Escola Politècnica Superior d'Enginyeria de Manresa

UNIVERSITAT POLITÈCNICA DE CATALUNYA

Estudi del problema de la ruta òptima i la seva aplicació a una empresa de distribució de la Catalunya Central

ANNEXOS

Autor: Oriol Boix Morros
Autor: Carlos Talavera Serradilla
Director: Jordi Fortuny Santos
EPSEM
Esp: mecànica
Convocatòria: Gener 2012

ÍNDEX ANNEXOS

ANNEX A

Pàg.

1.ALGORISME PRINCIPAL AMB RECORREGUT TANCAT. 2.

1.1.Implementació 2.

1.2.Pseudocodi 14.

1.3.Explicació 26.

2.ALGORISME VARIANT AMB RECORREGUT TANCAT 28.

2.1 Implementació 28.

2.2.Pseudocodi 41.

2.3 Explicació 52.

ANNEX B

3.ALGORISME PRINCIPAL AMB RECORREGUT TANCAT. 53.

3.1.Implementació 53.

3.2.Pseudocodi 67.

3.3.Explicació 81.

4.ALGORISME VARIANT AMB RECORREGUT TANCAT 83.

4.1 Implementació 83.

4.2.Pseudocodi 96.

2.3 Explicació 109.

ANNEX A

1.ALGORISME PRINCIPAL AMB RECORREGUT TANCAT.

1.1.Implementació

Es el codi de programació fet en Emcas en el qual s'ha generat un ruta. El programa presenta primer la presa de dades i seguidament la resolució principal amb les seves subfuncions especificades a continuació**.** L'algorisme és:

algorisme EVC1TXT

const

L: **enter** =20;

fconst

Tipus

mesura: tupla

g:**booleà**;

l: **real**;

ftupla

t_nom=taula de [1...L] de **caràcters**;

t_metres= taula [1...L] de mesura;

distancies: tupla

nom: t_nom;

m: **enter**;

metres: t_metres;

ftupla

nom ciutat= taula de [1...L] de distancies;

t_lloc= taula de [1...L] de **enters**;

recorregut: tupla

lloc: t_lloc;

dist: **real**;

```
        ftupla
        unitat= taula de [1...L]de caràcter;
ftipus
var
        a: enter;
        max: real;
        w: recorregut;
        d: recorregut;
        e: distancies;
        f: nom_ciutat;
        q: mesura;
        p1: unitat;
        p2: caràcter;
        t: caràcter;
        i: enter;
        k: enter;
        c: enter;
        caux: caràcter;
fvar
        c=0;
        llegircaracter (p2);
        mentre (p2 ≠ '\n') fer
                p1[c]=p2;
                llegircaracter (p2);
                c=c+1;
        mentre
        escriuremissatge ("\n");
```

```
escriuremissatge (“Numero de ciutats per les que es passa: “);
llegirenter (a);
escirureenter (a);
escriuremissatge (“\n”);
escriuremissatge (“\n”);
llegircaracter (caux);
i=0;
llegirparaula (ent/sor f:nom_ciutat , ent/sor i:enter);
mentre (i < a) fer
        llegirparaula (ent/sor f:nom_ciutat , ent/sor i:enter);
fmentre
k=0;
mentre (k < a-1) fer
        i=k+1;
        mentre (i<a) fer
                escriuremissatge (“cost entre”);
                escriureparaula (ent/sor f:nom_ciutat, ent k:enter);
                escriuremissatge (“i”);
                escriureparaula (ent/sor f:nom_ciutat, ent k:enter);
                escriuremissatge (“ ”);
                per (t=0; t<c; t=t+1) fer
                        escriurecaracter (p1[t]);
                fper
                escriuremissatge (“\n”);
                i=i+1;
        fmentre
        k=k+1;
```

```
        fmentre


        maxim2 (ent/sor f:nom_ciutat, ent a:enter, sor max:real);

        per (k=0; k< a; k=k+1) fer

                f[k].metres[k].l=max+1;

        fper

        buscarrecorregut ( ent/sor f:nom_ciutat, ent w:recorregut, ent a:enter, ent
        d:recorregut);

        escriuremissatge (" ");

        per (t=0; t<c; t=t+1) fer

                escriurecaracter (p1[t]);

        fper

        escriuremissatge ("\n");

falgorisme
```

SUBPROGRAMES

```
accio llegirparaula (ent/sor f:nom_ciutat, ent/sor i:enter)
    var
        j:enter;
        h:carácter;
    fvar
    j=0;
    llegircaracter (h);
    si (h=0) fer
        res
    sino (h ≠ 0) fer
        f[i].nom[j]=h;
        f[i].m=f[i].m+1;
        mentre (f[i].nom[j] ≠'\n') fer
            j=j+1;
            llegircarcter(f[i].nom[j]);
            f[i].m=f[i].m+1;
        fmentre
    fsi
    i=i+1;
Faccio
```

```
Accio escriuparaula (ent/sor f:nom_ciutat, ent: i:enter)
        Var
                j:enter
        Fvar

        j=0;
        mentre (j<f[i].m-1) fer
                escriurecaracter(f[i].nom[j]);
                j=j+1
        fmentre
Facció

Acció llegirdistancia (ent/sor f:nom_ciutat, ent o:enter, ent n:enter)
        g:real;
        llegirenter (g);
        mentre (g=0) fer
                llegirreal (g);
        fmentre
        f[n].metres[o].l= g;
        f[o].metres[n].l= g;
        escriurereal (g);
Facció
```

```
Acció maxim2(ent/sor f:nom_ciutat, ent a:enter, ent/sor max:real)
    Var
        i:enter;
        j: enter;
    fvar
    max=0;
    per (i=0; i < a; i=i+1) fer
        per (j=0; j < a; j=j+1) fer
            max= f[i].metres[j].l +max;
        fper
    fper
Faccio

Acció boleacert(ent/sor f:nom_ciutat, ent a:enter)
    Var
        i,j:enter;
    Fvar
    i=0;
    mentre (i<a) fer
        j=0;
        mentre(j<a) fer
            f[i].metres[j].g =cert;
            j=j+1;
        fmentre
        i=i+1;
    fmentre
Facció
```

```
Acció Inicialitzartaula (ent/sor w:recorregut)
    var
        i:enter;
    fvar
    per (i=0; i>L; i=i+1) fer
        w.lloc[i] =0;
    fper
Facció

Acció falsprimer(ent/sor f:nom_ciutat, ent a:enter)
    Var
        z:enter;
    fvar
    z=0;
    mentre (z<a) fer
        f[z].metres[0].g=fals;
        z=z+1;
    fmentre
Facció
```

```
Acció buscardistancia (ent/sor f:nom_ciutat, ent a,y:enter, sor b:enter ent max:real)
    var
        o, min_dist:enter;
    fvar
    o=0;
    min_dist=max + 1;
    mentre (o<a) fer
        si (f[y].metres[o].g= fals) fer
        res
        sino (f[y].metres[o].g = cert) fer
            si (min_dist<= f[y].metres[o].l) fer
                res
            sino (min_dist > f[y].metres[o].l) fer
                mindist=f[y].metres[o].l;
                b=o;
            fsi
        fsi
        o=o+1
    fmentre
Faccio
```

```
Accio canviarestat (ent/sor f:nom_ciutat, ent a,y:enter)
        Var
                i:enter;
        fvar
        i=0;
        mentre (i < a) fer
                f[i].metres[y].g=fals;
                i=i+1;
        fmentre
Faccio


Acció comparartaula(ent/sor w,d:recorregut, ent a:enter)
        var
                i:enter;
        fvar
        i=0;
        si (w.dist<=d.dist) fer
        res
        sino (w.dist>d.dist) fer
                mentre (i<=a)fer
                        w.lloc[i]= d.lloc[i];
                        i=i+1;
                fmentre
                w.dist=d.dist;
        fsi
Facció
```

```
Acció escriurerecorregut(ent/sor f:nom_ciutat, ent/sor w:recorregut, ent a:enter)
    var
        i,j,x:enter
    fvar
    i=0;
    j=0;
    mentre (i<=a) fer
        x=w.lloc[i];
        mentre (j<f[x].m-1) fer
            escriurecaracter(f[x].nom[j];
            j=j+1;
        fmentre
        j=0;
        i=i+1;
        escriuremissatge (" ");
    fmentre
Faccio

acció buscarrecorregut (ent/sor f:nom_ciutat, ent w, d: recorregut, ent a:enter, ent max:real)
    var
        p, m, b, k, e, y: enter;
    fvar
    inicialitzartaula (ent/sor w:recorregut);
    boleacert (ent/sor f:nom_ciutat, ent a:enter);
    m=0;
    y=0;
```

```
w.lloc[m]=0;
m=1;
w.dist=f[y].metres[1].l;
falsprimer(ent/sor f:nom_ciutat, ent a:enter);
y=y+1;
mentre (m < a-1) fer
        buscardistancia (ent/sor f:nom_ciutat, ent a,y:enter, sor b:enter ent max:real);
        w.dist=w.dist+f[y].metres[b].l;
        w.lloc[m]=y;
        m=m+1;
        canviarestat (ent/sor f:nom_ciutat, ent a,y:enter);
        y=b;
fmentre
w.lloc[m]=y;
m=m+1;
w.dist=w.dist+f[b]metres[0].l;
w.lloc[m]=0;
k=1;
boleacert(ent/sor f:nom_ciutat, ent a:enter);
y=1;
mentre (k < a) fer
        y=k;
        Inicialitzartaula (ent/sor w:recorregut);
        d.dist=f[0].metres[k].l;
        d.lloc[0]=0;
        falsprimer (ent/sor f:nom_ciutat, ent a:enter);
        p=1;
```

mentre (p < a-1) **fer**

buscardistancia (ent/sor f:nom_ciutat, ent a,y:enter, sor b:enter ent max:real);

d.dist=d.dist+f[y].metres[b].l;

d.lloc[p]=y;

p=p+1;

canviarestat (ent/sor f:nom_ciutat, ent a,y:enter);

y=b;

fmentre

d.lloc[p]=y;

p=p+1;

d.dist=d.dist+f[b].metres[0].l;

d.lloc[p]=0;

boleacert(ent/sor f:nom_ciutat, ent a:enter);

comparartaula (ent/sor w,d:recorregut, ent a:enter);

k=k+1;

fmentre

escriuremissatge (“\n”);

escriuremissatge (“El recorregut es: “);

escriurerecorregut (ent/sor f:nom_ciutat, ent/sor w:recorregut, ent a:enter);

escriuremissatge (“\n”);

escriuremissatge (“\n”);

escriuremissatge (“El cost minim es: ”);

escriurecaracter (w.dist);

Faccio

1.2.Pseudocodi

Pel que fa el pseudocodi és:

```
#include <assert.h>
#include <stdbool.h>
#include <stdio.h>

#define L 20≠

typedef struct{
        bool g;
        float l;
}mesura;
typedef char t_nom [L];
typedef mesura t_metres [L];
typedef struct {
      t_nom nom;
        int m;
        t_metres metres;
}distancies;
typedef distancies nom_ciutat [L];
typedef int t_lloc [L];
typedef struct {
        t_lloc lloc;
      float dist;
}recorregut;
typedef char unitat[L];
```

```
void llegirparaula(nom_ciutat f,int *const i);

void escriuparaula (nom_ciutat f, int i);

void llegirdistancia (nom_ciutat f,  int o,int n);

void maxim2(nom_ciutat f, int a, float *const max);

void boleacert (nom_ciutat f, int a);

void inizialitazartaula (recorregut *const w);

void falsprimer(nom_ciutat f, int a);

void buscardistancia (nom_ciutat f, int a,int y,int *const b, float max);

void canviarestat(nom_ciutat f, int a, int y);

void comparartaula (recorregut *const w, recorregut d, int a);

void escriurerecorregut (nom_ciutat f, recorregut w, int a);

void buscarrecorregut (nom_ciutat f, recorregut w, int a, recorregut d, float max);


int main(void){
            int a;
             float max;
             recorregut w;
              recorregut d;
             distancies e;
              nom_ciutat f;
              mesura q;
              unitat p1;
              char p2;
              char t;
              int i;
              int k;
              int c;
```

```
c=0;
scanf("%c",&p2);
 while(p2!='\n'){
              p1[c]=p2;
           scanf("%c",&p2);
            c=c+1;
 }
 printf("\n");
 printf("Numero de ciutats per les que es passa:  ");
  scanf("%i",&a);
   printf("%i",a);
printf("\n");
printf("\n");
 char caux;
   scanf("%c",&caux);
  i=0;
  llegirparaula(f,&i);
  while (i < a){
          llegirparaula (f,&i);
  }
  k=0;
 while (k<a-1){
          i = k + 1;
           while (i < a){
                    printf("Cost entre ");
                    escriuparaula (f,k);
                    printf(" i ");
```

```
                        escriuparaula (f, i);
                        printf(": ");
                        llegirdistancia (f,k,i);
                        printf(" ");
                        for(t=0; t<c; ++t){
                                printf("%c",p1[t]);
                        }
                        printf("\n");
                        i=i+1;
                }
                k = k + 1;
        }
        maxim2(f,a,&max);
        for(k=0; k<a; ++k){
                f[k].metres[k].l = max +1;
        }
        buscarrecorregut (f, w, a, d, max);
        printf(" ");
        for(t=0; t<c; ++t){
                printf("%c",p1[t]);
        }
    printf("\n");
    return 0;
}

void llegirparaula(nom_ciutat f, int *const i){
    int j;
```

```
        char h;
        j=0;
        scanf("%c",&h);
        if (h==0){
        }
       else if (h!=0){
              f[*i].nom[j]=h;
              f[*i].m = f[*i].m + 1;
               while (f[*i].nom[j] !='\n'){
                     j = j + 1;
                     scanf("%c",&(f[*i].nom[j]));
                    f[*i].m = f[*i].m + 1;
               }
        }else assert(0);
       *i=*i+1;
}

void escriuparaula (nom_ciutat f, int i){
        int j;
        j=0;
       while(j < f[i].m-1){
              printf("%c", f[i].nom[j]);
             j=j+1;
        }
       }
void llegirdistancia (nom_ciutat f, int o,int n){
        float g;
```

```
        scanf("%f",&g);
        while ( g==0){
                scanf("%f",&g);
        }
        f[n].metres[o].l=g;
        f[o].metres[n].l=g;
        printf("%.2f",g);
}

void maxim2(nom_ciutat f, int a, float *const max){
        int i;
        int j;
        *max=0;
        for(i=0; i<a; i++)
                for(j=0; j<a; j++)
                        *max=f[i].metres[j].l+*max;
}
```

```
void boleacert (nom_ciutat f, int a){
        int i;
        int j;
        i=0;
        while (i<a){
                j=0;
                while(j<a){
                        f[i].metres[j].g=true;
                        j=j+1;
                }
                i=i+1;
        }
}

void inicialitazartaula (recorregut *w){
        int i;
        (*w).dist = 0;
        for(i=0; i<L; i=i+1){
                (*w).lloc[i] = 0;
        }
}
```

```
void falsprimer(nom_ciutat f, int a){
        int z;
        z=0;
        while(z<a){
                f[z].metres[0].g = false;
                z=z+1;
        }
}

void buscardistancia (nom_ciutat f, int a,int y, int *const b, float max){
        int o;
       int min_dist;
        o=0;
       min_dist=max+1;
        while(o<a){
                if(f[y].metres[o].g==false){
                }else if(f[y].metres[o].g==true){
                        if(min_dist<=f[y].metres[o].l){
                        }else if(min_dist>f[y].metres[o].l){
                                min_dist=f[y].metres[o].l;
                                *b=o;
                       }else assert(0);
                }else assert(0);
                 o=o+1;
        }
}
```

```
void canviarestat(nom_ciutat f, int a, int y){
        int i;
        i=0;
        while (i<a){
                f[i].metres[y].g=false;
                i= i + 1;
        }
}

void comparartaula (recorregut *w, recorregut d, int a){
        int i;
        i=0;
        if((*w).dist<=d.dist){
        }else if((*w).dist>d.dist){
                while(i<=a){
                        (*w).lloc[i]=d.lloc[i];
                        i=i+1;
                }
                (*w).dist=d.dist;
        }else assert(0);
}
```

```
void escriurerecorregut (nom_ciutat f, recorregut w, int a){
        int i;
        int j;
        int x;
        i=0;
        j=0;
        while(i<=a){
                x=w.lloc[i];
                while(j<f[x].m-1){
                        printf("%c",f[x].nom[j]);
                        j=j+1;
                }
                j=0;
                i=i+1;
                printf(" ");
        }
}

void buscarrecorregut ( nom_ciutat f, recorregut w,int a, recorregut d, float max){
        int p;
        int m;
        int b;
        int k;
        int e;
        int y;
        inicialitazartaula (&w);
```

```
boleacert (f,a);
 m=0;
 y=0;
 w.lloc[m]=0;
 m=1;
w.dist=f[y].metres[1].l;
 falsprimer(f,a);
y=y+1;
 while(m<a-1){
            buscardistancia (f,a,y,&b,max);
           w.dist=w.dist+f[y].metres[b].l;
          w.lloc[m]=y;
           m=m+1;
           canviarestat(f,a,y);
            y=b;
 }
w.lloc[m]=y;
m=m+1;
 w.dist=w.dist+f[b].metres[0].l;
 w.lloc[m]=0;
 k=1;
 boleacert (f,a);
 y=1;
while(k<a){
            y=k;
            inizialitazartaula (&d);
           d.dist=f[0].metres[k].l;
```

```
                d.lloc[0]=0;
                falsprimer(f,a);
                p=1;
                while(p<a-1){
                        buscardistancia (f,a,y,&b,max);
                        d.dist=d.dist+f[y].metres[b].l;
                        d.lloc[p]=y;
                        p=p+1;
                        canviarestat(f,a,y);
                        y=b;
                }
                d.lloc[p]=y;
                p=p+1;
                d.dist=d.dist+f[b].metres[0].l;
                d.lloc[p]=0;
                boleacert (f,a);
                comparartaula (&w,d,a);
                k=k+1;
        }
        printf("\n");
        printf("El recorregut es: ");
        escriurerecorregut (f,w,a);
        printf("\n");
        printf("\n");
        printf("El cost  minim es: ");
        printf("%.2f", w.dist);
}
```

1.3.Explicació

La funció del programa es que un cop entrades les dades necessàries que son:

- Unitat de mesura.
- Ciutats que tens de visitar.
- Distancia entre totes les ciutats

La manera d'introduir les dades es a traves d'omplir un full de dades creat amb el programa Excel.

El que fa el programa un ja han estat introduïdes les dades és:

1. Anar de la ciutat inicial a la ciutat 2
2. Un cop a la ciutat dos, mirar a quina ciutat el cost és menor anar-hi.
3. Repetir el pas 2 (amb les ciutats que encara no s'han visitat), fins que s'hagin visitat totes les ciutats.
4. Repetir els passos 1, 2 i 3, començant per la resta de ciutats, fins que hagis començat per totes.
5. Un cop acabat un recorregut comprar la millor opció amb la opció torbada.
6. Escriure el millor recorregut trobat amb el seu cost corresponent.

L'explicació dels subprogrames és:

Llegirparaula	La funció d'aquest subprograma és la de llegir les ciutats que tens de passar, ajudat per el mentre que hi ha al algorisme principal.
Escriureparaula	La funció d'aquest subprograma és la d'escriure les ciutats que tens de passar, ajudat per el mentre que hi ha al algorisme principal.
Llegirdistancia	La funció d'aquest subprograma és de la de agafar i guardar totes les distàncies entre les ciutats pertinents.

Maxim2	La funció d'aquest subprograma és la de crear el valor màxim, és a dir, la suma de tots el punts per tal de posar-ho a dintre de la matriu de distàncies al lloc on correspon a la distancia entre la mateixa ciutat, d'aquesta manera s'evita que des de la mateixa ciutat, pugis tornar a passar.
Boleacert	La funció d'aquest subprograma és la de un cop ja ha estat creat un recorregut posar totes les ciutats que encara les pots visitar per així poder tornar a fer un nou recorregut.
Inicialitzartaula	La funció d'aquest subprograma es posar a zero la taula en que es guarda el recorregut, és un pas necessari ja que si no el fas el programa sol assignar un valor aleatori a cada lloc i podria donar erros alhora d'executar-lo.
Falsprimer	La funció d'aquest subprograma és la de dir que no es pot anar a la primera ciutat (ciutat inicial) fins que s'hagi passat per totes les altres.
Buscardistancia	La funció d'aquest subprograma és la de comparar totes les distàncies de la ciutat actual fins a les altres ue encara no s'ha passat per poder escollir la ciutat més aprop que si pot anar per anar-hi.
Canviarestat	La funció d'aquest subprograma és negar la possibilitat de retornar a una ciutat que ja s'ha passat anteriorment.
Comparartaula	La funció d'aquest subprograma és la de comparar el millor recorregut obtingut fins el moment amb el nou recorregut que has obtingut, per tal d'observar quin del dos és més curt.

Escriurerecorregut	La funció d'aquest subprograma és un cop fet tots els recorreguts possibles, escriure el millor recorregut trobat.
Buscarrecorregut	La funció d'aquest subprograma és la de buscar una ruta que encara no s'hagi observat i guardar-la a un taula per poder comparar-la.

2.ALGORISME VARIANT AMB RECORREGUT TANCAT

2.1 Implementació

També s'ha creat un variant d'aquest programa, en el que el seu algorisme és:

algorisme EVC1

const

L: **enter** =20;

fconst

Tipus

mesura: tupla

g:**booleà**;

l: **real**;

ftupla

t_nom=taula de [1...L] de **caràcters**;

t_metres= taula [1...L] de mesura;

distancies: tupla

nom: t_nom;

m: **enter**;

metres: t_metres;

```
        ftupla
        nom ciutat= taula de [1...L] de distancies;
        t_lloc= taula de [1...L] de enters;
        recorregut: tupla
                lloc: t_lloc;
                dist: real;
        ftupla
        unitat= taula de [1...L]de caràcter;
ftipus
var
        a: enter;
        max: real;
        w: recorregut;
        d: recorregut;
        e: distancies;
        f: nom_ciutat;
        q: mesura;
        p1: unitat;
        p2: caràcter;
        t: caràcter;
        i: enter;
        k: enter;
        c: enter;
        caux: caràcter;
fvar
        c=o;
        escriuremissatge ("Quina unitat es fara servir? ");
```

```
llegircaracter (p2);
mentre (p2 ≠ '\n') fer
        p1[c]=p2;
        llegircaracter (p2);
        c=c+1;



mentre
escriuremissatge ("\n");
escriuremissatge ("Per quantes ciutats es passa?  ");
llegirenter (a);
escriuremissatge ("\n");
escriuremissatge ("Quina es la ciutat inicial? ");
llegircaracter (caux);
i=0;
llegirparaula (ent/sor f:nom_ciutat , ent/sor i:enter);
escriuremissatge ("\n");
escriuremissatge ("Escriu el nom de les ciutats restants (separades per un intró)\n");
mentre (i < a) fer
        llegirparaula (ent/sor f:nom_ciutat , ent/sor i:enter);
fmentre
k=0;
escriuremissatge ("\n");
mentre (k < a-1) fer
        i=k+1;
        mentre (i<a) fer
                escriuremissatge ("Escriu cost entre");
```

```
                    escriureparaula (ent/sor f:nom_ciutat, ent k:enter);
                    escriuremissatge ("i");
                    escriureparaula (ent/sor f:nom_ciutat, ent k:enter);
                    escriuremissatge (" ");
                    escriuremissatge ("\n");
                    i=i+1;
            fmentre
            k=k+1;
        fmentre
        maxim2 (ent/sor f:nom_ciutat, ent a:enter, sor max:real);
        per (k=0; k< a; k=k+1) fer
            f[k].metres[k].l=max+1;
        fper
        buscarrecorregut ( ent/sor f:nom_ciutat, ent w:recorregut, ent a:enter, ent
        d:recorregut);
        escriuremissatge (" ");
        per (t=0; t<c; t=t+1) fer
            escriurecaracter (p1[t]);
        fper
        escriuremissatge ("\n");
falgorisme
```

SUBPROGRAMES

accio llegirparaula (ent/sor f:nom_ciutat, ent/sor i:enter)

var

j:**enter**;

h:**carácter**;

fvar

j=0;

llegircaracter (f[i].nom[j]);

f[i].m=f[i].m+1

mentre (f[i].nom[j] ≠'\n') **fer**

j=j+1;

llegircarcter(f[i].nom[j]);

f[i].m=f[i].m+1;

fmentre

i=i+1;

Faccio

Accio escriuparaula (ent/sor f:nom_ciutat, ent: i:enter);

Var

j:**enter**

Fvar

j=0;

mentre (j<f[i].m-1) **fer**

escriurecaracter(f[i].nom[j]);

j=j+1

```
        fmentre
Facci
Acció llegirdistancia (ent/sor f:nom_ciutat, ent o:enter, ent n:enter);
        g:real;
        llegirenter (g);
        f[n].metres[o].l= g;
        f[o].metres[n].l= g;
Facció
Acció maxim2 (ent/sor f:nom_ciutat, ent a:enter, ent/sor max:real);
        Var
                i:enter;
                j: enter;
        fvar
        max=0;
        per (i=0; i < a; i=i+1) fer
                per (j=0; j < a; j=j+1) fer
                        max= f[i].metres[j].l +max;
                fper
        fper
Faccio
```

```
Acció boleacert(ent/sor f:nom_ciutat, ent a:enter)
    Var
        i,j:enter;
    Fvar
    i=0;
    mentre (i<a) fer
        j=0;
        mentre(j<a) fer
            f[i].metres[j].g =cert;
            j=j+1;
        fmentre
        i=i+1;
    fmentre
Facció

Acció Inicialitzartaula (ent/sor w:recorregut)
    var
        i:enter;
    fvar
    per (i=0; i>L; i=i+1) fer
        w.lloc[i] =0;
    fper
Facció
```

```
Acció falsprimer(ent/sor f:nom_ciutat, ent a:enter)
    Var
        z:enter;
    fvar
    z=0;
    mentre (z<a) fer
        f[z].metres[0].g=fals;
        z=z+1;
    fmentre
Facció

Acció buscardistancia (ent/sor f:nom_ciutat, ent a,y:enter, sor b:enter ent max:real);
    var
        o, min_dist:enter;
    fvar
    o=0;
    min_dist=max + 1;
    mentre (o<a) fer
        si (f[y].metres[o].g= fals) fer
        res
        sino (f[y].metres[o].g = cert) fer
            si (min_dist<= f[y].metres[o].l) fer
                res
            sino (min_dist > f[y].metres[o].l) fer
                mindist=f[y].metres[o].l;
                b=o;
```

```
                fsi
        fsi
        o=o+1
    fmentre
Faccio
Accio canviarestat (ent/sor f:nom_ciutat, ent a,y:enter)
    Var
        i:enter;
    fvar
    i=0;
    mentre (i < a) fer
        f[i].metres[y].g=fals;
        i=i+1;
    fmentre
Faccio
Acció comparartaula(ent/sor w,d:recorregut, ent a:enter)
    var
        i:enter;
    fvar
    i=0;
    si (w.dist<=d.dist) fer
    res
    sino (w.dist>d.dist) fer
        mentre (i<=a)fer
            w.lloc[i]= d.lloc[i];
            i=i+1;
        fmentre
```

```
            w.dist=d.dist;
        fsi
Facció
Acció escriurerecorregut(ent/sor f:nom_ciutat, ent/sor w:recorregut, ent a:enter)
        var
            i,j,x:enter
        fvar
        i=0;
        j=0;
        mentre (i<=a) fer
            x=w.lloc[i];
            mentre (j<f[x].m-1) fer
                escriurecaracter(f[x].nom[j];
                j=j+1;
            fmentre
            j=0;
            i=i+1;
            escriuremissatge (" ");
        fmentre
Facci
Acció buscarrecorregut (ent/sor f:nom_ciutat, ent w, d: recorregut, ent a:enter, ent max:real)
        var
            p, m, b, k, e, y: enter;
        fvar
        inicialitzartaula (ent/sor w:recorregut);
        boleacert (ent/sor f:nom_ciutat, ent a:enter);
        m=0;
```

```
y=0;
w.lloc[m]=0;
m=1;
w.dist=f[y].metres[1].l;
falsprimer(ent/sor f:nom_ciutat, ent a:enter);
y=y+1;
mentre (m < a-1) fer
        buscardistancia (ent/sor f:nom_ciutat, ent a,y:enter, sor b:enter ent max:real);
        w.dist=w.dist+f[y].metres[b].l;
        w.lloc[m]=y;
        m=m+1;
        canviarestat (ent/sor f:nom_ciutat, ent a,y:enter);
        y=b;
fmentre
w.lloc[m]=y;
m=m+1;
w.dist=w.dist+f[b]metres[0].l;
w.lloc[m]=0;
k=1;
boleacert(ent/sor f:nom_ciutat, ent a:enter);
y=1;
mentre (k < a) fer
        y=k;
        Inicialitzartaula (ent/sor w:recorregut);
        d.dist=f[0].metres[k].l;
        d.lloc[0]=0;
        falsprimer (ent/sor f:nom_ciutat, ent a:enter);
```

```
p=1;
mentre (p < a-1) fer
    buscardistancia (ent/sor f:nom_ciutat, ent a,y:enter, sor b:enter ent max:real);
    d.dist=d.dist+f[y].metres[b].l;
    d.lloc[p]=y;
    p=p+1;
    canviarestat (ent/sor f:nom_ciutat, ent a,y:enter);
    y=b;
fmentre
d.lloc[p]=y;
p=p+1;
d.dist=d.dist+f[b].metres[0].l;
d.lloc[p]=0;
boleacert(ent/sor f:nom_ciutat, ent a:enter);
comparartaula (ent/sor w,d:recorregut, ent a:enter);
k=k+1;
fmentre
escriuremissatge ("\n");
escriuremissatge ("El recorregut es: ");
escriurerecorregut (ent/sor f:nom_ciutat, ent/sor w:recorregut, ent a:enter);
escriuremissatge ("\n");
escriuremissatge ("\n");
escriuremissatge ("El cost minim es: ");
escriurecaracter (w.dist);
```

Faccio

2.2 Pseudocodi

Pel que fa el pseudocodi és:

```
#include <assert.h>

#include <stdbool.h>

#include <stdio.h>

#define L 20

typedef struct{

        bool g;

        float l;

}mesura;

typedef char t_nom [L];

typedef mesura t_metres [L];

typedef struct {

        t_nom nom;

        int m;

  t_metres metres;

}distancies;

typedef distancies nom_ciutat [L];

typedef int t_lloc [L];

typedef struct {

        t_lloc lloc;

        float dist;

}recorregut;

typedef char unitat[L];
```

```
void llegirparaula(nom_ciutat f,int *const i);

void escriuparaula (nom_ciutat f, int i);

void llegirdistancia (nom_ciutat f,  int o,int n);

void maxim2(nom_ciutat f, int a, float *const max);

void boleacert (nom_ciutat f, int a);

void inizialitazartaula (recorregut *const w);

void falsprimer(nom_ciutat f, int a);

void buscardistancia (nom_ciutat f, int a,int y,int *const b, float max);

void canviarestat(nom_ciutat f, int a, int y);

void comparartaula (recorregut *const w, recorregut d, int a);

void escriurerecorregut (nom_ciutat f, recorregut w, int a);

void buscarrecorregut (nom_ciutat f, recorregut w, int a, recorregut d, float max);

int main(void){
        int a;
        float max;
        recorregut w;
        recorregut d;
        distancies e;
        nom_ciutat f;
        mesura q;
        unitat p1;
        char p2;
        char t;
        int i;
        int k;
```

```
 int c;
c=0;
printf("Quina unitat es fara servir? ");
 scanf("%c",&p2);
 while(p2!='\n'){
              p1[c]=p2;
              scanf("%c",&p2);
            c=c+1;
 }
 printf("\n");
printf("Per quantes ciutats es passa?  ");
 scanf("%i",&a);
  printf("\n");
 printf("Quina es la ciutat inicial? ");
char caux; scanf("%c",&caux);
 i=0;
 llegirparaula(f,&i);
  printf("\n");
   printf("Escriu el nom de les ciutats restants (separades per un intro) \n");
   while (i < a){
            llegirparaula (f,&i);
   }
  k=0;
   printf("\n");
   while (k<a-1){
           i = k + 1;
            while (i < a){
```

```
                    printf("Escriu el cost  entre ");
                    escriuparaula (f,k);
                    printf(" i ");
                    escriuparaula (f, i);
                    printf(": ");
                    llegirdistancia (f,k,i);
                    printf("\n");
                    i=i+1;
                }
                k = k + 1;
            }
            maxim2(f,a,&max);
           for(k=0; k<a; ++k){
                f[k].metres[k].l = max +1;
             }
            buscarrecorregut (f, w, a, d, max);
            printf(" ");
            for(t=0; t<c; ++t){
                printf("%c",p1[t]);
           }
           printf("\n");
            return 0;
}
```

```
void llegirparaula(nom_ciutat f, int *const i){
            int j;
             char h;
            j=0;
             scanf("%c", &f[*i].nom[j]);
            f[*i].m = f[*i].m + 1;
             while (f[*i].nom[j] !='\n'){
                        j = j + 1;
                       scanf("%c",&(f[*i].nom[j]));
                       f[*i].m = f[*i].m + 1;
             }
           *i=*i+1;
}
void escriuparaula (nom_ciutat f, int i){
             int j;
             j=0;
             while(j < f[i].m-1){
                        printf("%c", f[i].nom[j]);
             j=j+1;
             }
}
 void llegirdistancia (nom_ciutat f, int o,int n){
             float g;
            scanf("%f",&g);
            f[n].metres[o].l=g;
```

```
            f[o].metres[n].l=g;
}
void maxim2(nom_ciutat f, int a, float *const max){
            int i;
            int j;
             *max=0;
             for(i=0; i<a; i++)
                          for(j=0; j<a; j++)
                                       *max=f[i].metres[j].l+*max;
}
void boleacert (nom_ciutat f, int a){
             int i;
             int j;
             i=0;
             while (i<a){
                          j=0;
                           while(j<a){
                                        f[i].metres[j].g=true;
                                      j=j+1;
                                    }
              i=i+1;
             }         }
void inizialitazartaula (recorregut *w){
            int i;
             (*w).dist = 0;
             for(i=0; i<L; i=i+1){
                          (*w).lloc[i] = 0;
```

```
        }
}


void falsprimer(nom_ciutat f, int a){
        int z;
        z=0;
        while(z<a){
                f[z].metres[0].g = false;
                z=z+1;
        }
}
void buscardistancia (nom_ciutat f, int a,int y, int *const b, float max){
        int o;
        int min_dist;
        o=0;
        min_dist=max+1;
        while(o<a){
                if(f[y].metres[o].g==false){
                }else if(f[y].metres[o].g==true){
                        if(min_dist<=f[y].metres[o].l){
                        }else if(min_dist>f[y].metres[o].l){
                                min_dist=f[y].metres[o].l;
                                *b=o;
                        }else assert(0);
                }else assert(0);
                o=o+1;
        }
```

```
}
void canviarestat(nom_ciutat f, int a, int y){
        int i;
        i=0;
        while (i<a){
                f[i].metres[y].g=false;
                i= i + 1;
        }
}
void comparartaula (recorregut *w, recorregut d, int a){
        int i;
        i=0;
        if((*w).dist<=d.dist){
        }else if((*w).dist>d.dist){
                while(i<=a){
                        (*w).lloc[i]=d.lloc[i];
                        i=i+1;
                }
               (*w).dist=d.dist;
        }else assert(0);
}
```

```
void escriurerecorregut (nom_ciutat f, recorregut w, int a){
        int i;
        int j;
        int x;
        i=0;
        j=0;
        while(i<=a){
                x=w.lloc[i];
                while(j<f[x].m-1){
                        printf("%c",f[x].nom[j]);
                         j=j+1;
                }
               j=0;
              i=i+1;
               printf(" ");
        }
}
void buscarrecorregut ( nom_ciutat f, recorregut w,int a, recorregut d, float max){
        int p;
        int m;
        int b;
        int k;
        int e;
        int y;
        inizialitazartaula (&w);
```

```
 boleacert (f,a);
 m=0;
 y=0;
 w.lloc[m]=0;
 m=1;
 w.dist=f[y].metres[1].l;
 falsprimer(f,a);
 y=y+1;
while(m<a-1){
        buscardistancia (f,a,y,&b,max);
         w.dist=w.dist+f[y].metres[b].l;
        w.lloc[m]=y;
         m=m+1;
         canviarestat(f,a,y);
         y=b;
 }
w.lloc[m]=y;
 m=m+1;
 w.dist=w.dist+f[b].metres[0].l;
 w.lloc[m]=0;
 k=1;
 boleacert (f,a);
 y=1;
 while(k<a){
        y=k;
        inizialitazartaula (&d);
         d.dist=f[0].metres[k].l;
```

```
                    d.lloc[0]=0;
                    falsprimer(f,a);
                     p=1;
                    while(p<a-1){
                            buscardistancia (f,a,y,&b,max);
                            d.dist=d.dist+f[y].metres[b].l;
                            d.lloc[p]=y;
                             p=p+1;
                             canviarestat(f,a,y);
                            y=b;
                    }
                    d.lloc[p]=y;
                     p=p+1;
                    d.dist=d.dist+f[b].metres[0].l;
                    d.lloc[p]=0;
                     boleacert (f,a);
                     comparartaula (&w,d,a);
                    k=k+1;
            }
            printf("\n");
           printf("El recorregut es: ");
            escriurerecorregut (f,w,a);
            printf("\n");
            printf("\n");
          printf("El cost  minim es: ");
           printf("%.2f", w.dist);
}
```

2.3 Explicació

La diferencia entre el programa original i la seva variant és en la manera d'obtenir les dades ja que el programa original (evc1txt) les dades s'obtenen a traves d'un full Excel, en canvi en la variació les dades s'introdueixen quan el programa te les demana, un cop ja l'has començat a executar a dintre de el MS-DOS.

ANNEX B:

3.ALGORISME PRINCIPAL AMB RECORREGUT OBERT.

3.1.Implementació

Es el codi de programació fet en Emacs en el qual s'ha generat un ruta. El programa presenta primer la presa de dades i seguidament la resolució principal amb les seves subfuncions especificades a continuació**.** L'algorisme és:

algorisme EVC2TXT

const

L: **enter** =20;

fconst

Tipus

mesura: tupla

g:**booleà**;

l: **real**;

ftupla

t_nom=taula de [1...L] de **caràcters**;

t_metres= taula [1...L] de mesura;

distancies: tupla

nom: t_nom;

m: **enter**;

metres: t_metres;

ftupla

nom ciutat= taula de [1...L] de distancies;

t_lloc= taula de [1...L] de **enters**;

recorregut: tupla

lloc: t_lloc;

dist: **real**;

```
        ftupla
        unitat= taula de [1...L]de caràcter;
ftipus
var
        a: enter;
        max: real;
        w: recorregut;
        d: recorregut;
        e: distancies;
        f: nom_ciutat;
        q: mesura;
        p1: unitat;
        p2: caràcter;
        t: caràcter;
        i: enter;
        k: enter;
        c: enter;
        caux: caràcter;
fvar
        c=0;
        llegircaracter (p2);
        mentre (p2 ≠ '\n') fer
                p1[c]=p2;
                llegircaracter (p2);
                c=c+1;
        fmentre
        escriuremissatge ("\n");
```

```
escriuremissatge (“Numero de ciutats per les que es passa: “);
llegirenter (a);
escirureenter (a);
escriuremissatge (“\n”);
escriuremissatge (“\n”);
llegircaracter (caux);
i=0;
llegirparaula (ent/sor f:nom_ciutat , ent/sor i:enter);
llegirparaula (ent/sor f:nom_ciutat , ent/sor i:enter);
mentre (i < a) fer
        llegirparaula (ent/sor f:nom_ciutat , ent/sor i:enter);
fmentre
k=0;
escriuremissatge (“\n”);
mentre (k < a-1) fer
        i=k+1;
        mentre (i<a) fer
                escriuremissatge (“cost entre”);
                escriureparaula (ent/sor f:nom_ciutat, ent k:enter);
                escriuremissatge (“i”);
                escriureparaula (ent/sor f:nom_ciutat, ent k:enter);
                escriuremissatge (“ :”);
                llegirdistancia()
                per (t=0; t<c; t=t+1) fer
                        escriurecaracter (p1[t]);
                fper
                escriuremissatge (“\n”);
```

```
                i=i+1;
            fmentre
            k=k+1;
        fmentre

        maxim2 (ent/sor f:nom_ciutat, ent a:enter, sor max:real);
        per (k=0; k< a; k=k+1) fer
            f[k].metres[k].l=max+1;
        fper
        buscarrecorregut ( ent/sor f:nom_ciutat, ent w:recorregut, ent a:enter, ent
        d:recorregut);
        escriuremissatge (" ");
        per (t=0; t<c; t=t+1) fer
            escriurecaracter (p1[t]);
        fper
        escriuremissatge ("\n");
falgorisme
```

SUBPROGRAMES

accio llegirparaula (ent/sor f:nom_ciutat, ent/sor i:**enter**)

var

j:**enter**;

h:**carácter**;

fvar

j=0;

llegircaracter (h);

si (h=0) **fer**

res

sino (h ≠ 0) **fer**

f[i].nom[j]=h;

f[i].m=f[i].m+1;

mentre (f[i].nom[j] ≠'\n') **fer**

j=j+1;

llegircarcter(f[i].nom[j]);

f[i].m=f[i].m+1;

fmentre

fsi

i=i+1;

Faccio

```
Accio escriuparaula (ent/sor f:nom_ciutat, ent: i:enter)
    Var
        j:enter
    Fvar

    j=0;
    mentre (j<f[i].m-1) fer
        escriurecaracter(f[i].nom[j]);
        j=j+1
    fmentre
Facció

Acció llegirdistancia (ent/sor f:nom_ciutat, ent o:enter, ent n:enter)
    g:real;
    llegirenter (g);
    mentre (g=0) fer
        llegirreal (g);
    fmentre
    f[n].metres[o].l= g;
    f[o].metres[n].l= g;
    escriurereal (g);
Facció
```

```
Acció maxim2(ent/sor f:nom_ciutat, ent a:enter, ent/sor max:real)
    Var
        i:enter;
        j: enter;
    fvar
    max=0;
    per (i=0; i < a; i=i+1) fer
        per (j=0; j < a; j=j+1) fer
            max= f[i].metres[j].l +max;
        fper
    fper
Faccio

Acció boleacert(ent/sor f:nom_ciutat, ent a:enter)
    Var
        i,j:enter;
    Fvar
    i=0;
    mentre (i<a) fer
        j=0;
        mentre(j<a) fer
            f[i].metres[j].g =cert;
            j=j+1;
        fmentre
        i=i+1;
    fmentre
Facció
```

```
Acció Inicialitzartaula (ent/sor w:recorregut)
        var
                i:enter;
        fvar
        (w).dist=0;
        per (i=0; i>L; i=i+1) fer
                w.lloc[i] =0;
        fper
Facció


Acció falsprimer(ent/sor f:nom_ciutat, ent a:enter)
        Var
                z:enter;
        fvar
        z=0;
        mentre (z<a) fer
                f[z].metres[0].g=fals;
                z=z+1;
        fmentre
Facció
```

```
Acció falsultim(ent/sor f:nom_ciutat, ent a:enter)
    Var
        z:enter;
    fvar
    z=0;
    mentre (z<a) fer
        f[z].metres[1].g=fals;
        f[1].metres[z].g=fals;
        z=z+1;
    fmentre
Facció

Acció buscardistancia (ent/sor f:nom_ciutat, ent a,y:enter, sor b:enter ent max:real)
    var
        o, min_dist:enter;
    fvar
    o=0;
    min_dist=max + 1;
    mentre (o<a) fer
        si (f[y].metres[o].g= fals) fer
        res
        sino (f[y].metres[o].g = cert) fer
            si (min_dist<= f[y].metres[o].l) fer
                res
            sino (min_dist > f[y].metres[o].l) fer
                mindist=f[y].metres[o].l;
```

```
                    b=o;
                fsi
            fsi
            o=o+1
        fmentre
Faccio

Accio canviarestat (ent/sor f:nom_ciutat, ent a,y:enter)
        Var
            i:enter;
        fvar
        i=0;
        mentre (i < a) fer
            f[i].metres[y].g=fals;
            i=i+1;
        fmentre
Faccio

Acció comparartaula(ent/sor w,d:recorregut, ent a:enter)
        var
            i:enter;
        fvar
        i=0;
        si (w.dist<=d.dist) fer
        res
        sino (w.dist>d.dist) fer
            mentre (i<=a)fer
```

```
                w.lloc[i]= d.lloc[i];
                i=i+1;
            fmentre
            w.dist=d.dist;
        fsi
Facció

Acció escriurerecorregut(ent/sor f:nom_ciutat, ent/sor w:recorregut, ent a:enter)
        var
            i,j,x:enter
        fvar
        i=0;
        j=0;
        mentre (i<=a-1) fer
            x=w.lloc[i];
            mentre (j<f[x].m-1) fer
                escriurecaracter(f[x].nom[j];
                j=j+1;
            fmentre
            j=0;
            i=i+1;
            escriuremissatge (" ");
        fmentre
Faccio
```

Acció buscarrecorregut (ent/sor f:nom_ciutat, ent w, d: recorregut, ent a:**enter**, ent max:**real**)

var

p, m, b, k, e, y: **enter;**

fvar

inicialitzartaula (ent/sor w:recorregut);

boleacert (ent/sor f:nom_ciutat, ent a:enter);

m=0;

y=0;

w.lloc[m]=0;

m=1;

w.dist=f[y].metres[2].l;

falsprimer(ent/sor f:nom_ciutat, ent a:enter);

falsultim(ent/sor f:nom_ciutat, ent a:enter);

y=2;

mentre (m < a-2) **fer**

buscardistancia (ent/sor f:nom_ciutat, ent a,y:enter, sor b:enter ent max:real);

w.dist=w.dist+f[y].metres[b].l;

w.lloc[m]=y;

m=m+1;

canviarestat (ent/sor f:nom_ciutat, ent a,y:enter);

y=b;

fmentre

w.lloc[m]=y;

m=m+1;

w.dist=w.dist+f[b]metres[1].l;

w.lloc[m]=1;

k=2;

boleacert(ent/sor f:nom_ciutat, ent a:enter);

y=1;

mentre (k < a) **fer**

y=k;

Inicialitzartaula (ent/sor w:recorregut);

d.dist=f[0].metres[k].l;

d.lloc[0]=0;

falsprimer (ent/sor f:nom_ciutat, ent a:enter);

falsultim(ent/sor f:nom_ciutat, ent a:enter);

p=1;

mentre (p < a-2) **fer**

buscardistancia (ent/sor f:nom_ciutat, ent a,y:enter, sor b:enter ent max:real);

d.dist=d.dist+f[y].metres[b].l;

d.lloc[p]=y;

p=p+1;

canviarestat (ent/sor f:nom_ciutat, ent a,y:enter);

y=b;

fmentre

d.lloc[p]=y;

p=p+1;

d.dist=d.dist+f[b].metres[1].l;

d.lloc[p]=1;

boleacert(ent/sor f:nom_ciutat, ent a:enter);

comparartaula (ent/sor w,d:recorregut, ent a:enter);

```
            k=k+1;
        fmentre
        escriuremissatge ("\n");
        escriuremissatge ("El recorregut es: ");
        escriurerecorregut (ent/sor f:nom_ciutat, ent/sor w:recorregut, ent a:enter);
        escriuremissatge ("\n");
        escriuremissatge ("\n");
        escriuremissatge ("El cost minim es: ");
        escriurecaracter (w.dist);
Faccio
```

3.2.Pseudocodi

Pel que fa al pseudocodi es:

```
#include <assert.h>

#include <stdbool.h>

#include <stdio.h>


#define L 20

typedef struct{

        bool g;

        float l;

}mesura;

typedef char t_nom [L];

typedef mesura t_metres [L];

typedef struct {

  t_nom nom;

        int m;

        t_metres metres;

}distancies;

typedef distancies nom_ciutat [L];

typedef int t_lloc [L];

typedef struct {

  t_lloc lloc;

  float dist;

}recorregut;

typedef char unitat[L];
```

```
void llegirparaula(nom_ciutat f,int *const i);

void escriuparaula (nom_ciutat f, int i);

void llegirdistancia (nom_ciutat f,  int o,int n);

void maxim2(nom_ciutat f, int a, float *const max);

void boleacert (nom_ciutat f, int a);

void inizialitazartaula (recorregut *const w);

void falsprimer(nom_ciutat f, int a);

void falsultim(nom_ciutat f, int a);

void buscardistancia (nom_ciutat f, int a,int y,int *const b, float max);

void canviarestat(nom_ciutat f, int a, int y);

void comparartaula (recorregut *const w, recorregut d, int a);

void escriurerecorregut (nom_ciutat f, recorregut w, int a);

void buscarrecorregut (nom_ciutat f, recorregut w, int a, int c, recorregut d, float max);


int main(void){
        int a;
        float max;
        recorregut w;
        recorregut d;
        distancies e;
        nom_ciutat f;
       mesura q;
        unitat p1;
        char p2;
        int i;
        int k;
        char t;
```

```
int c;
c=0;
printf("\n");
scanf("%c",&p2);
while(p2!='\n'){
 p1[c]=p2;
 scanf("%c",&p2);
 c=c+1;
}
printf("\n");
 printf("Numero de  ciutats per les que es passa: ");
 scanf("%i",&a);
 printf("%i",a);
 printf("\n");
 printf("\n");
 char caux; scanf("%c",&caux);
  i=0;
  llegirparaula(f,&i);
  llegirparaula(f,&i);

  while (i < a){
            llegirparaula (f,&i);
 }
 k=0;
 printf("\n");
 while (k<a-1){
```

```
            i = k + 1;
            while (i < a){
                printf("Cost entre ");
                escriuparaula (f,k);
                printf(" i ");
                 escriuparaula (f, i);
                 printf(": ");
                   llegirdistancia (f,k,i);
                printf(" ");
                for (t=0; t<c; ++t){
                 printf("%c",p1[t]);
                }
          printf("\n");
          i=i+1;
           }
     k = k + 1;
   }
   maxim2(f,a,&max);
   for(k=0; k<a; ++k){
     f[k].metres[k].l = max +1;
   }
   buscarrecorregut (f, w, a, c, d, max);
 printf(" ");
for (t=0; t<c; ++t){
  printf("%c",p1[t]);
}
 printf("\n");
```

```
  return 0;
}


void llegirparaula(nom_ciutat f, int *const i){
  int j;
  char h;


  j=0;
  scanf("%c",&h);
  if (h==0){
  }
  else if (h!=0){
    f[*i].nom[j]=h;
    f[*i].m=f[*i].m+1;
  while (f[*i].nom[j] !='\n'){
    j = j + 1;
    scanf("%c",&(f[*i].nom[j]));
    f[*i].m = f[*i].m + 1;
  }
  }else assert (0);
  *i=*i+1;
}
```

```
void escriuparaula (nom_ciutat f, int i){


  int j;


  j=0;
  while(j < f[i].m-1){
    printf("%c", f[i].nom[j]);
    j=j+1;
  }
}



void llegirdistancia (nom_ciutat f, int o,int n){
  float g;
  scanf("%f",&g);
  while (g==0){
    scanf("%f",&g);}
  f[n].metres[o].l=g;
  f[o].metres[n].l=g;
  printf("%.2f",g);


}
```

```
void maxim2(nom_ciutat f, int a, float *const max){


  int i;

  int j;


  *max=0;

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

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

      *max=f[i].metres[j].l+*max;

}


void boleacert (nom_ciutat f, int a){

  int i;

  int j;

  i=0;

  while (i<a){

    j=0;

    while(j<a){

      f[i].metres[j].g=true;

      j=j+1;

    }

    i=i+1;

  }

}
```

```
void inicialitazartaula (recorregut *w){

  int i;

  (*w).dist = 0;
  for(i=0; i<L; i=i+1){
   (*w).lloc[i] = 0;
  }
}

void falsprimer(nom_ciutat f, int a){

  int z;
  z=0;
  while(z<a){
    f[z].metres[0].g = false;
    f[0].metres[z].g = false;
    z=z+1;
  }
}
```

```
void falsultim(nom_ciutat f, int a){


  int z;

  z=0;

  while (z<a){

    f[z].metres[1].g = false;

    f[1].metres[z].g = false;

    z=z+1;

  }

}

void buscardistancia (nom_ciutat f, int a,int y, int *const b, float max){


  int o;


  int min_dist;

  o=0;

  min_dist=max+1;

  while(o<a){

    if(f[y].metres[o].g==false){

    }else if(f[y].metres[o].g==true){

      if(min_dist<=f[y].metres[o].l){

      }else if(min_dist>f[y].metres[o].l){

            min_dist=f[y].metres[o].l;

            *b=o;

      }else assert(0);

    }else assert(0);
```

```
    o=o+1;
  }
}

void canviarestat(nom_ciutat f, int a, int y){
  int i;
  i=0;
  while (i<a){
    f[i].metres[y].g=false;
    i= i + 1;
  }
}

void comparartaula (recorregut *w, recorregut d, int a){

  int i;

  i=0;
  if((*w).dist<=d.dist){
  }else if((*w).dist>d.dist){
    while(i<=a){
      (*w).lloc[i]=d.lloc[i];
      i=i+1;
    }
    (*w).dist=d.dist;
  }else assert(0)
```

```
void escriurerecorregut (nom_ciutat f, recorregut w, int a){


  int i;

  int j;

  int x;

  i=0;

  j=0;

  while(i<=a-1){

    x=w.lloc[i];

    while(j<f[x].m-1){

      printf("%c",f[x].nom[j]);

      j=j+1;

    }

    j=0;

    i=i+1;

    printf(" ");

  }

}
```

```
void buscarrecorregut ( nom_ciutat f, recorregut w,int a, int c, recorregut d, float max){
  int p;
  int m;
  int b;
  int k;
  int e;
  int y;
  inizialitazartaula (&w);
  boleacert (f,a);
  m=0;
  y=0;
  w.lloc[m]=0;
  m=1;
  w.dist=f[y].metres[2].l;
  falsprimer(f,a);
  falsultim(f,a);
  y=2;
  while(m<a-2){
    buscardistancia (f,a,y,&b,max);
    w.dist=w.dist+f[y].metres[b].l;
    w.lloc[m]=y;
    m=m+1;
    canviarestat(f,a,y);
    y=b;
  }
  w.lloc[m]=y;
  m=m+1;
```

```
w.dist=w.dist+f[b].metres[1].l;
w.lloc[m]=1;
k=2;
boleacert (f,a);
y=1;

while(k<a){
 y=k;
 inizialitazartaula (&d);
 d.dist=f[0].metres[k].l;
 d.lloc[0]=0;
 falsprimer(f,a);
 falsultim(f,a);
 p=1;
 while(p<a-2){
  buscardistancia (f,a,y,&b,max);
  d.dist=d.dist+f[y].metres[b].l;
  d.lloc[p]=y;
  p=p+1;
  canviarestat(f,a,y);
  y=b;
 }
 d.lloc[p]=y;
 p=p+1;
 d.dist=d.dist+f[b].metres[1].l;
 d.lloc[p]=1;
 boleacert (f,a);
```

```
    comparartaula (&w,d,a);
    k=k+1;
  }
  printf("\n");
  printf("El recorregut es: ");
  escriurerecorregut (f,w,a);
  printf("\n");
  printf("\n");
  printf("El cost minim es: ");
  printf("%.2f", w.dist);
}
```

3.3.Explicació

La funció del programa es que un cop entrades les dades necessàries que son:

- Unitat de mesura.
- Ciutat inicial.
- Ciutat final.
- Resta de ciutats que tens de visitar.
- Cost entre totes les ciutats.

La manera d'introduir les dades es a traves d'omplir un full de dades creat amb el programa Excel.

El que fa el programa un ja han estat introduïdes les dades és:

7. Anar de la ciutat inicial a la ciutat 2
8. Un cop a la ciutat dos, mirar a quina ciutat el cost és menor, anar-hi.
9. Repetir el pas 2 (amb les ciutats que encara no s'han visitat), fins que s'hagin visitat totes les ciutats excepte la ciutat final.
10. Un cop visitades totes les ciutats, es dirigeix a la final.
11. Repetir els passos 1, 2, 3 i 4 començant per la resta de ciutats, fins que totes excepte la ciutat final hagin estat visitades en primera posició.
12. Un cop acabat un recorregut comparar la millor opció amb la opció torbada.
13. Escriure el millor recorregut trobat amb el seu cost corresponent.

L'explicació dels subprogrames és:

Llegirparaula	La funció d'aquest subprograma és la de llegir les ciutats que tens de passar, ajudat per el mentre que hi ha al algorisme principal.
Escriureparaula	La funció d'aquest subprograma és la d'escriure les ciutats que tens de passar, ajudat per el mentre que hi ha al algorisme principal.
Llegirdistancia	La funció d'aquest subprograma és de la de agafar i guardar totes les distàncies entre les ciutats pertinents.

Maxim2	La funció d'aquest subprograma és la de crear el valor màxim, és a dir, la suma de tots el punts per tal de posar-ho a dintre de la matriu de distàncies al lloc on correspon a la distancia entre la mateixa ciutat, d'aquesta manera s'evita que des de la mateixa ciutat, pugis tornar a passar.
Boleacert	La funció d'aquest subprograma és la de un cop ja ha estat creat un recorregut posar totes les ciutats que encara les pots visitar per així poder tornar a fer un nou recorregut.
Inicialitzartaula	La funció d'aquest subprograma es posar a zero la taula en que es guarda el recorregut, és un pas necessari ja que si no el fas el programa sol assignar un valor aleatori a cada lloc i podria donar errors alhora d'executar-lo.
Falsprimer	La funció d'aquest subprograma és la de dir que no es pot anar a la primera ciutat (ciutat inicial) fins que s'hagi passat per totes les altres.
Falsultim	La funció d'aquest subprograma és la de dir que no es pot anar a la ultima ciutat (ciutat final) fins que s'hagi passat per totes les altres.
Buscardistancia	La funció d'aquest subprograma és la de comparar totes les distàncies de la ciutat actual fins a les altres que encara no s'ha passat per poder escollir la ciutat més a prop possible per anar-hi.
Canviarestat	La funció d'aquest subprograma és negar la possibilitat de retornar a una ciutat que ja s'ha passat anteriorment.

Comparartaula	La funció d'aquest subprograma és la de comparar el millor recorregut obtingut fins el moment amb el nou recorregut que has obtingut, per tal d'observar quin del dos és més curt.
Escriurerecorregut	La funció d'aquest subprograma és un cop fet tots els recorreguts possibles, escriure el millor recorregut trobat.
Buscarrecorregut	La funció d'aquest subprograma és la de buscar una ruta que encara no s'hagi observat i guardar-la a un taula per poder comparar-la.

4.ALGORISME VARIANT AMB RECORREGUT TANCAT

4.1 Implementació

També s'ha creat un variant d'aquest programa, en el que el seu algorisme és:

algorisme EVC2

const

L: **enter** =20;

fconst

Tipus

mesura: tupla

g:**booleà**;

l: **real**;

ftupla

t_nom=taula de [1...L] de **caràcters**;

t_metres= taula [1...L] de mesura;

```
        distancies: tupla
                nom: t_nom;
                m: enter;
                metres: t_metres;
        ftupla
        nom ciutat= taula de [1...L] de distancies;
        t_lloc= taula de [1...L] de enters;
        recorregut: tupla
                lloc: t_lloc;
                dist: real;
        ftupla
        unitat= taula de [1...L]de caràcter;
ftipus
var
        a: enter;
        max: real;
        w: recorregut;
        d: recorregut;
        e: distancies;
        f: nom_ciutat;
        q: mesura;
        p1: unitat;
        p2: caràcter;
        t: caràcter;
        i: enter;
        k: enter;
        c: enter;
```

caux: **caràcter**;

fvar

c=o;

escriuremissatge (“Quina unitat es fara servir? ”);

llegircaracter (p2);

mentre (p2 ≠ ‘\n’) **fer**

p1[c]=p2;

llegircaracter (p2);

c=c+1;

fmentre

escriuremissatge (“\n”);

escriuremissatge (“Per quantes ciutats es passa? “);

llegirenter (a);

escriuremissatge (“\n”);

llegircaracter (caux);

escriuremissatge (“Quina es la ciutat inicial? ”);

i=0;

llegirparaula (ent/sor f:nom_ciutat , ent/sor i:enter);

escriuremissatge (“\n”);

escriuremissatge ("Quina es la ciutat final? ");

c=a-1;

llegirparaula (ent/sor f:nom_ciutat , ent/sor i:enter);

escriuremissatge (“\n”);

escriuremissatge ("Escriu el nom de les ciutats restants (separades per un intro) \n");

mentre (i < a) **fer**

llegirparaula (ent/sor f:nom_ciutat , ent/sor i:enter);

```
fmentre
k=0;
escriuremissatge ("\n");
mentre (k < a-1) fer
        i=k+1;
        mentre (i<a) fer
                escriuremissatge ("Escriu el cost entre");
                escriureparaula (ent/sor f:nom_ciutat, ent k:enter);
                escriuremissatge (" i ");
                escriureparaula (ent/sor f:nom_ciutat, ent k:enter);
                escriuremissatge (":  ");
                llegirdistancia();
                escriuremissatge ("\n");
                i=i+1;
        fmentre
        k=k+1;
fmentre
maxim2 (ent/sor f:nom_ciutat, ent a:enter, sor max:real);
per (k=0; k< a; k=k+1) fer
        f[k].metres[k].l=max+1;
fper
buscarrecorregut ( ent/sor f:nom_ciutat, ent w:recorregut, ent a:enter, ent
d:recorregut);
escriuremissatge (" ");
per (t=0; t<c; t=t+1) fer
        escriurecaracter (p1[t]);
fper
```

```
        escriuremissatge ("\n");
falgorisme

SUBPROGRAMES

accio llegirparaula (ent/sor f:nom_ciutat, ent/sor i:enter)
        var
                j:enter;
                h:carácter;
        fvar
        j=0;
        llegircaracter (f[i].nom[j]);
        mentre (f[i].nom[j] ≠'\n') fer
                j=j+1;
                llegircarcter(f[i].nom[j]);
                f[i].m=f[i].m+1;
        fmentre
        i=i+1;
Faccio


Accio escriuparaula (ent/sor f:nom_ciutat, ent: i:enter)
        Var
                j:enter
        Fvar

        j=0;
        mentre (j<f[i].m-1) fer
                escriurecaracter(f[i].nom[j]);
                j=j+1
        fmentre
```

```
Facció


Acció llegirdistancia (ent/sor f:nom_ciutat, ent o:enter, ent n:enter)
        g:real;
        llegirenter (g);
        f[n].metres[o].l= g;
        f[o].metres[n].l= g;
        escriurereal (g);
Facció


Acció maxim2(ent/sor f:nom_ciutat, ent a:enter, ent/sor max:real)
        Var
                i:enter;
                j: enter;
        fvar
        max=0;
        per (i=0; i < a; i=i+1) fer
                per (j=0; j < a; j=j+1) fer
                        max= f[i].metres[j].l +max;
                fper
        fper
Faccio
```

```
Acció boleacert(ent/sor f:nom_ciutat, ent a:enter)
    Var
        i,j:enter;
    Fvar
    i=0;
    mentre (i<a) fer
        j=0;
        mentre(j<a) fer
            f[i].metres[j].g =cert;
            j=j+1;
        fmentre
        i=i+1;
    fmentre
Facció

Acció Inicialitzartaula (ent/sor w:recorregut)
    var
        i:enter;
    fvar
    (w).dist=0;
    per (i=0; i>L; i=i+1) fer
        w.lloc[i] =0;
    fper
Facció
```

```
Acció falsprimer(ent/sor f:nom_ciutat, ent a:enter)
    Var
        z:enter;
    fvar
    z=0;
    mentre (z<a) fer
        f[z].metres[0].g=fals;
        z=z+1;
    fmentre
Facció

Acció falsultim(ent/sor f:nom_ciutat, ent a:enter)
    Var
        z:enter;
    fvar
    z=0;
    mentre (z<a) fer
        f[z].metres[1].g=fals;
        f[1].metres[z].g=fals;
        z=z+1;
    fmentre
Facció
```

```
Acció buscardistancia (ent/sor f:nom_ciutat, ent a,y:enter, sor b:enter ent max:real)
    var
        o, min_dist:enter;
    fvar
    o=0;
    min_dist=max + 1;
    mentre (o<a) fer
        si (f[y].metres[o].g= fals) fer
        res;
        sino (f[y].metres[o].g = cert) fer
            si (min_dist<= f[y].metres[o].l) fer
                res;
            sino (min_dist > f[y].metres[o].l) fer
                mindist=f[y].metres[o].l;
                b=o;
            fsi
        fsi
        o=o+1;
    fmentre
Faccio
```

```
Accio canviarestat (ent/sor f:nom_ciutat, ent a,y:enter)
        Var
                i:enter;
        fvar
        i=0;
        mentre (i < a) fer
                f[i].metres[y].g=fals;
                i=i+1;
        fmentre
Faccio


Acció comparartaula(ent/sor w,d:recorregut, ent a:enter)
        var
                i:enter;
        fvar
        i=0;
        si (w.dist<=d.dist) fer
        res;
        sino (w.dist>d.dist) fer
                mentre (i<=a)fer
                        w.lloc[i]= d.lloc[i];
                        i=i+1;
                fmentre
                w.dist=d.dist;
        fsi
Facció
```

```
Acció escriurerecorregut(ent/sor f:nom_ciutat, ent/sor w:recorregut, ent a:enter)
    var
        i,j,x:enter
    fvar
    i=0;
    j=0;
    mentre (i<=a-1) fer
        x=w.lloc[i];
        mentre (j<f[x].m-1) fer
            escriurecaracter(f[x].nom[j];
            j=j+1;
        fmentre
        j=0;
        i=i+1;
        escriuremissatge (" ");
    fmentre
Faccio

acció buscarrecorregut (ent/sor f:nom_ciutat, ent w, d: recorregut, ent a:enter, ent max:real)
    var
        p, m, b, k, e, y: enter;
    fvar
    inicialitzartaula (ent/sor w:recorregut);
    boleacert (ent/sor f:nom_ciutat, ent a:enter);
    m=0;
    y=0;
```

```
w.lloc[m]=0;
m=1;
w.dist=f[y].metres[2].l;
falsprimer(ent/sor f:nom_ciutat, ent a:enter);
falsultim(ent/sor f:nom_ciutat, ent a:enter);
y=2;
mentre (m < a-2) fer
        buscardistancia (ent/sor f:nom_ciutat, ent a,y:enter, sor b:enter ent max:real);
        w.dist=w.dist+f[y].metres[b].l;
        w.lloc[m]=y;
        m=m+1;
        canviarestat (ent/sor f:nom_ciutat, ent a,y:enter);
        y=b;
fmentre
w.lloc[m]=y;
m=m+1;
w.dist=w.dist+f[b]metres[1].l;
w.lloc[m]=1;
k=1;
boleacert(ent/sor f:nom_ciutat, ent a:enter);
y=1;
mentre (k < a) fer
        y=k;
        Inicialitzartaula (ent/sor w:recorregut);
        d.dist=f[0].metres[k].l;
        d.lloc[0]=0;
        falsprimer (ent/sor f:nom_ciutat, ent a:enter);
```

```
        falsultim(ent/sor f:nom_ciutat, ent a:enter);
        p=1;
        mentre (p < a-2) fer
            buscardistancia (ent/sor f:nom_ciutat, ent a,y:enter, sor b:enter ent
            max:real);
            d.dist=d.dist+f[y].metres[b].l;
            d.lloc[p]=y;
            p=p+1;
            canviarestat (ent/sor f:nom_ciutat, ent a,y:enter);
            y=b;
            fmentre
        d.lloc[p]=b;
        p=p+1;
        d.dist=d.dist+f[b].metres[1].l;
        d.lloc[p]=1;
        boleacert(ent/sor f:nom_ciutat, ent a:enter);
        comparartaula (ent/sor w,d:recorregut, ent a:enter);
        k=k+1;
fmentre
escriuremissatge ("\n");
escriuremissatge ("El recorregut es: ");
escriurerecorregut (ent/sor f:nom_ciutat, ent/sor w:recorregut, ent a:enter);
escriuremissatge ("\n");
escriuremissatge ("\n");
escriuremissatge ("El cost minim es: ");
escriurecaracter (w.dist);
Faccio
```

4.2 Pseudocodi

El pseudocodi es:

```
#include <assert.h>

#include <stdbool.h>

#include <stdio.h>


#define L 100

typedef struct{

  bool g;

  float l;

}mesura;

typedef char t_nom [L];

typedef mesura t_metres [L];

typedef struct {

  t_nom nom;

  int m;

  t_metres metres;

}distancies;

typedef distancies nom_ciutat [L];

typedef int t_lloc [L];

typedef struct {

  t_lloc lloc;

  float dist;

}recorregut;

typedef char unitat[L];


void llegirparaula(nom_ciutat f,int *const i);
```

```
void escriuparaula (nom_ciutat f, int i);

void llegirdistancia (nom_ciutat f,  int o,int n);

void maxim2(nom_ciutat f, int a, float *const max);

void boleacert (nom_ciutat f, int a);

void inizialitazartaula (recorregut *const w);

void falsprimer(nom_ciutat f, int a);

void falsultim(nom_ciutat f, int a);

void buscardistancia (nom_ciutat f, int a,int y,int *const b, float max);

void canviarestat(nom_ciutat f, int a, int y);

void comparartaula (recorregut *const w, recorregut d, int a);

void escriurerecorregut (nom_ciutat f, recorregut w, int a);

void buscarrecorregut (nom_ciutat f, recorregut w, int a, int c, recorregut d, float max);


int main(void){
  int a;
  float max;
  recorregut w;
  recorregut d;
  distancies e;
  nom_ciutat f;
  mesura q;
  unitat p1;
  char p2;
  int i;
  int k;
  char t;
  int c;
```

```
c=0;
printf("\n");
printf("Quina unitat es fara servir? ");
scanf("%c",&p2);
while(p2!='\n'){
  p1[c]=p2;
  scanf("%c",&p2);
  c=c+1;
}
printf("\n");
  printf("Per quantes ciutats es passa? ");
  scanf("%i",&a);
  printf("\n");
  char caux; scanf("%c",&caux);
  printf("Quina es la ciutat inicial? ");
  i=0;
  llegirparaula(f,&i);
  printf("\n");
  printf("Quina es la ciutat final? ");
  llegirparaula(f,&i);
  printf("\n");
  printf("Escriu el nom de les ciutats restants (separades per un intro) \n");
  while (i < a){
    llegirparaula (f,&i);
  }

  k=0;
```

```
    printf("\n");
    while (k<a-1){

     i = k + 1;
     while (i < a){
           printf("Escriu el cost entre ");
           escriuparaula (f,k);
       printf(" i ");
       escriuparaula (f, i);
       printf(": ");
       llegirdistancia (f,k,i);
           printf("\n");
           i=i+1;
     }
     k = k + 1;
    }
    maxim2(f,a,&max);
    for(k=0; k<a; ++k){
     f[k].metres[k].l = max +1;
    }
    buscarrecorregut (f, w, a, c, d, max);
 printf(" ");
for (t=0; t<c; ++t){
  printf("%c",p1[t]);
}
 printf("\n");
  return 0;
```

```
}


void llegirparaula(nom_ciutat f, int *const i){
  int j;
  char h;

  j=0;
  scanf("%c",&(f[*i].nom[j]));
  while (f[*i].nom[j] !='\n'){
    j = j + 1;
    scanf("%c",&(f[*i].nom[j]));
    f[*i].m = f[*i].m + 1;
  }
  *i=*i+1;
}



void escriuparaula (nom_ciutat f, int i){

  int j;

  j=0;
  while(j < f[i].m){
    printf("%c", f[i].nom[j]);
    j=j+1;
  }
}
```

```
void llegirdistancia (nom_ciutat f, int o,int n){
  float g;
  scanf("%f",&g);

  f[n].metres[o].l=g;
  f[o].metres[n].l=g;

}

void maxim2(nom_ciutat f, int a, float *const max){

  int i;
  int j;

  *max=0;
  for(i=0; i<a; i++)
    for(j=0; j<a; j++)
      *max=f[i].metres[j].l+*max;
}

void boleacert (nom_ciutat f, int a){
```

```
  int i;
  int j;
  i=0;
  while (i<a){
    j=0;
    while(j<a){
      f[i].metres[j].g=true;
      j=j+1;
    }
    i=i+1;
  }
}


void inizialitazartaula (recorregut *w){

  int i;

  (*w).dist = 0;
  for(i=0; i<L; i=i+1){
   (*w).lloc[i] = 0;
  }
}


void falsprimer(nom_ciutat f, int a){
```

```
  int z;
  z=0;
  while(z<a){
    f[z].metres[0].g = false;
    f[0].metres[z].g = false;
    z=z+1;
  }
}

void falsultim(nom_ciutat f, int a){

  int z;
  z=0;
  while (z<a){
    f[z].metres[1].g = false;
    f[1].metres[z].g=false;
    z=z+1;
  }
}
```

```
void buscardistancia (nom_ciutat f, int a,int y, int *const b, float max){
```

```
  int o;


  int min_dist;
  o=0;
  min_dist=max+1;
  while(o<a){
    if(f[y].metres[o].g==false){
    }else if(f[y].metres[o].g==true){
      if(min_dist<=f[y].metres[o].l){
      }else if(min_dist>f[y].metres[o].l){
            min_dist=f[y].metres[o].l;
            *b=o;
      }else assert(0);
    }else assert(0);
    o=o+1;
  }
}
void canviarestat(nom_ciutat f, int a, int y){


  int i;
  i=0;
  while (i<a){
    f[i].metres[y].g=false;
    i= i + 1;
  }
}
```

```
void comparartaula (recorregut *w, recorregut d, int a){

  int i;



  i=0;
  if((*w).dist<=d.dist){
  }else if((*w).dist>d.dist){
    while(i<=a){
      (*w).lloc[i]=d.lloc[i];
      i=i+1;
    }
    (*w).dist=d.dist;
  }else assert(0);
}



void escriurerecorregut (nom_ciutat f, recorregut w, int a){

  int i;
  int j;
  int x;
  i=0;
  j=0;
  while(i<=a-1){
```

```
   x=w.lloc[i];
   while(j<f[x].m){
     printf("%c",f[x].nom[j]);
     j=j+1;
   }
   j=0;
   i=i+1;
   printf(" ");
  }
}


void buscarrecorregut ( nom_ciutat f, recorregut w,int a, int c, recorregut d, float max){
  int p;
  int m;
  int b;
  int k;
  int e;
  int y;
  inizialitazartaula (&w);
  boleacert (f,a);
  m=0;
  y=0;
  w.lloc[m]=0;
  m=1;
  w.dist=f[y].metres[2].l;
  falsprimer(f,a);
```

```
falsultim(f,a);;
y=2;
while(m<a-2){
  buscardistancia (f,a,y,&b,max);
  w.dist=w.dist+f[y].metres[b].l;
  w.lloc[m]=y;
  m=m+1;
  canviarestat(f,a,y);
  y=b;
}
w.lloc[m]=b;
m=m+1;
w.dist=w.dist+f[b].metres[1].l;
w.lloc[m]=1;
k=2;
boleacert (f,a);
y=1;

while(k<a){
  y=k;
  inizialitazartaula (&d);
  d.dist=f[0].metres[k].l;
  d.lloc[0]=0;
  falsprimer(f,a);
  falsultim(f,a);
  p=1;
  while(p<a-2){
```

```
    buscardistancia (f,a,y,&b,max);
    d.dist=d.dist+f[y].metres[b].l;
    d.lloc[p]=y;
    p=p+1;
    canviarestat(f,a,y);
    y=b;
   }
   d.lloc[p]=b;
   p=p+1;
   d.dist=d.dist+f[b].metres[1].l;
   d.lloc[p]=1;
   boleacert (f,a);
   comparartaula (&w,d,a);
   k=k+1;
  }
  printf("\n");
  printf("El recorregut es: ");
  escriurerecorregut (f,w,a);
  printf("\n");
  printf("\n");
  printf("El cost minim es: ");
  printf("%.2f", w.dist);
}
```

4.3 Explicació

La diferencia entre el programa original i la seva variant és en la manera d'obtenir les dades ja que el programa original (evc2txt) les dades s'obtenen a traves d'un full Excel, en canvi en la variació les dades s'introdueixen quan el programa te les demana, un cop ja l'has començat a executar a dintre de el MS-DOS.