for i=1:u
    for j=1:u
       if b(i,j)>0 & b(i,j)~=inf
          if i>j
             b(i,j)=b(j,i);
          else disp ('el nodo i es:   ')
             i
             disp('     ')
             disp ('el nodo j es:   ')
             j
             disp('     ')
             b(i,j)=input ('Introduce la distancia del nodo i al nodo j      ');
             clc
           end
       end
    end
 end
W=b;
disp('     ')
disp('     ')
[m,n]=size(W);
if m~=n
   disp('La matriz de distancias debe ser cuadrada')
   return
end
L=1;  % starting vertex for shortest path
for j=1:n
   P(j)=0;
   D(j)=W(j,1);
   if D(j) < Inf
      P(j)=1;
   end
end
Lnew=L;
searchset=1:n;
```

```
while length(L) < n
  % Find search set for next vertex
  searchset=strrep(num2str(searchset),num2str(Lnew),'');
  searchset=str2num(searchset);
  [Dmin,kind]=min(D(searchset));
  k=searchset(kind);  % vertex k is next closest to vertex 1
  Lnew=k;
  L=[L Lnew];
  for j=2:n
    if all(j~=L)
      % For all j not in L find
      % D(j) = current shortest distance from vertex 1 to vertex j
      if D(j) > D(k)+W(k,j)
        D(j)=D(k)+W(k,j);
        P(j)=k;
      end
    end
  end
end
E=[];
for j=L
  if j > 1 & P(j) ~= 0
    E=[E; [P(j) j]];
  end
end
 V=max(D(L));
 Z=(V/T);
 clc
 g=input('Introduce el número de usuarios afectados por el corte de arcos:    ')
 disp('     ')
 disp('     ')
 h=input('Introduce el tiempo medio extra invertido por los usuarios afectados para llegar a su destino en minutos:      ')
 clc
 disp('El perjuicio por incremento de diámetro es:     ')
 A=Z*g*h;
elseif S==4
   % Ver si hay conexión entre nodos
clear
clc
u=input ('Mete el numero de nodos   ');
disp('     ')
disp('     ')
b=zeros(u,u);
for i=1:u
   for j=1:u
      if i==j
         b(i,j)=0;
      else if i>j
```

```
            b(i,j)=b(j,i);
        else disp ('el nodo i es:   ')
            i
            disp('     ')
            disp ('el nodo j es:   ')
            j
            disp('     ')
            b(i,j)=input ('Si hay conexión del nodo i al j, teclea 1, si no teclea 0:      ');
            disp(' ')
            clc
         end
      end
   end
end
disp('     ')
disp('Las conexiones entre los nodos son:     ')
b
disp('    ')
disp(' ')
disp('Datos de la red modificada')
disp(' ')
b=zeros(u,u);
for i=1:u
   for j=1:u
      if i==j
         b(i,j)=0;
      else if i>j
            b(i,j)=b(j,i);
         else disp ('el nodo i es:   ')
            i
            disp('     ')
            disp ('el nodo j es:   ')
            j
            disp('     ')
            b(i,j)=input ('Si hay conexión del nodo i al j, teclea 1, si no teclea 0:      ');
            disp(' ')
            clc
         end
      end
   end
end
disp('     ')
disp('Las nuevas conexiones entre los nodos son:     ')
b
disp(' ')
disp('El valor de la resistencia de la linea es:    ')
A=(rank(b))./u;
elseif S==5
   % Ver si hay conexión entre nodos
```



```
clear
clc
u=input ('Mete el número de nodos   ');
disp('     ')
disp('     ')
b=zeros(u,u);
R=[];
E=[];
for i=1:u
   for j=1:u
      if i==j
         b(i,j)=0;
      else if i>j
            b(i,j)=b(j,i);
         else disp ('el nodo i es:   ')
            i
            disp('     ')
            disp ('el nodo j es:   ')
            j
            disp('     ')
            b(i,j)=input ('Si hay conexión del nodo i al j, teclea 1, si no teclea 0:      ');
            clc
          end
      end
   end
end
disp('Las conexiones entre los nodos son:     ')
b
disp('    ')
c=input('Escriba entre cuantos valores desea hacer la media:    ');
disp(' ')
for n=1:c
   disp('Datos de la red modificada número:   ')
   n
   disp(' ')
   b=zeros(u,u);
   for i=1:u
      for j=1:u
         if i==j
            b(i,j)=0;
         else if i>j
               b(i,j)=b(j,i);
            else disp ('el nodo i es:   ')
               i
               disp('     ')
               disp ('el nodo j es:   ')
               j
               disp('     ')
               b(i,j)=input ('Si hay conexión del nodo i al j, teclea 1, si no teclea 0:      ');
```

```
                disp(' ')
                clc
            end
        end
    end
    disp('     ')
    disp('Las nuevas conexiones entre los nodos son:     ')
    b
    disp(' ')
    disp('El valor de la resistencia de la línea es:    ')
    R=(rank(b))./u
    E=[E,R];
end
disp(' ')
disp('El valor medio de la resistencia de la línea:     ')
A=(sum(E))/n;
elseif S==6
    % Ver si hay conexión entre nodos
clear
clc
u=input ('Mete el número de nodos   ');
disp('     ')
disp('     ')
b=zeros(u,u);
disp('Empiece por el nodo al que desea hallar la distancia media a los nodos restantes')
disp('     ')
disp('     ')
for i=1:u
    for j=1:u
        if i==j
            b(i,j)=0;
        else if i>j
                b(i,j)=b(j,i);
            else disp ('el nodo i es:   ')
                i
                disp('     ')
                disp ('el nodo j es:   ')
                j
                disp('     ')
                b(i,j)=input ('Si hay conexión del nodo i al j, teclea 1, si no teclea inf      ');
                clc
            end
        end
    end
end
disp('     ')
disp('Las conexiones entre los nodos son:     ')
b
```

```
% Ver las distancias entre los nodos
disp('     ')
disp('     ')
for i=1:u
    for j=1:u
       if b(i,j)>0 & b(i,j)~=inf
          if i>j
             b(i,j)=b(j,i);
           else disp ('el nodo i es:   ')
              i
              disp('     ')
              disp ('el nodo j es:   ')
              j
              disp('     ')
              b(i,j)=input ('Introduce la distancia del nodo i al nodo j      ');
              clc
            end
        end
     end
 end
W=b;
disp('     ')
disp('     ')
[m,n]=size(W);
if m~=n
   disp('La matriz de distancias debe ser cuadrada')
   return
end
L=1;  % starting vertex for shortest path
for j=1:n
   P(j)=0;
   D(j)=W(j,1);
   if D(j) < Inf
      P(j)=1;
   end
end
Lnew=L;
searchset=1:n;
while length(L) < n
   % Find search set for next vertex
   searchset=strrep(num2str(searchset),num2str(Lnew),'');
   searchset=str2num(searchset);
   [Dmin,kind]=min(D(searchset));
   k=searchset(kind);  % vertex k is next closest to vertex 1
   Lnew=k;
   L=[L Lnew];
   for j=2:n
      if all(j~=L)
         % For all j not in L find
```

```
        % D(j) = current shortest distance from vertex 1 to vertex j
        if D(j) > D(k)+W(k,j)
           D(j)=D(k)+W(k,j);
           P(j)=k;
        end
      end
   end
end
E=[];
for j=L
   if j > 1 & P(j) ~= 0
      E=[E; [P(j) j]];
   end
end
 disp('     ')
disp('La distancia media del vértice inicial a los demás es:  ')
 disp('     ')
 [S,Q]=size(D);
 A=(sum(D))/Q;
elseif S==7
    % Ver si hay conexión entre nodos
clear
clc
u=input ('Mete el número de nodos   ');
disp('     ')
disp('     ')
b=zeros(u,u);
disp('Empiece por el nodo al que desea hallar la distancia y acabe con el otro')
disp('     ')
disp('     ')
for i=1:u
    for j=1:u
        if i==j
           b(i,j)=0;
        else if i>j
              b(i,j)=b(j,i);
           else disp ('el nodo i es:   ')
                i
                disp('     ')
                disp ('el nodo j es:   ')
                j
                disp('     ')
                b(i,j)=input ('Si hay conexión del nodo i al j, teclea 1, si no teclea inf      ');
                clc
            end
        end
    end
end
disp('     ')
```

```
disp('Las conexiones entre los nodos son:     ')
b
% Ver las distancias entre los nodos
disp('     ')
disp('     ')
for i=1:u
    for j=1:u
        if b(i,j)>0 & b(i,j)~=inf
            if i>j
                b(i,j)=b(j,i);
             else disp ('el nodo i es:   ')
                 i
                 disp('     ')
                 disp ('el nodo j es:   ')
                 j
                 disp('     ')
                 b(i,j)=input ('Introduce la distancia del nodo i al nodo j      ');
                 clc
              end
          end
      end
 end
W=b;
disp('     ')
disp('     ')
[m,n]=size(W);
if m~=n
   disp('La matriz de distancias debe ser cuadrada')
   return
end
L=1;  % starting vertex for shortest path
for j=1:n
   P(j)=0;
   D(j)=W(j,1);
   if D(j) < Inf
      P(j)=1;
   end
end
Lnew=L;
searchset=1:n;
while length(L) < n
   % Find search set for next vertex
   searchset=strrep(num2str(searchset),num2str(Lnew),'');
   searchset=str2num(searchset);
   [Dmin,kind]=min(D(searchset));
   k=searchset(kind);  % vertex k is next closest to vertex 1
   Lnew=k;
   L=[L Lnew];
   for j=2:n
```

```
      if all(j~=L)
        % For all j not in L find
        % D(j) = current shortest distance from vertex 1 to vertex j
        if D(j) > D(k)+W(k,j)
          D(j)=D(k)+W(k,j);
          P(j)=k;
        end
      end
    end
end
E=[];
for j=L
   if j > 1 & P(j) ~= 0
      E=[E; [P(j) j]];
   end
end
T=max(D(L));
 disp('     ')
c=input('Indique cuantas modificaciones desea hacer a la red:  ')
 for n=1:c
    disp('Datos de la red modificada numero:   ')
    n
    disp('     ')
b=zeros(u,u);
disp('Empiece por el nodo al que desea hallar la distancia y acabe con el otro')
disp('     ')
disp('     ')
for i=1:u
    for j=1:u
        if i==j
           b(i,j)=0;
        else if i>j
              b(i,j)=b(j,i);
           else disp ('el nodo i es:   ')
                i
                disp('     ')
                disp ('el nodo j es:   ')
                j
                disp('     ')
                b(i,j)=input ('Si hay conexión del nodo i al j, teclea 1, si no teclea inf      ');
                clc
            end
        end
    end
end
disp('     ')
disp('Las conexiones entre los nodos son:     ')
b
% Ver las distancias entre los nodos
```

```
disp('     ')
disp('     ')
for i=1:u
    for j=1:u
        if b(i,j)>0 & b(i,j)~=inf
            if i>j
                b(i,j)=b(j,i);
             else disp ('el nodo i es:   ')
                 i
                 disp('     ')
                 disp ('el nodo j es:   ')
                 j
                 disp('     ')
                 b(i,j)=input ('Introduce la distancia del nodo i al nodo j      ');
                 clc
              end
          end
      end
 end
W=b;
disp('     ')
disp('     ')
[m,n]=size(W);
if m~=n
   disp('La matriz de distancias debe ser cuadrada')
   return
end
L=1;  % starting vertex for shortest path
for j=1:n
   P(j)=0;
   D(j)=W(j,1);
   if D(j) < Inf
      P(j)=1;
   end
end
Lnew=L;
searchset=1:n;
while length(L) < n
   % Find search set for next vertex
   searchset=strrep(num2str(searchset),num2str(Lnew),'');
   searchset=str2num(searchset);
   [Dmin,kind]=min(D(searchset));
   k=searchset(kind);  % vertex k is next closest to vertex 1
   Lnew=k;
   L=[L Lnew];
   for j=2:n
      if all(j~=L)
         % For all j not in L find
         % D(j) = current shortest distance from vertex 1 to vertex j
```

```
        if D(j) > D(k)+W(k,j)
           D(j)=D(k)+W(k,j);
           P(j)=k;
        end
      end
   end
end
E=[];
for j=L
   if j > 1 & P(j) ~= 0
      E=[E; [P(j) j]];
   end
end
G=[]
G=[G max(D(L))];
end
[S,Q]=size(G);
A=(((sum(G))/Q)/T)*100;
disp('El incremento de la distancia entre los dos vértices deseados es:    ')
elseif S==8
    % Ver si hay conexión entre nodos
clear
clc
u=input ('Mete el número de nodos   ');
disp('     ')
disp('     ')
b=zeros(u,u);
disp('Empiece por el nodo al que desea hallar la distancia y acabe con el otro')
disp('     ')
disp('     ')
for i=1:u
    for j=1:u
        if i==j
           b(i,j)=0;
        else if i>j
              b(i,j)=b(j,i);
           else disp ('el nodo i es:   ')
              i
              disp('     ')
              disp ('el nodo j es:   ')
              j
              disp('     ')
              b(i,j)=input ('Si hay conexión del nodo i al j, teclea 1, si no teclea inf      ');
              clc
           end
        end
    end
end
disp('     ')
```

```
disp('Las conexiones entre los nodos son:     ')
b
% Ver las distancias entre los nodos
disp('     ')
disp('     ')
for i=1:u
    for j=1:u
        if b(i,j)>0 & b(i,j)~=inf
            if i>j
                b(i,j)=b(j,i);
             else disp ('el nodo i es:   ')
                 i
                 disp('     ')
                 disp ('el nodo j es:   ')
                 j
                 disp('     ')
                 b(i,j)=input ('Introduce la distancia del nodo i al nodo j      ');
                 clc
              end
          end
      end
 end
W=b;
disp('     ')
disp('     ')
[m,n]=size(W);
if m~=n
   disp('La matriz de distancias debe ser cuadrada')
   return
end
L=1;  % starting vertex for shortest path
for j=1:n
   P(j)=0;
   D(j)=W(j,1);
   if D(j) < Inf
      P(j)=1;
   end
end
Lnew=L;
searchset=1:n;
while length(L) < n
   % Find search set for next vertex
   searchset=strrep(num2str(searchset),num2str(Lnew),'');
   searchset=str2num(searchset);
   [Dmin,kind]=min(D(searchset));
   k=searchset(kind);  % vertex k is next closest to vertex 1
   Lnew=k;
   L=[L Lnew];
   for j=2:n
```

```
      if all(j~=L)
        % For all j not in L find
        % D(j) = current shortest distance from vertex 1 to vertex j
        if D(j) > D(k)+W(k,j)
          D(j)=D(k)+W(k,j);
          P(j)=k;
        end
      end
    end
end
E=[];
for j=L
   if j > 1 & P(j) ~= 0
      E=[E; [P(j) j]];
   end
end
T=max(D(L));
 disp('     ')
c=input('Indique cuantas modificaciones desea hacer a la red:  ')
 for n=1:c
    disp('Datos de la red modificada numero:   ')
    n
    disp('     ')
b=zeros(u,u);
disp('Empiece por el nodo al que desea hallar la distancia y acabe con el otro')
disp('     ')
disp('     ')
for i=1:u
    for j=1:u
        if i==j
           b(i,j)=0;
        else if i>j
              b(i,j)=b(j,i);
           else disp ('el nodo i es:   ')
                i
                disp('     ')
                disp ('el nodo j es:   ')
                j
                disp('     ')
                b(i,j)=input ('Si hay conexión del nodo i al j, teclea 1, si no teclea inf      ');
                clc
            end
        end
    end
end
disp('     ')
disp('Las conexiones entre los nodos son:     ')
b
% Ver las distancias entre los nodos
```

```
disp('     ')
disp('     ')
for i=1:u
    for j=1:u
        if b(i,j)>0 & b(i,j)~=inf
            if i>j
                b(i,j)=b(j,i);
             else disp ('el nodo i es:   ')
                 i
                 disp('     ')
                 disp ('el nodo j es:   ')
                 j
                 disp('     ')
                 b(i,j)=input ('Introduce la distancia del nodo i al nodo j      ');
                 clc
              end
          end
      end
 end
W=b;
disp('     ')
disp('     ')
[m,n]=size(W);
if m~=n
   disp('La matriz de distancias debe ser cuadrada')
   return
end
L=1;  % starting vertex for shortest path
for j=1:n
   P(j)=0;
   D(j)=W(j,1);
   if D(j) < Inf
      P(j)=1;
   end
end
Lnew=L;
searchset=1:n;
while length(L) < n
   % Find search set for next vertex
   searchset=strrep(num2str(searchset),num2str(Lnew),'');
   searchset=str2num(searchset);
   [Dmin,kind]=min(D(searchset));
   k=searchset(kind);  % vertex k is next closest to vertex 1
   Lnew=k;
   L=[L Lnew];
   for j=2:n
      if all(j~=L)
         % For all j not in L find
         % D(j) = current shortest distance from vertex 1 to vertex j
```

```
        if D(j) > D(k)+W(k,j)
           D(j)=D(k)+W(k,j);
           P(j)=k;
        end
      end
   end
end
E=[];
for j=L
   if j > 1 & P(j) ~= 0
      E=[E; [P(j) j]];
   end
end
G=[]
G=[G max(D(L))];
end
A=(min(G)/T);
disp('El incremento mínimo de la distancia entre los vértices deseados es:    ')
elseif S==9
    % Ver si hay conexión entre nodos
clear
clc
u=input ('Mete el número de nodos   ');
disp('     ')
disp('     ')
b=zeros(u,u);
disp('Empiece por el nodo al que desea hallar la distancia y acabe con el otro')
disp('     ')
disp('     ')
for i=1:u
    for j=1:u
        if i==j
           b(i,j)=0;
        else if i>j
              b(i,j)=b(j,i);
           else disp ('el nodo i es:   ')
                i
                disp('     ')
                disp ('el nodo j es:   ')
                j
                disp('     ')
                b(i,j)=input ('Si hay conexión del nodo i al j, teclea 1, si no teclea inf      ');
                clc
            end
        end
    end
end
disp('     ')
disp('Las conexiones entre los nodos son:     ')
```

```
b
% Ver las distancias entre los nodos
disp('     ')
disp('     ')
for i=1:u
    for j=1:u
        if b(i,j)>0 & b(i,j)~=inf
            if i>j
                b(i,j)=b(j,i);
             else disp ('el nodo i es:   ')
                 i
                 disp('     ')
                 disp ('el nodo j es:   ')
                 j
                 disp('     ')
                 b(i,j)=input ('Introduce la distancia del nodo i al nodo j      ');
                 clc
              end
         end
     end
 end
W=b;
disp('     ')
disp('     ')
[m,n]=size(W);
if m~=n
   disp('La matriz de distancias debe ser cuadrada')
   return
end
L=1;  % starting vertex for shortest path
for j=1:n
   P(j)=0;
   D(j)=W(j,1);
   if D(j) < Inf
      P(j)=1;
   end
end
Lnew=L;
searchset=1:n;
while length(L) < n
   % Find search set for next vertex
   searchset=strrep(num2str(searchset),num2str(Lnew),'');
   searchset=str2num(searchset);
   [Dmin,kind]=min(D(searchset));
   k=searchset(kind);  % vertex k is next closest to vertex 1
   Lnew=k;
   L=[L Lnew];
   for j=2:n
      if all(j~=L)
```

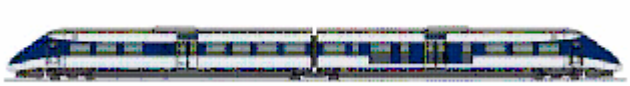
```
        % For all j not in L find
        % D(j) = current shortest distance from vertex 1 to vertex j
        if D(j) > D(k)+W(k,j)
          D(j)=D(k)+W(k,j);
          P(j)=k;
        end
      end
    end
end
E=[];
for j=L
   if j > 1 & P(j) ~= 0
     E=[E; [P(j) j]];
   end
end
T=max(D(L));
 disp('     ')
c=input('Indique cuantas modificaciones desea hacer a la red:  ')
 for n=1:c
    disp('Datos de la red modificada numero:   ')
    n
    disp('     ')
b=zeros(u,u);
disp('Empiece por el nodo al que desea hallar la distancia y acabe con el otro')
disp('     ')
disp('     ')
for i=1:u
    for j=1:u
       if i==j
          b(i,j)=0;
       else if i>j
             b(i,j)=b(j,i);
          else disp ('el nodo i es:   ')
               i
               disp('     ')
               disp ('el nodo j es:   ')
               j
               disp('     ')
               b(i,j)=input ('Si hay conexión del nodo i al j, teclea 1, si no teclea inf      ');
               clc
           end
       end
    end
end
disp('     ')
disp('Las conexiones entre los nodos son:     ')
b
% Ver las distancias entre los nodos
disp('     ')
```

```
disp('     ')
for i=1:u
   for j=1:u
      if b(i,j)>0 & b(i,j)~=inf
         if i>j
            b(i,j)=b(j,i);
          else disp ('el nodo i es:   ')
               i
               disp('     ')
               disp ('el nodo j es:   ')
               j
               disp('     ')
               b(i,j)=input ('Introduce la distancia del nodo i al nodo j      ');
               clc
           end
       end
    end
 end
W=b;
disp('     ')
disp('     ')
[m,n]=size(W);
if m~=n
  disp('La matriz de distancias debe ser cuadrada')
  return
end
L=1;  % starting vertex for shortest path
for j=1:n
  P(j)=0;
  D(j)=W(j,1);
  if D(j) < Inf
    P(j)=1;
  end
end
Lnew=L;
searchset=1:n;
while length(L) < n
  % Find search set for next vertex
  searchset=strrep(num2str(searchset),num2str(Lnew),'');
  searchset=str2num(searchset);
  [Dmin,kind]=min(D(searchset));
  k=searchset(kind);  % vertex k is next closest to vertex 1
  Lnew=k;
  L=[L Lnew];
  for j=2:n
    if all(j~=L)
      % For all j not in L find
      % D(j) = current shortest distance from vertex 1 to vertex j
      if D(j) > D(k)+W(k,j)
```

```
            D(j)=D(k)+W(k,j);
            P(j)=k;
          end
        end
      end
    end
E=[];
for j=L
  if j > 1 & P(j) ~= 0
     E=[E; [P(j) j]];
  end
end
G=[]
G=[G max(D(L))];
end
A=(max(G)/T);
disp('El incremento máximo de la distancia entre los vértices deseados es:    ')
elseif S==10
   % Ver si hay conexión entre nodos
clear
clc
u=input ('Mete el número de nodos   ');
disp('     ')
disp('     ')
b=zeros(u,u);
disp('Empiece por el nodo al que desea hallar la distancia y acabe con el otro')
disp('     ')
disp('     ')
for i=1:u
   for j=1:u
      if i==j
         b(i,j)=0;
      else if i>j
            b(i,j)=b(j,i);
         else disp ('el nodo i es:   ')
              i
              disp('     ')
              disp ('el nodo j es:   ')
              j
              disp('     ')
              b(i,j)=input ('Si hay conexión del nodo i al j, teclea 1, si no teclea inf      ');
              clc
          end
      end
   end
end
disp('     ')
disp('Las conexiones entre los nodos son:     ')
b
```

```
% Ver las distancias entre los nodos
disp('     ')
disp('     ')
for i=1:u
    for j=1:u
       if b(i,j)>0 & b(i,j)~=inf
          if i>j
             b(i,j)=b(j,i);
           else disp ('el nodo i es:   ')
              i
              disp('     ')
              disp ('el nodo j es:   ')
              j
              disp('     ')
              b(i,j)=input ('Introduce la distancia del nodo i al nodo j      ');
              clc
            end
        end
     end
 end
W=b;
disp('     ')
disp('     ')
[m,n]=size(W);
if m~=n
   disp('La matriz de distancias debe ser cuadrada')
   return
end
L=1;  % starting vertex for shortest path
for j=1:n
   P(j)=0;
   D(j)=W(j,1);
   if D(j) < Inf
      P(j)=1;
   end
end
Lnew=L;
searchset=1:n;
while length(L) < n
   % Find search set for next vertex
   searchset=strrep(num2str(searchset),num2str(Lnew),'');
   searchset=str2num(searchset);
   [Dmin,kind]=min(D(searchset));
   k=searchset(kind);  % vertex k is next closest to vertex 1
   Lnew=k;
   L=[L Lnew];
   for j=2:n
      if all(j~=L)
         % For all j not in L find
```

```
        % D(j) = current shortest distance from vertex 1 to vertex j
        if D(j) > D(k)+W(k,j)
          D(j)=D(k)+W(k,j);
          P(j)=k;
        end
      end
    end
end
E=[];
for j=L
   if j > 1 & P(j) ~= 0
     E=[E; [P(j) j]];
   end
end
T=max(D(L));
 disp('     ')
c=input('Indique cuantas modificaciones desea hacer a la red:  ')
 for n=1:c
    disp('Datos de la red modificada numero:   ')
    n
    disp('     ')
b=zeros(u,u);
disp('Empiece por el nodo al que desea hallar la distancia y acabe con el otro')
disp('     ')
disp('     ')
for i=1:u
    for j=1:u
       if i==j
          b(i,j)=0;
       else if i>j
              b(i,j)=b(j,i);
          else disp ('el nodo i es:   ')
               i
               disp('     ')
               disp ('el nodo j es:   ')
               j
               disp('     ')
               b(i,j)=input ('Si hay conexión del nodo i al j, teclea 1, si no teclea inf      ');
               clc
           end
       end
    end
end
disp('     ')
disp('Las conexiones entre los nodos son:     ')
b
% Ver las distancias entre los nodos
disp('     ')
disp('     ')
```



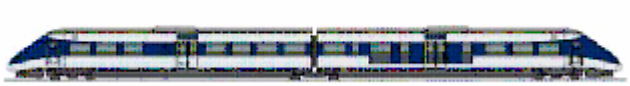
```
for i=1:u
    for j=1:u
       if b(i,j)>0 & b(i,j)~=inf
          if i>j
              b(i,j)=b(j,i);
           else disp ('el nodo i es:   ')
                i
                disp('     ')
                disp ('el nodo j es:   ')
                j
                disp('     ')
                b(i,j)=input ('Introduce la distancia del nodo i al nodo j      ');
                clc
            end
        end
     end
 end
W=b;
disp('     ')
disp('     ')
[m,n]=size(W);
if m~=n
   disp('La matriz de distancias debe ser cuadrada')
   return
end
L=1;  % starting vertex for shortest path
for j=1:n
   P(j)=0;
   D(j)=W(j,1);
   if D(j) < Inf
      P(j)=1;
   end
end
Lnew=L;
searchset=1:n;
while length(L) < n
   % Find search set for next vertex
   searchset=strrep(num2str(searchset),num2str(Lnew),'');
   searchset=str2num(searchset);
   [Dmin,kind]=min(D(searchset));
   k=searchset(kind);  % vertex k is next closest to vertex 1
   Lnew=k;
   L=[L Lnew];
   for j=2:n
      if all(j~=L)
         % For all j not in L find
         % D(j) = current shortest distance from vertex 1 to vertex j
         if D(j) > D(k)+W(k,j)
            D(j)=D(k)+W(k,j);
```

```
          P(j)=k;
        end
      end
    end
end
E=[];
for j=L
  if j > 1 & P(j) ~= 0
    E=[E; [P(j) j]];
  end
end
G=[]
G=[G max(D(L))];
end
[S,Q]=size(G);
% U=(((sum(G))/Q)/T)*;
X=input('introduce el número de personas afectadas por el corte de lineas:      ')
A=(((sum(G))/Q)/T)*X;
disp('El perjuicio por el incremento de la distancia entre los dos vertices deseados es:
')
elseif S==11
   % Ver si hay conexión entre nodos
clear
clc
u=input ('Mete el número de nodos   ');
disp('     ')
disp('     ')
b=zeros(u,u);
disp('Empiece por el nodo más exterior de la red y acabe por el más alejado del
primero')
disp('     ')
disp('     ')
for i=1:u
   for j=1:u
      if i==j
         b(i,j)=0;
      else if i>j
            b(i,j)=b(j,i);
         else disp ('el nodo i es:   ')
            i
            disp('     ')
            disp ('el nodo j es:   ')
            j
            disp('     ')
            b(i,j)=input ('Si hay conexión del nodo i al j, teclea 1, si no teclea inf      ');
            clc
          end
      end
   end
```

```
end
disp('     ')
disp('Las conexiones entre los nodos son:     ')
b
disp('     ')
z=cumsum(b);
R=[];
for j=1:u
   R=[R z(u,j)];
end
A=(sum(R))/u;
elseif S==12
   % Ver si hay conexión entre nodos
clear
clc
u=input ('Mete el número de nodos   ');
disp('     ')
disp('     ')
b=zeros(u,u);
disp('Empiece por el nodo mas exterior de la red y acabe por el mas alejado del
primero')
disp('     ')
disp('     ')
for i=1:u
   for j=1:u
      if i==j
         b(i,j)=0;
      else if i>j
            b(i,j)=b(j,i);
         else disp ('el nodo i es:   ')
            i
            disp('     ')
            disp ('el nodo j es:   ')
            j
            disp('     ')
            b(i,j)=input ('Si hay conexión del nodo i al j, teclea 1, si no teclea inf      ');
            clc
          end
      end
   end
end
disp('     ')
disp('Las conexiones entre los nodos son:     ')
b
disp('     ')
z=cumsum(b);
R=[];
for j=1:u
   R=[R z(u,j)];
```

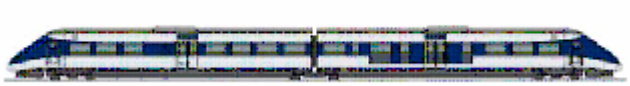
```
end
disp(' ')
F=input('Introduzca el nodo del que desea ver la conectividad:    ');
A=R(F);
clear
disp('La conectividad del nodo deseado es:   ')
elseif S==13
   % Ver si hay conexión entre nodos
clear
clc
u=input ('Mete el número de nodos   ');
disp('     ')
disp('     ')
b=zeros(u,u);
disp('Empiece por el nodo más exterior de la red y acabe por el más alejado del
primero')
disp('     ')
disp('     ')
for i=1:u
   for j=1:u
      if i==j
         b(i,j)=1;
      else if i>j
            b(i,j)=b(j,i);
         else disp ('el nodo i es:   ')
            i
            disp('     ')
            disp ('el nodo j es:   ')
            j
            disp('     ')
            b(i,j)=input ('Si hay conexión del nodo i al j, teclea 1, si no teclea inf      ');
            clc
          end
      end
   end
end
disp('     ')
disp('Las conexiones entre los nodos son:     ')
b
% Ver las probabilidades de que se interrumpa un arco
disp('     ')
disp('     ')
for i=1:u
   for j=1:u
      if b(i,j)>0 & b(i,j)~=inf
         if i>j
            b(i,j)=b(j,i);
          elseif i==j
            b(i,j)=1;
```

```
            else disp ('el nodo i es:   ')
                i
                disp('     ')
                disp ('el nodo j es:   ')
                j
                disp('     ')
                b(i,j)=input ('Introduce la probabilidad de que se interrumpa el arco ij    ');
                clc
             end
        end
    end
 end
 disp('     ')
disp('Las probabilidades de que se interrumpan los arcos son:     ')
b
disp('     ')
z=prod(b);
disp(' ')
F=input('Introduzca el nodo del que desea ver la probabilidad de quedar aislado:    ');
A=z(F);
disp('     ')
disp('La probabilidad de que el nodo deseado quede aislado es:   ')
elseif S==14
    % Max Flow Matlab Code
clear
clc
b=input('Introduce en número de arcos de la red:    ');
Tail=[];
Head=[];
for i=1:b
    disp('     ')
    disp('Datos del arco número:')
    disp(' ')
    i
    disp('  ')
    Tail(1,i)=input('Introduce el nodo de salida:    ');
    disp('   ')
    Head(1,i)=input('Introduce el nodo de llegada:    ');
    disp('   ')
end
Cap=ones(1,b);
% Tail=[1 1 2 2 3 3 4 5 5]; %input tail end node of all arcs
% Head=[2 3 3 4 4 5 6 4 6]; %input head end node of all arcs
% Cap=[4 4 2 1 9 7 8 2 4]; %input total possible capacity of each arc
Res=Cap; %residue of each capacity
n=max(Head);
Mark=zeros(1,n);
Pred=Mark;
Mark(n)=1;
```



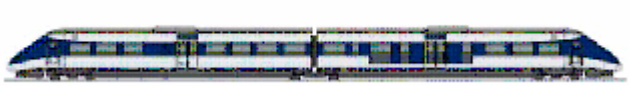
```
List=[];
TestList=[];
Path=[];
maxflow=0;
while Mark(n)==1 %finds path from source node to sink node
Mark=zeros(1,n);
Pred=Mark;
Mark(1)=1;
TestList=[]; %checks to make sure that an arc is not listed twice
List=[1]; %starting path search at source node
while (length(List)>0) & (Mark(n)==0) %loops while list is not empty and sink node is
not labeled
Path=[];
node=List(1);
TestList=[TestList node];
List=List(2:length(List));
g=find(Tail==node);
m=[];
for i=1:length(g)
if Res(g(i))>0
m=[m g(i)];
end
end
g=m; %tests for arcs that have residual capacity of zero
for i=1:length(g)
if Mark(Head(g(i)))==0
Pred(Head(g(i)))=Tail(g(i));
Mark(Head(g(i)))=1;
if length(find(TestList==Head(g(i))))==0
List=[List Head(g(i))];
end
end
end
end
if Mark(n)==1 %if path exists from source to sink then residual graph is updated
h=n;
while h>0
node=Pred(h);
k=find(Head==h);
for i=1:length(k)
if Tail(k(i))==Pred(h)
Path=[k(i) Path];
end
end
h=Pred(h);
end
delta=min(Res(Path));
Res(Path)=Res(Path)-delta;
maxflow=maxflow+delta;
```

```
A=maxflow;
end
end
clc
disp('El grado de la red es:   ')
elseif S==15
end
elseif Z==2
    S=menu('¿Que índice desea comprobar?:  ',...
    'Porcentaje de servicios operados',...
    'Utilización  de la vía',...
    'Utilización  de la vía por pasajeros',...
    'Disponibilidad de la locomotora',...
    'Ciclo de una locomotora',...
    'Margen de locomotoras',...
    'Índice de fiabilidad total de locomotoras',...
    'Cortes de suministro',...
    'Duracion media de los cortes de suministro',...
    'Número medio de caidas de tension',...
    'Duracion media de las caidas de tension',...
    'Porcentaje de la caida de tension',...
    'Salir');
if S==1
    %Porcentaje de servicios operados
clear
clc
disp('Porcentaje de servicios operados')
disp(' ')
disp(' ')
b=input('Introduzca el número de servicios realizados  ');
disp(' ')
t=input('Introduzca el número de servicios totales     ');
A=(b/t)*100;
clc
disp(' ')
disp('El porcentaje es:')
elseif S==2
    %Utilizacion de la via
clear
clc
disp('Utilizacion de la via')
disp(' ')
disp(' ')
b=input('Introduzca el numero de kilometros totales recorridos:   ');
disp(' ')
t=input('Introduzca el numero de kilometros de via totales:   ');
A=(b/t);
clc
disp(' ')
```

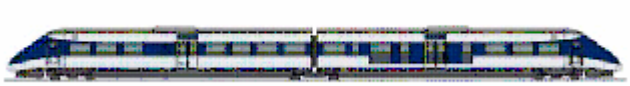


```
disp('El resultado es:   ')
elseif S==3
  %Utilizacion de la via por pasajeros
clear
clc
disp('Utilizacion de la via por pasajeros')
disp(' ')
disp(' ')
b=input('Introduzca el numero de viajes totales   ');
disp(' ')
p=0;
k=0;
for i=1:b
   disp(' ')
   disp('Datos para el viaje:   '), i
   disp(' ')
   disp(' ')
   k=k+input('Introduce el numero de kilometros en este viaje:   ');
   disp(' ')
   disp(' ')
   p=p+input('Introduce el numero de pasajeros en este viaje:   ');
end
disp(' ')
disp(' ')
t=input('Introduce el numero de kilometros totales de la via:   ');
clc
A=(p*k)/b;
disp(' ')
disp(' ')
disp('El resultado es:   ')
elseif S==4
   %    Disponibilidad de la locomotora
clear
clc
disp('Disponibilidad de la locomotora')
disp(' ')
disp(' ')
r=input('Introduzca el número de dias con la locomotora disponible:  ');
disp(' ')
b=input('Introduzca el número de dias con la locomotora averiada:     ');
A=(r/(r+b));
clc
disp(' ')
disp('La disponibilidad de la locomotora es:   ')
elseif S==5
   % Ciclo de una locomotora
clear
clc
disp('Ciclo de una locomotora')
```

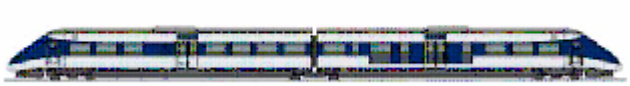
```
disp(' ')
disp(' ')
b=input('Introduzca el número de dias con la locomotora disponible:  ');
disp(' ')
t=input('Introduzca el número de dias con la locomotora averiada hasta ser arreglada:
');
A=(b+t);
clc
disp(' ')
disp('El ciclo de una locomotora es:   ')
elseif S==6
   % Margen de locomotoras
clear
clc
disp('Margen de locomotoras')
disp(' ')
disp(' ')
b=input('Introduzca el número de dias con los que quiere evaluar el indice:  ')
disp(' ')
disp(' ')
r=0;
t=0;
u=0;
v=0;
for i=1:b
   disp(' ')
   disp('Datos del dia:   ')
   i
   disp(' ')
   r=r+input('Introduzca el número de locomotoras disponibles:  ');
   disp(' ')
   t=t+input('Introduzca el número de locomotoras utilizadas:  ');
   disp(' ')
   u=u+input('Introduzca el número de locomotoras averiadas:  ');
   disp(' ')
   v=v+input('Introduzca el número de locomotoras en mantenimiento:  ');
   disp(' ')
end
A=(r-t-u-v)/b;
if A<=0
   disp('El margen es nulo. Necesita mas locomotoras ')
   disp(' ')
   A=0;
else
   disp(' ')
   disp('El margen medio de locomotoras es:   ')
end
elseif S==7
   % Índice de fiabilidad total de locomotoras
```



```
clear
clc
format long;
q=0;
p=input('Introduce la probabilidad de que una locomotora se estropee:   ');
   disp(' ')
while p>1
   p=input('Introduce la probabilidad de que una locomotora se estropee:   ');
   disp(' ')
end
b=input('Introduce el número de locomotoras que se desean analizar por estar
disponibles:     ');
   disp(' ')
P=zeros(1,(b+1));
P(1)=1;
q=0;
lista=[];
for i=1:(b+1)
   for j=1:i
      if (j-1) < 1
         q=(P(j).*(1-p))+p;
         P(j)=q;
          if i==(b+1)
            lista=[lista;(j-1) P(j)];
         end
      else
         q=(P(j).*(1-p))+((P(j-1))*p);
          P(j)=q;
         if i==(b+1)
            lista=[lista;(j-1) P(j)];
         end
      end
  end
end
A=lista;
   disp('La probabilidad de que haya locomotoras averiadas es:   ')
elseif S==8
   % Cortes de suministro
clear
clc
disp('Cortes de suministro')
disp(' ')
disp(' ')
b=input('Introduzca el número de años totales para evaluar el índice:   ');
t=0;
for i=1:b
   disp(' ')
   disp('Datos del año:   ')
   i
```

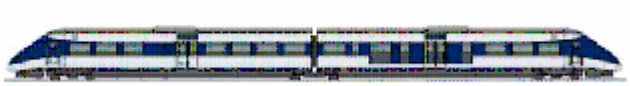


```
   disp(' ')
   t=t+input('Introduzca el número de cortes totales en el año indicado:   ');
end
A=(t/b);
clc
disp(' ')
disp('El número medio de cortes anuales es:   ')
elseif S==9
   % Duración media de los cortes de suministro
clear
clc
disp('Duración media de los cortes de suministro')
disp(' ')
disp(' ')
a=input('Introduzca el número de años totales para evaluar el índice:   ');
r=0;
for i=1:a
   t=0;
   disp(' ')
   disp('Datos del año:   ')
   i
   disp(' ')
   b=input('Introduzca el número de cortes totales en el año indicado:   ');
   for j=1:b
      disp(' ')
      disp('Datos del corte:   ')
      j
      disp(' ')
      t=t+input('Introduzca la duración del corte indicado en minutos:   ');
   end
   t=t/b
   r=r+t
end
A=(r/a);
clc
disp(' ')
disp('La duración media anual de los cortes de suministro en minutos es:   ')
elseif S==10
   % Número medio de caídas de tensión
clear
clc
disp('Número medio de caídas de tensión')
disp(' ')
disp(' ')
a=input('Introduzca el número de años totales para evaluar el índice:   ');
b=0;
for i=1:a
   disp(' ')
   disp('Datos del año:   ')
```



```
   i
   disp(' ')
   b=b+input('Introduzca el número de caídas de tensión totales en el año indicado:   ');
end
A=(b/a);
clc
disp(' ')
disp('El número medio anual de cortes es:   ')
elseif S==11
   % Duración media de las caídas de tensión
clear
clc
disp('Duración media de las caídas de tensión')
disp(' ')
disp(' ')
a=input('Introduzca el número de años totales para evaluar el índice:   ');
r=0;
for i=1:a
   t=0;
   disp(' ')
   disp('Datos del año:   ')
   i
   disp(' ')
   b=input('Introduzca el número de caídas totales en el año indicado:   ');
   for j=1:b
      disp(' ')
      disp('Datos de la caída de tensión:   ')
      j
      disp(' ')
      t=t+input('Introduzca la duración de la caída de tensóon indicada en minutos:   ');
   end
   t=t/b
   r=r+t
end
P=(r/a);
clc
disp(' ')
disp('La duración media anual de las caídas de tensión en minutos es:   ')
elseif S==12
   % Porcentaje de la caída de tensión
clear
clc
u=input('Introduce el número de caídas de tensión que se desean analizar:    ');
disp(' ')
b=0;
for i=1:u
   disp('Caída de tensión número:   ')
   i
   disp(' ')
```

```
    b=b+input('Introduce la tensión total durante la falta:   ');
end
r=input('Introduce la tensión nominal:    ');
disp('  ')
A=b/(r*u);
disp('El porcentaje de la caída de tensión es:   ')
elseif S==13
end
elseif Z==3
    S=menu('¿Que índice desea comprobar?:  ',...
    'Retrasos',...
    'Duración media del  retraso',...
    'Perjuicio del retraso',...
    'Índice de calidad del retraso permitido',...
    'Estabilidad del retraso',...
    'Retrasos por factores internos',...
    'Retrasos por factores externos',...
    'Retrasos secundarios por factores internos',...
    'Retrasos secundarios por factores externos',...
    'Salir');
if S==1
    % Retrasos
clear
clc
b=input('Introduce el número de servicios realizados:    ')
disp(' ')
r=input('Introduce el número de retrasos:    ')
disp(' ')
A=(r/b)*100;
disp('El porcentaje de retrasos es:     ')
elseif S==2
    % Duración del retraso
clear
clc
b=input('Introduce el número de retrasos:   ')
disp(' ')
r=0;
for i=1:b
    disp('Retraso número:  ')
    i
    disp(' ')
    r=r+input('Introduce los minutos de demora en este retraso:   ');
end
A=r/b;
disp('La duración media del retraso es:   ')
elseif S==3
    % Perjuicio del retraso
clear
clc
```



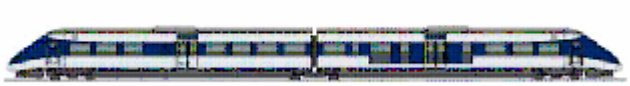
```
n=input('Introduce el número de retrasos:   ')
disp(' ')
r=0;
p=0;
for i=1:b
   disp('Retraso número:  ')
   i
   disp(' ')
   r=r+input('Introduce los minutos de demora en este retraso:   ');
   p=p+input('Introduce las personas afectadas en este retraso:   ');
end
A=(r*p)/b;
disp('El perjuicio del retraso es:   ')
elseif S==4
   % Índice de calidad del retraso permitido
clear
clc
b=input('Introduce el retraso permitido:    ');
disp(' ')
disp('La calidad del retraso permitido es (ver grafica):    ')
disp(' ')
A=1/(1+0.1*b);
lista=[];
for r=[0:100]
   i=1/(1+0.1*r);
   lista=[lista; i r];
end
plot(lista(:,2),lista(:,1),'r',b,A,'bo')
legend('Calidad del retraso permitido','Situacion del índice')
elseif S==5
   % Estabilidad del retraso
clear
clc
b=input('Introduce el número de retrasos primarios:     ');
disp(' ')
q=input('Introduce el número de retrasos secundarios:     ');
disp(' ')
A=q/b;
disp('La estabilidad del retraso es:     ')
elseif S==6
   % Retrasos por factores internos
clear
clc
e=input('Introduce el número de retrasos totales:   ');
disp(' ')
b=input('Introduce el número de retrasos debidos a factores internos:   ');
disp(' ')
A=b/e;
disp('Los retrasos por factores internos son:    ')
```

```
elseif S==7
   % Retrasos por factores externos
clear
clc
e=input('Introduce el número de retrasos totales:   ');
disp(' ')
b=input('Introduce el número de retrasos debidos a factores externos:   ');
disp(' ')
A=b/e;
disp('Los retrasos por factores externos son:    ')
elseif S==8
   % Retrasos secundarios por factores internos
clear
clc
e=input('Introduce el número de retrasos secundarios totales:   ');
disp(' ')
b=input('Introduce el número de retrasos secundarios debidos a factores internos:   ');
disp(' ')
A=b/e;
disp('Los retrasos secundarios por factores internos son:    ')
elseif S==9
   % Retrasos secundarios por factores externos
clear
clc
e=input('Introduce el número de retrasos secundarios totales:   ');
disp(' ')
b=input('Introduce el número de retrasos secundarios debidos a factores externos:   ');
disp(' ')
A=b/e;
disp('Los retrasos secundarios por factores externos son:    ')
elseif S==10
end
elseif Z==4
   S=menu('¿Que índice desea comprobar?:  ',...
   'Averías por kilómetro',...
   'Averías por kilómetro recorrido',...
   'Retrasos por avería en las vías',...
   'Retrasos por avería en el material rodante',...
   'Retrasos',...
   'Accidentes por kilómetro',...
   'Accidentes segun mantenimiento',...
   'Accidentes segun la frecuencia del tren auscultador',...
   'Accidentes por fallo humano',...
   'Accidentes por fallo informatico',...
   'Índice  bajas de personal',...
   'Duración media de las bajas',...
   'Accidentes de personal',...
   'Índice  de gravedad de las bajas',...
   'Puntualidad en el puesto de trabajo',...
```

```
   'Salir');
if S==1
   % Averias por kilometro
clear
clc
w=input('Introduce el número de kilometros de via:    ');
disp('  ')
b=input('Introduce el número de averias que ha sufrido la via:    ');
disp('  ')
A=b/w;
disp('El número de averías por kilómetro es:    ')
elseif S==2
   % Averias por kilómetro recorrido
clear
clc
u=input('Introduce el número de trenes que se desea analizar:      ');
disp(' ')
w=0;
b=0;
for i=1:u
   clc
   disp('Tren número:    ')
   i
   disp('       ')
   w=w+input('Introduce el número de kilómetros recorridos:    ');
   disp('  ')
   b=b+input('Introduce el número de averías que ha dicho tren:    ');
   disp('  ')
end
A=b/w;
disp('El número de averías por kilómetro recorrido es:    ')
elseif S==3
   %    Retrasos por avería en las vías
clear
clc
w=input('Introduce el número de minutos de retraso por averías de la vía:    ');
disp('  ')
b=input('Introduce el número de averías que ha sufrido la vía:    ');
disp('  ')
A=w/b;
disp('Los retrasos por avería en las vías son:    ')
elseif S==4
   %    Retrasos por avería en el material rodante
clear
clc
w=input('Introduce el número de minutos de retraso por averías en el material rodante:
');
disp('  ')
b=input('Introduce el número de averías que ha sufrido el material rodante:    ');
```

```
disp('  ')
A=w/b;
disp('Los retrasos por avería en el material rodante son:    ')
elseif S==5
   % Retrasos
clear
clc
w=input('Introduce el numero de retrasos debido a averias:    ');
disp('  ')
b=input('Introduce el número de servicios realizados por la compañia:    ');
disp('  ')
A=w/b;
disp('Los retrasos por avería en funcion de los servicios realizados son:    ')
elseif S==6
   % Accidentes por kilómetro
clear
clc
w=input('Introduce el número de accidentes:    ');
disp('  ')
b=input('Introduce el número de kilómetros realizados por el material movil:    ');
disp('  ')
A=w/b;
disp('Los accidentes por kilómetro son:    ')
elseif S==7
   % Accidentes segun mantenimiento
clear
clc
w=input('Introduce el número de accidentes:    ');
disp('  ')
b=input('Introduce el número de horas trabajadas en mantenimiento:    ');
disp('  ')
A=w/b;
disp('Los accidentes segun mantenimiento son:    ')
elseif S==8
   % Accidentes segun la frecuencia del tren auscultador
clear
clc
w=input('Introduce el número de accidentes:    ');
disp('  ')
b=input('Introduce el número de dias de trabajo del tren auscultador:    ');
disp('  ')
A=w/b;
disp('Los accidentes segun la frecuencia del tren auscultador son:    ')
elseif S==9
   % Accidentes por fallo humano
clear
clc
w=input('Introduce el número de accidentes:    ');
disp('  ')
```

```
b=input('Introduce el número de horas de trabajo del personal de mantenimiento:    ');
disp('  ')
A=w/b;
disp('Los accidentes por fallo humano son:    ')
elseif S==10
   % Accidentes por fallo informatico
clear
clc
w=input('Introduce el número de accidentes y fallos informaticos que podrian haberlos causado:    ');
disp('  ')
b=input('Introduce el número de horas de trabajo del personal de mantenimiento de informatica:    ');
disp('  ')
A=w/b;
disp('Los accidentes por fallo informatico son:    ')
elseif S==11
   % Indice bajas de personal
clear
clc
w=input('Introduce el número de jornadas totales:    ');
disp('  ')
b=input('Introduce el número de jornadas no trabajadas:    ');
disp('  ')
A=b/w;
disp('El índice de bajas de personal es:    ')
elseif S==12
   % Duración media de las bajas
clear
clc
w=input('Introduce el número de accidentes sufridos:    ');
disp('  ')
b=input('Introduce el número de jornadas no trabajadas por los empleados:    ');
disp('  ')
A=b/w;
disp('La duración media de las bajas de personal es:    ')
elseif S==13
   % Accidentes de personal
clear
clc
w=input('Introduce el número de accidentes sufridos:    ');
disp('  ')
b=input('Introduce el número de jornadas totales:    ');
disp('  ')
A=w/b;
disp('El índice accidentes de personal es:    ')
elseif S==14
   % Índice de gravedad de las bajas
clear
```



```
clc
w=input('Introduce el número de altas totales:    ');
disp('  ')
b=input('Introduce el número de jornadas no trabajadas por los empleados:    ');
disp('  ')
A=b/w;
disp('El índice de gravedad de las bajas es:    ')
elseif S==15
   % Puntualidad en el puesto de trabajo
clear
clc
w=input('Introduce el número de jornadas totales:    ');
disp('  ')
b=input('Introduce el número de jornadas de llegada con retraso al puesto de trabajo:
');
disp('  ')
A=b/w;
disp('La puntualidad en el puesto de trabjo es:    ')
elseif S==16
end
elseif Z==5
end
```