Pràctica PRO2 Reproducció al Laboratori

versió 0.2

Generat per Doxygen 1.8.6

Dc Abr 23 2014 23:06:51

Índex

Capítol 1

Pràctica de PRO2: Reproducció al laboratori

En aquesta pràctica de PRO2 s'utilitza el disseny modular per a la interacció amb organismes de manera que puguin créixer, decréixer, reproduir-se i morir especificat per l'usuari del programa. S'utilitza les classes *Organisme*, *ConjuntOrg* i *Ranking*.

Capítol 2

Índex Jeràrquic

2.1 Jerarquia de Classes

Aquesta llista d'herència està ordenada toscament, però no completa, de forma alfabètica:

Capítol 3

Índex de Classes

3.1 Llista de Classes

Aquestes són les classes, estructures, unions i interfícies acompanyades amb breus descripcions:

Capítol 4

Índex de Fitxers

4.1 Llista dels Fitxers

Aquesta és la llista de tots els fitxers acompanyats amb breus descripcions:

Capítol 5

Documentació de les Classes

5.1 Referència de la Classe Template Arbre< T >

Classes

- struct node_arbre

Mètodes públics

- Arbre ()
- Arbre (const Arbre &original)
- ~Arbre ()
- Arbre & operator= (const Arbre &original)
- void a_buit ()
- void swap (Arbre &a)
- void plantar (const T &x, Arbre &a1, Arbre &a2)
- void fills (Arbre &fe, Arbre &fd)
- T arrel () const
- bool es_buit () const

Mètodes Privats

- node_arbre * copia_node_arbre (node_arbre *m)
- void esborra_node_arbre (node_arbre *m)

Atributs Privats

- node_arbre * primer_node

5.1.1 Descripció Detallada

template<class T>class Arbre< T >

Definició a la línia 6 del fitxer Arbre.hpp.

5.1.2 Documentació del Constructor i el Destructor

5.1.2.1 template<class T> Arbre< T >::Arbre ( )

Definició a la línia 55 del fitxer Arbre.hpp.

```
58   {
59     primer_node= NULL;
60   }
```

5.1.2.2 template<class T> Arbre< T >::Arbre (const Arbre< T > & *original*)

Definició a la línia 62 del fitxer Arbre.hpp.

```
65   {
66     if (this != &original)
67       primer_node = copia_node_arbre(original.
      primer_node);
68   }
```

5.1.2.3 template<class T> Arbre< T >::~Arbre ( )

Definició a la línia 70 del fitxer Arbre.hpp.

```
70           {
71     esborra_node_arbre(primer_node);
72   }
```

5.1.3 Documentació de les Funcions Membre

5.1.3.1 template<class T> node_arbre* Arbre< T >::copia_node_arbre (node_arbre * *m*) [private]

Definició a la línia 20 del fitxer Arbre.hpp.

```
26   {
27     node_arbre* n;
28     if (m==NULL) n=NULL;
29     else {
30       n = new node_arbre;
31       n->info = m->info;
32       n->segE = copia_node_arbre(m->segE);
33       n->segD = copia_node_arbre(m->segD);
34     }
35     return n;
36   }
```

5.1.3.2 template<class T> void Arbre< T >::esborra_node_arbre (node_arbre * *m*) [private]

Definició a la línia 38 del fitxer Arbre.hpp.

```
43   {
44     if (m != NULL) {
45       esborra_node_arbre(m->segE);
46       esborra_node_arbre(m->segD);
47       delete m;
48     }
49   }
```

5.1.3.3 template<class T> Arbre& Arbre< T >::operator= (const Arbre< T > & *original*)

Definició a la línia 74 del fitxer Arbre.hpp.

```
74                                          {
75     if (this != &original) {
76       esborra_node_arbre(primer_node);
77       primer_node = copia_node_arbre(original.
       primer_node);
78     }
79     return *this;
80   }
```

5.1.3.4 template<class T> void Arbre< T >::a_buit ( )

Definició a la línia 82 del fitxer Arbre.hpp.

```
85   {
86     esborra_node_arbre(primer_node);
87     primer_node= NULL;
88   }
```

5.1.3.5 template<class T> void Arbre< T >::swap (Arbre< T > & *a*)

Definició a la línia 90 del fitxer Arbre.hpp.

```
93   {
94     node_arbre* aux;
95     aux = a.primer_node;
96     a.primer_node = primer_node;
97     primer_node = aux;
98   }
```

5.1.3.6 template<class T> void Arbre< T >::plantar (const T & *x*, Arbre< T > & *a1*, Arbre< T > & *a2*)

Definició a la línia 100 del fitxer Arbre.hpp.

```
105   {
106     if (this != &a1 and this != &a2) {
107       if (primer_node==NULL) {
108         node_arbre* aux;
109         aux= new node_arbre;
110         aux->info= x;
111         aux->segE= a1.primer_node;
112         if (a1.primer_node == a2.primer_node) aux->segD=
       copia_node_arbre(a1.primer_node);
113         else  aux->segD= a2.primer_node;
114         primer_node= aux;
115         a1.primer_node= NULL;
116         a2.primer_node= NULL;
117       }
118       else
119         throw PRO2Excepcio ("El p.i. de plantar ha de ser buit a la crida");
120     }
121     else
122       throw PRO2Excepcio ("El p.i. de plantar no pot coincidir amb els partres");
123   }
```

5.1.3.7 template<class T> void Arbre< T >::fills (Arbre< T > & *fe*, Arbre< T > & *fd*)

Definició a la línia 126 del fitxer Arbre.hpp.

```
130   {
131     if (primer_node!=NULL and fe.primer_node==NULL
132         and fd.primer_node==NULL) {
133       if (&fe != &fd) {
134         node_arbre* aux;
135         aux= primer_node;
136         fe.primer_node= aux->segE;
137         fd.primer_node= aux->segD;
138         primer_node= NULL;
139         delete aux;
140       }
141       else
142         throw PRO2Excepcio
143               ("Els dos partres de fills no poden coincidir");
144     }
145     else if (primer_node==NULL)
146       throw PRO2Excepcio ("Un arbre buit no tills");
147     else
148       throw PRO2Excepcio
149         ("Els dos partres de fills han de ser buits a la crida");
150   }
```

5.1.3.8 template<class T> T Arbre< T >::arrel ( ) const

Definició a la línia 152 del fitxer Arbre.hpp.

```
155   {
156     if (primer_node!=NULL)
157       return primer_node->info;
158     else
159       throw PRO2Excepcio ("Un arbre buit no trrel");
160   }
```

5.1.3.9 template<class T> bool Arbre< T >::es_buit ( ) const

Definició a la línia 162 del fitxer Arbre.hpp.

```
165   {
166     return (primer_node==NULL);
167   }
```

5.1.4 Documentació de les Dades Membre

5.1.4.1 template<class T> node_arbre* Arbre< T >::primer_node [private]

Definició a la línia 16 del fitxer Arbre.hpp.

La documentació d'aquesta classe es va generar a partir del següent fitxer:

- Arbre.hpp

5.2 Referència de l'Estructura Organisme::Celula

Element bàsic de cada organisme.

Atributs Públics

- int id

  És el número que identifica la cèl·lula.
- bool activa

  Booleà que indica si la cèl·lula és activa o no.

5.2.1 Descripció Detallada

Element bàsic de cada organisme.

Definició a la línia 19 del fitxer Organisme.hpp.

5.2.2 Documentació de les Dades Membre

5.2.2.1 Organisme::Celula::id

És el número que identifica la cèl·lula.

Definició a la línia 23 del fitxer Organisme.hpp.

5.2.2.2 Organisme::Celula::activa

Booleà que indica si la cèl·lula és activa o no.

Definició a la línia 26 del fitxer Organisme.hpp.

La documentació d'aquesta estructura es va generar a partir del següent fitxer:

- Organisme.hpp

5.3 Referència de la Classe ConjuntOrg

És un conjunt d'organismes.

Mètodes públics

- ConjuntOrg (int M)

  Constructora per defecte.
- ConjuntOrg (const ConjuntOrg &C)

  Constructora per còpia.
- ~ConjuntOrg ()

  Destructora per defecte.
- void estirar (int p)

  Modificadora per estirar un subconjunt d'organismes.
- void retallar (int p)

  Modificadora per retallar un subconjunt d'organismes.
- bool reproduir (Ranking &rank, int &fills)

  Modificadora que fa una ronda de reproducció dels organismes.
- int consultar_tamany () const

  Consultora que retorna el nombre d'organismes del conjunt.
- bool morts () const

  Consultora que ens diu si els organismes estan morts.
- void escriure_ultims (int n)

  Escriu els últims 'n' elements del conjutn.
- void llegir ()

  Llegeix un conjunt d'organismes.
- void estat (int p) const

  Imprimeix l'estat d'un subconjunt d'organismes.

Atributs Privats

- vector< Organisme > V

  Vector on es guardaran tots els organismes.
- vector< vector< bool > > Aparellat

  Matriu que ens dirà quins organismes s'han aparellat i amb qui ho han fet.
- int tamany

  Variable que ens dona el número de organismes que hi ha al vector.

5.3.1 Descripció Detallada

És un conjunt d'organismes.

Definició a la línia 15 del fitxer ConjuntOrg.hpp.

5.3.2 Documentació del Constructor i el Destructor

5.3.2.1 ConjuntOrg::ConjuntOrg (int *M*)

Constructora per defecte.

S'executa automàticament al declarar un conjunt

Precondició

M ha de ser un nombre enter més gran que '0'

Postcondició

El resultat és un conjunt d'organismes de tamany M però buit

Definició a la línia 11 del fitxer ConjuntOrg.cpp.

```
11                              {
12   V = vector<Organisme> (M);
13   Aparellat = vector< vector<bool> > (M, vector<bool> (M));
14   tamany = 0;
15 }
```

5.3.2.2 ConjuntOrg::ConjuntOrg (const ConjuntOrg & *C*)

Constructora per còpia.

Precondició

Cert

Postcondició

El paràmetre implícit passa a ser igual al afegit a la funció

Definició a la línia 17 del fitxer ConjuntOrg.cpp.

```
18 {
19   V = c.V;
20     Aparellat = c.Aparellat;
21   tamany = c.tamany;
22 }
```

5.3.2.3 ConjuntOrg::∼ConjuntOrg ( )

Destructora per defecte.

Esborra automàticament l'objecte al sortir d'un àmbit de visibilitat

Definició a la línia 24 del fitxer ConjuntOrg.cpp.

```
24 {}
```

5.3.3 Documentació de les Funcions Membre

5.3.3.1 void ConjuntOrg::estirar (int *p*)

Modificadora per estirar un subconjunt d'organismes.

Precondició

Es passa una pila amb identificadors d'organismes vàlids

Postcondició

Els organismes que tenen l'identificador de la pila han estat estirats

Definició a la línia 30 del fitxer ConjuntOrg.cpp.

```
31 {
32     V[p - 1].estirar_organisme();
33 }
```

5.3.3.2 void ConjuntOrg::retallar (int *p*)

Modificadora per retallar un subconjunt d'organismes.

Precondició

Es passa una pila amb identificadors d'organismes vàlids

Postcondició

Els organismes amb l'identificador de la pila han estat retallats

Definició a la línia 35 del fitxer ConjuntOrg.cpp.

```
36 {
37     V[p - 1].retallar_organisme();
38 }
```

5.3.3.3 bool ConjuntOrg::reproduir (Ranking & *rank*, int & *fills*)

Modificadora que fa una ronda de reproducció dels organismes.

Si la reproducció no s'ha pogut realitzar correctament es retorna un booleà 'false', en cas contrari retorna 'true'

Precondició

Cert

Postcondició

Tots els organismes que poden s'han reproduit un cop com a màxim a més a més s'imprimeixen els fills nous de la ronda

Definició a la línia 40 del fitxer ConjuntOrg.cpp.

```
41 {
42     return true;
43 }
```

5.3.3.4 int ConjuntOrg::consultar_tamany ( ) const

Consultora que retorna el nombre d'organismes del conjunt.

Precondició

Cert

Postcondició

Es retorna el nombre d'organismes (vius o morts) que hi ha al conjunt

Definició a la línia 49 del fitxer ConjuntOrg.cpp.

```
50 { return tamany; }
```

5.3.3.5 bool ConjuntOrg::morts ( ) const

Consultora que ens diu si els organismes estan morts.

Si tots els organismes del Conjunt estan morts es retorna *true* i en cas contrari es retorna *false*

Precondició

Cert

Postcondició

Es retorna un booleà amb l'estat dels organismes

Definició a la línia 52 del fitxer ConjuntOrg.cpp.

```
53 {
54   bool mort = true;
55   for (int i = 0; i < tamany; ++i) if (not V[i].es_mort()) mort = false;
56   return mort;
57 }
```

5.3.3.6 void ConjuntOrg::escriure_ultims (int *n*)

Escriu els últims 'n' elements del conjutn.

Precondició

Hi ha com a mínim 'n' elemnts

Postcondició

Pel canal estàndard de sortida s'escriuen els 'n' últims elements

5.3.3.7 void ConjuntOrg::llegir ( )

Llegeix un conjunt d'organismes.

Precondició

Cert

Postcondició

Es llegeixen els organismes inicials del conjunt

5.3.3.8 void ConjuntOrg::estat (int *p*) const

Imprimeix l'estat d'un subconjunt d'organismes.

Precondició

Cert

Postcondició

S'imprimeix l'estat de cada organisme que es passa a la pila

Definició a la línia 63 del fitxer ConjuntOrg.cpp.

```
64 {
65     V[p - 1].escriure_organisme();
66 }
```

5.3.4 Documentació de les Dades Membre

5.3.4.1 vector<Organisme> ConjuntOrg::V [private]

Vector on es guardaran tots els organismes.

Definició a la línia 19 del fitxer ConjuntOrg.hpp.

5.3.4.2 vector< vector<bool> > ConjuntOrg::Aparellat [private]

Matriu que ens dirà quins organismes s'han aparellat i amb qui ho han fet.

Definició a la línia 23 del fitxer ConjuntOrg.hpp.

5.3.4.3 int ConjuntOrg::tamany [private]

Variable que ens dona el número de organismes que hi ha al vector.

Definició a la línia 27 del fitxer ConjuntOrg.hpp.

La documentació d'aquesta classe es va generar a partir dels següents fitxers:

- ConjuntOrg.hpp
- ConjuntOrg.cpp

5.4 Referència de l'Estructura Arbre< T >::node_arbre

Atributs Públics

- T info
- node_arbre * segE
- node_arbre * segD

5.4.1 Descripció Detallada

template<class T>struct Arbre< T >::node_arbre

Definició a la línia 10 del fitxer Arbre.hpp.

5.4.2 Documentació de les Dades Membre

5.4.2.1 template<class T> T Arbre< T >::node_arbre::info

Definició a la línia 11 del fitxer Arbre.hpp.

5.4.2.2 template<class T> node_arbre* Arbre< T >::node_arbre::segE

Definició a la línia 12 del fitxer Arbre.hpp.

5.4.2.3 template<class T> node_arbre* Arbre< T >::node_arbre::segD

Definició a la línia 13 del fitxer Arbre.hpp.

La documentació d'aquesta estructura es va generar a partir del següent fitxer:

- Arbre.hpp

5.5 Referència de la Classe Organisme

És un conjunt de cèl·lules posades en un arbre.

Classes

- struct Celula

  Element bàsic de cada organisme.

Mètodes públics

- Organisme ()

  Constructora per defecte.
- ~Organisme ()

  Destructora per defecte.
- void estirar_organisme ()

  Modificadora que fa créixer l'organisme.
- void retallar_organisme ()

Modificadora que elimina totes les cèl·lules que no tenen cap fill.

- void reproduir_organisme (const Organisme &o1, const Organisme &o2)

  Modificadora que modifica l'organisme implícit per tal que es converteixi en un fill dels altres dos organismes.
- bool compatibles (const Organisme &o) const

  Consultora que retorna si dos organismes són compatibles o no.
- int consultar_tamany () const

  Consultora que retorna el tamany de l'organisme.
- bool es_mort () const

  Consultora que ens diu si el paràmetre implícit està mort.
- void llegir_organisme ()

  Funció per llegir un organisme.
- void escriure_organisme () const

  Funció per escriure un organisme.

Mètodes Privats Estàtics

- static void estirar_recursiu (Arbre< Celula > &a, int &max_id, Celula c, int &tamany)

  Funció recursiva per estirar un organisme.
- static void retallar_recursiu (Arbre< Celula > &a, int &tamany)

  Funció per retallar l'arbre d'un organisme.
- static int intersec_recursiu (Arbre< Celula > &a1, Arbre< Celula > &a2)

  Funció per calcular el tamany de la intersecció de dos arbres de manera recursiva.

Atributs Privats

- Arbre< Celula > cels

  Arbre on estan guardades totes les cèl·lules de l'organisme.
- bool retallat

  Variable que ens indica si un organisme ha estat retallat.
- int tamany

  Variable que indica el tamany de l'organisme.
- int max_id

  Identificador màxim de les cèl·lules de l'organisme.
- bool mort

  Variable que ens diu si un organisme és viu o mort.

5.5.1 Descripció Detallada

És un conjunt de cèl·lules posades en un arbre.

Definició a la línia 12 del fitxer Organisme.hpp.

5.5.2 Documentació del Constructor i el Destructor

5.5.2.1 Organisme::Organisme ( )

Constructora per defecte.

S'executa automàticament al declarar un organisme

Precondició

Cert

Postcondició

El resultat és amb una cèl·lula activa i tamany '0'

Definició a la línia 10 del fitxer Organisme.cpp.

```
11 {
12     Arbre<Celula> a1, a2;
13     Celula c;
14     c.id = 1;
15     c.activa = true;
16     tamany = 1;
17     max_id = 1;
18   mort = false;
19     retallat = false;
20
21     cels.plantar(c, a1, a2);
22 }
```

5.5.2.2 Organisme::~Organisme ( )

Destructora per defecte.

Definició a la línia 24 del fitxer Organisme.cpp.

```
24 {}
```

5.5.3 Documentació de les Funcions Membre

5.5.3.1 void Organisme::estirar_recursiu (Arbre< Celula > & *a,* int & *max_id,* Celula *c,* int & *tamany*) [static], [private]

Funció recursiva per estirar un organisme.

Precondició

'c' és una cèl·lula vàlida, max_id està inicialitzat i no

Postcondició

Totes les cèl·lules que no s'havien dividit s'han dividit

Definició a la línia 43 del fitxer Organisme.cpp.

```
45 {
46     Arbre<Celula> a1, a2;
47     if (a.es_buit()) {
48         ++max_id;
49         ++tam;
50         a.plantar(c, a1, a2);
51     }
52     else {
53         c = a.arrel();
54         a.fills(a1, a2);
55         estirar_recursiu(a1, max_id, c, tam);
56         estirar_recursiu(a2, max_id, c, tam);
57         a.plantar(c, a1, a2);
58     }
59 }
```

5.5.3.2 void Organisme::retallar_recursiu (Arbre< Celula > & *a,* int & *tamany*) [static], [private]

Funció per retallar l'arbre d'un organisme.

Precondició

L'organisme no està mort

Postcondició

Totes les cèl·lules que no tenen cap filla han estat eliminades

Definició a la línia 80 del fitxer Organisme.cpp.

```
81 {
82   if(not a.es_buit()) {
83         Arbre<Celula> a1, a2;
84     Celula c = a.arrel();
85     a.fills(a1, a2);
86
87     // Si algun dels dos fills no està buit vol dir que la cèl·lula encara
88     // no s'ha d'eliminar. Si ja no té cap fill no tornem a plantar
89         // l'arbre i haurem eliminat la cèl·lula.
90     if(not(a1.es_buit() and a2.es_buit())) {
91       retallar_recursiu(a1, tam);
92       retallar_recursiu(a2, tam);
93       a.plantar(c, a1, a2);
94     }
95     else {
96       --tam;
97     }
98   }
99 }
```

5.5.3.3 int Organisme::intersec_recursiu (Arbre< Celula > & *a1,* Arbre< Celula > & *a2*) [static], [private]

Funció per calcular el tamany de la intersecció de dos arbres de manera recursiva.

Precondició

Cert

Postcondició

Retorna el nombre d'elements de l'arbre resultant de la intersecció dels dos arbres 'a1' i 'a2'

Definició a la línia 120 del fitxer Organisme.cpp.

```
121 {
122   int res = 0;
123
124     // Evaluem per cada branca de l'arbre, si hi ha una arrel sumem 1
125   // com que ho fem recursivament tots els resultats es van sumant fins
126   // a obtenir el resultat de la intersecció
127   if(not(aA.es_buit()) and not(aB.es_buit())) {
128     ++res;
129         Arbre<Celula> aA1, aA2, aB1, aB2;
130     aA.fills(aA1, aA2);
131     aB.fills(aB1, aB2);
132     res += intersec_recursiu(aA1, aB1);
133     res += intersec_recursiu(aA2, aB2);
134   }
135   return res;
136 }
```

5.5.3.4 void Organisme::estirar_organisme ( )

Modificadora que fa créixer l'organisme.

Precondició

L'organisme ha de tenir una cèl·lula o més

Postcondició

Fisiona totes les cèl·lules de l'organisme que no s'hagin fisionat

Definició a la línia 31 del fitxer Organisme.cpp.

```
32 {
33   if (not retallat) {
34         Celula c = cels.arrel();
35         Arbre<Celula> a1, a2;
36         cels.fills(a1, a2);
37     estirar_recursiu(a1, max_id, c, tamany);
38     estirar_recursiu(a2, max_id, c, tamany);
39         cels.plantar(c, a1, a2);
40   }
41 }
```

5.5.3.5 void Organisme::retallar_organisme ( )

Modificadora que elimina totes les cèl·lules que no tenen cap fill.

Precondició

L'organisme ha de tenir una cèl·lula o més

Postcondició

Totes les cèl·lules que no tenien cap fill han estat eliminades

Definició a la línia 61 del fitxer Organisme.cpp.

```
62 {
63   if (not mort) {
64         retallat = true;
65         Celula c = cels.arrel();
66         Arbre<Celula> a1, a2;
67         cels.fills(a1, a2);
68     if(a1.es_buit() and a2.es_buit()) {
69       mort = true;
70       tamany = 0;
71     }
72     else {
73       retallar_recursiu(a1, tamany);
74       retallar_recursiu(a2, tamany);
75             cels.plantar(c, a1, a2);
76     }
77   }
78 }
```

5.5.3.6 void Organisme::reproduir_organisme (const Organisme & *o1*, const Organisme & *o2*)

Modificadora que modifica l'organisme implícit per tal que es converteixi en un fill dels altres dos organismes.

Precondició

'o1' i 'o2' han de ser dos organismes que no estiguin morts i han de ser compatibles entre ells

Postcondició

L'organisme implícit ha passat a ser un organisme que és fill de 'o1' i 'o2'

Definició a la línia 101 del fitxer Organisme.cpp.

```
102 {}
```

5.5.3.7 bool Organisme::compatibles (const Organisme & *o*) const

Consultora que retorna si dos organismes són compatibles o no.

Precondició

Cert

Postcondició

Retorna un booleà que és 'true' si són compatibles i 'false' si no ho són

Definició a la línia 109 del fitxer Organisme.cpp.

```
110 {
111   int comp = (tamany + o.tamany)/4;
112
113     Arbre<Celula> aA = cels;
114     Arbre<Celula> aB = o.cels;
115   // Variable amb la que es mirarà el tamany de l'intersecció
116   int intersec = intersec_recursiu(aA, aB);
117   return intersec >= comp;
118 }
```

5.5.3.8 int Organisme::consultar_tamany ( ) const

Consultora que retorna el tamany de l'organisme.

Precondició

Cert

Postcondició

Retorna un int amb el tamany de l'organisme

Definició a la línia 138 del fitxer Organisme.cpp.

```
139 {
140   return tamany;
141 }
```

5.5.3.9 bool Organisme::es_mort ( ) const

Consultora que ens diu si el paràmetre implícit està mort.

Precondició

Cert

Postcondició

Retorna un booleà que és 'true' si el paràmetre implícit és mort

Definició a la línia 143 del fitxer Organisme.cpp.

```
144 {
145   return mort;
146 }
```

5.5.3.10 void Organisme::llegir_organisme ( )

Funció per llegir un organisme.

Precondició

Cert

Postcondició

El paràmetre implícit ha passat a ser tal i com se li han donat pel canal estàndard d'entrada. El que hi havia abans en aquest organisme ha estat eliminat.

Definició a la línia 152 del fitxer Organisme.cpp.

```
153 {}
```

5.5.3.11 void Organisme::escriure_organisme ( ) const

Funció per escriure un organisme.

Precondició

L'organisme no ha d'estar mort

Postcondició

Escriu l'arbre que forma l'organisme, no s'escriu si les cèl·lules són actives o passives

Definició a la línia 155 del fitxer Organisme.cpp.

```
156 {}
```

5.5.4 Documentació de les Dades Membre

5.5.4.1 Arbre<Celula> Organisme::cels [private]

Arbre on estan guardades totes les cèl·lules de l'organisme.

Definició a la línia 30 del fitxer Organisme.hpp.

5.5.4.2 bool Organisme::retallat [private]

Variable que ens indica si un organisme ha estat retallat.

Definició a la línia 33 del fitxer Organisme.hpp.

5.5.4.3 int Organisme::tamany [private]

Variable que indica el tamany de l'organisme.

Definició a la línia 36 del fitxer Organisme.hpp.

5.5.4.4 int Organisme::max_id [private]

Identificador màxim de les cèl·lules de l'organisme.

Definició a la línia 39 del fitxer Organisme.hpp.

5.5.4.5 bool Organisme::mort [private]

Variable que ens diu si un organisme és viu o mort.

Definició a la línia 42 del fitxer Organisme.hpp.

La documentació d'aquesta classe es va generar a partir dels següents fitxers:

- Organisme.hpp
- Organisme.cpp

5.6 Referència de l'Estructura Ranking::OrganRank

Tipus de dades per poder fer el rking.

Atributs Públics

- int id

  Identificador de l'organisme.
- int fills

  Número de fills que ha tingut l'organisme.

5.6.1 Descripció Detallada

Tipus de dades per poder fer el rking.

Definició a la línia 21 del fitxer Ranking.hpp.

5.6.2 Documentació de les Dades Membre

5.6.2.1 Ranking::OrganRank::id

Identificador de l'organisme.

Definició a la línia 26 del fitxer Ranking.hpp.

5.6.2.2 Ranking::OrganRank::fills

Número de fills que ha tingut l'organisme.

Definició a la línia 31 del fitxer Ranking.hpp.

La documentació d'aquesta estructura es va generar a partir del següent fitxer:

- Ranking.hpp

5.7 Referència de l'Estructura Ranking::ParFill

Estructura per poder saber quins fills ha tingut un organisme i amb qui els ha tingut.

Atributs Públics

- int parella

  Retorna l'identificador del pare.
- int fill

  Retorna l'identificador del fill.

5.7.1 Descripció Detallada

Estructura per poder saber quins fills ha tingut un organisme i amb qui els ha tingut.

Definició a la línia 38 del fitxer Ranking.hpp.

5.7.2 Documentació de les Dades Membre

5.7.2.1 Ranking::ParFill::parella

Retorna l'identificador del pare.

Definició a la línia 43 del fitxer Ranking.hpp.

5.7.2.2 Ranking::ParFill::fill

Retorna l'identificador del fill.

Definició a la línia 48 del fitxer Ranking.hpp.

La documentació d'aquesta estructura es va generar a partir del següent fitxer:

- Ranking.hpp

5.8 Referència de la Classe PRO2Excepcio

Mètodes públics

- PRO2Excepcio (const char ∗mot)
- const char ∗ what () const throw ()

Atributs Privats

- const char ∗ mensaje

5.8.1 Descripció Detallada

Definició a la línia 10 del fitxer utils.PRO2.

5.8.2 Documentació del Constructor i el Destructor

5.8.2.1 PRO2Excepcio::PRO2Excepcio (const char * *mot*)

Definició a la línia 12 del fitxer utils.PRO2.

```
12 : exception(), mensaje(mot) {}
```

5.8.3 Documentació de les Funcions Membre

5.8.3.1 const char* PRO2Excepcio::what ( ) const throw)

Definició a la línia 13 del fitxer utils.PRO2.

```
13 {return mensaje;};
```

5.8.4 Documentació de les Dades Membre

5.8.4.1 const char* PRO2Excepcio::mensaje [private]

Definició a la línia 13 del fitxer utils.PRO2.

La documentació d'aquesta classe es va generar a partir del següent fitxer:

- utils.PRO2

5.9 Referència de la Classe Ranking

Classe Ranking per poder imprimir el ranking dels organismes.

Classes

- struct OrganRank
  Tipus de dades per poder fer el rking.
- struct ParFill
  Estructura per poder saber quins fills ha tingut un organisme i amb qui els ha tingut.

Mètodes públics

- Ranking (int M)
  Constructora per defecte.
- void afegir_fill (int pare1, int pare2, int fill)
  Modificadora que afegeix els pares de un organisme per poder fer el rking.
- void ranking () const
  Funció que imprimeix el rking.

Atributs Privats

- vector< OrganRank > Rank
  Vector que utilitzarem per generar i guardar el rking.
- vector< list< ParFill > > Rel
  Vector per saber quins fills ha tingut cada organisme i amb qui els ha tingut.

5.9.1 Descripció Detallada

Classe Ranking per poder imprimir el ranking dels organismes.

Definició a la línia 15 del fitxer Ranking.hpp.

5.9.2 Documentació del Constructor i el Destructor

5.9.2.1 Ranking::Ranking (int *M*)

Constructora per defecte.

Precondició

Cert

Postcondició

Es crea un ranking de tamany 'M'

Definició a la línia 11 del fitxer Ranking.cpp.

```
12 {
13     Rank = vector<OrganRank> (M);
14     Rel = vector< list<ParFill> > (M);
15 }
```

5.9.3 Documentació de les Funcions Membre

5.9.3.1 void Ranking::afegir_fill (int *pare1*, int *pare2*, int *fill*)

Modificadora que afegeix els pares de un organisme per poder fer el rking.

Precondició

Cert

Postcondició

S'han afegit l'ID dels pares i dels fills al Ranking

Definició a la línia 21 del fitxer Ranking.cpp.

```
22 {
23     ParFill aux;
24     iterator::list<ParFill> it;
25 
26     aux.parella = pare1;
27     aux.fill = fill;
28     it = Rel[pare1 - 1].end();
29     Rel[pare1 - 1].insert(it, aux);
30 
31     aux.parella = pare2;
32     it = Rel[pare2 - 1].end();
33     Rel[pare2 - 1].insert(it, aux);
34 }
```

5.9.3.2 void Ranking::ranking ( ) const

Funció que imprimeix el rking.

Precondició

Hi ha com a mínim un organisme

Postcondició

Pel can estdard de sortida s'ha imprès el rking de reproducció dels organismes

5.9.4 Documentació de les Dades Membre

5.9.4.1 vector<OrganRank> Ranking::Rank [private]

Vector que utilitzarem per generar i guardar el rking.

Definició a la línia 52 del fitxer Ranking.hpp.

5.9.4.2 vector< list<ParFill> > Ranking::Rel [private]

Vector per saber quins fills ha tingut cada organisme i amb qui els ha tingut.

Definició a la línia 57 del fitxer Ranking.hpp.

La documentació d'aquesta classe es va generar a partir dels següents fitxers:

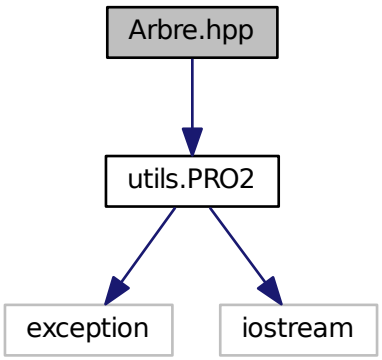
- Ranking.hpp
- Ranking.cpp

Capítol 6

Documentació dels Fitxers

6.1 Referència del Fitxer Arbre.hpp

Inclou el graf de dependències per a Arbre.hpp:



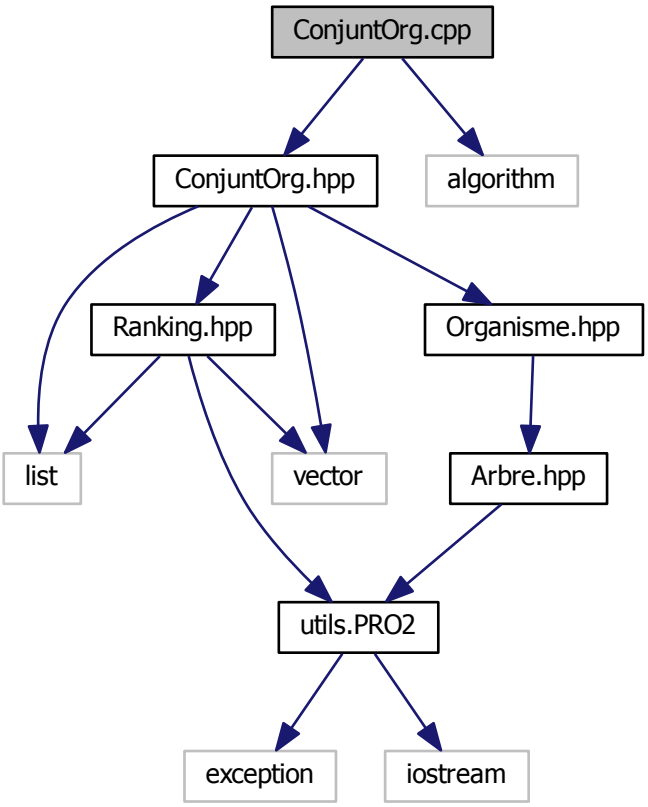
Classes

- class Arbre< T >
- struct Arbre< T >::node_arbre

6.2 Referència del Fitxer ConjuntOrg.cpp

Implementació de la classe ConjuntOrg.

Inclou el graf de dependències per a ConjuntOrg.cpp:



6.2.1 Descripció Detallada

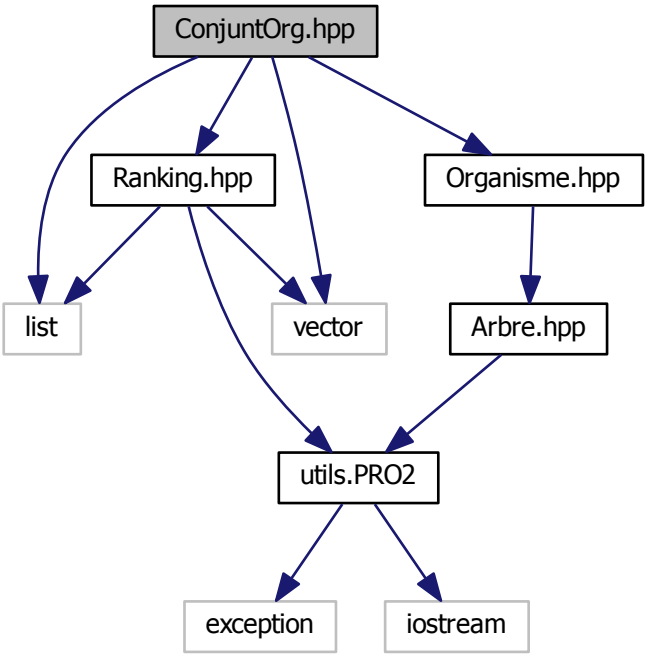
Implementació de la classe ConjuntOrg.

Definició al fitxer ConjuntOrg.cpp.

6.3 Referència del Fitxer ConjuntOrg.hpp

Especificació de la classe ConjuntOrg.

Inclou el graf de dependències per a ConjuntOrg.hpp:



Classes

- class ConjuntOrg

  És un conjunt d'organismes.

6.3.1 Descripció Detallada

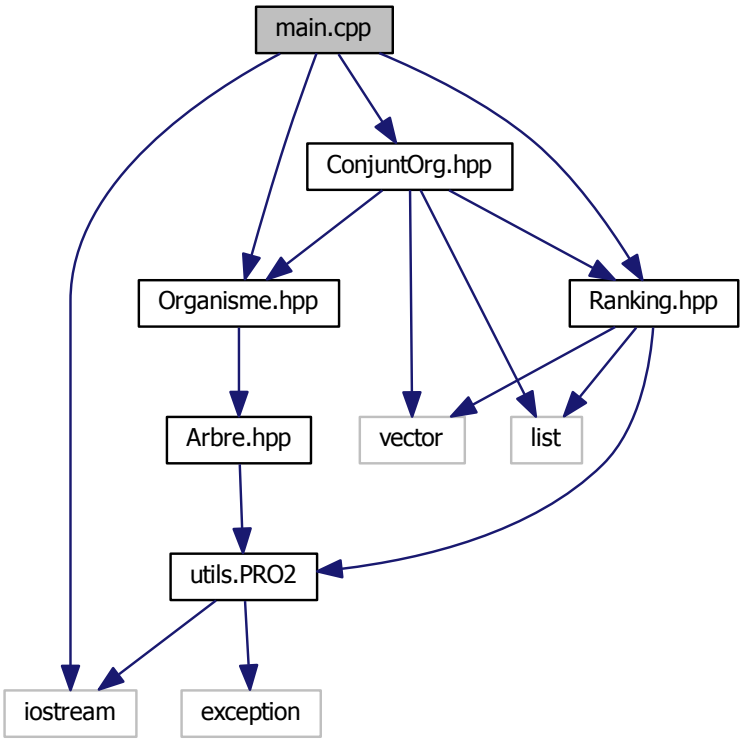
Especificació de la classe ConjuntOrg.

Definició al fitxer ConjuntOrg.hpp.

6.4 Referència del Fitxer main.cpp

Programa principal per a la pràctica.

Inclou el graf de dependències per a main.cpp:



Definicions

- #define MARCA -1

Funcions

- int main ()

  Programa principal de la Pràctica de PRO2.

6.4.1 Descripció Detallada

Programa principal per a la pràctica.

Definició al fitxer main.cpp.

6.4.2 Documentació de les Definicions

6.4.2.1 #define MARCA -1

Definició a la línia 18 del fitxer main.cpp.

6.4.3 Documentació de les Funcions

6.4.3.1 int main ( )

Programa principal de la *Pràctica de PRO2*.

Definició a la línia 22 del fitxer main.cpp.

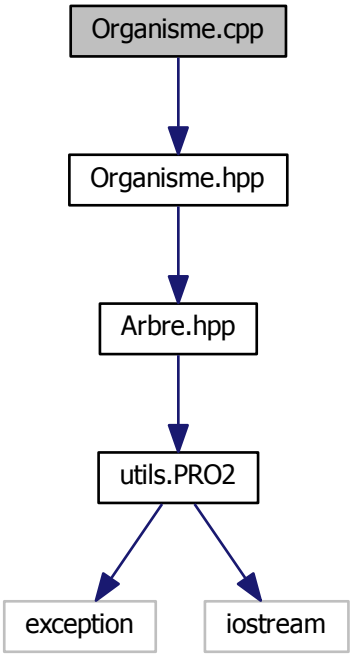
```
23 {
24   // M És el màxim històric permés
25   // N És el nombre d'organismes inicials
26   int N, M;
27   cin >> N >> M;
28
29   // Conjunt que ens permetrà guardar tots els organismes existents
30   ConjuntOrg Conj(M);
31     Ranking Rank(M);
32
33     // Cridem la funicó per llegir un conjunt d'organismes de la classe
34     // ConjuntOrg
35     Conj.llegir();
36
37   // Variable per seleccionar la opció d'entrada
38   int x;
39
40   /*  Variable de tipus int que quan sigui diferent de '0' farà
41     acabar l'experiment, un número diferent de 0 indicarà el motiu pel
42     qual s'acaba l'experiment:
43     - 1 => Tots els organismes han mort
44     - 2 => S'ha arribat al límit d'organismes
45     - 3 => S'ha donat per finalitzat l'experiment manualment  */
46   int fi = 0;
47     cin >> x;
48   while (x != MARCA and fi == 0) {
49     // Opció per estirar un conjunt d'organismes
50         if (x == 1) {
51             int a;
52
53             cin >> a;
54       while(a != MARCA) {
55                 Conj.estirar(a);
56         cin >> a;
57             }
58     }
59
60     // Opció per retallar un conjunt d'organismes
61         else if (x == 2) {
62             int a;
63
64             cin >> a;
65       while(a != MARCA) {
66                 Conj.retallar(a);
67         cin >> a;
68             }
69             if (Conj.morts()) fi = 1;
70     }
71
72     // Aplicar una ronda de reproducció a TOTS els organismes, actualitzar
73     // el rànking i imprimir els fills nascuts a la ronda
74     else if (x == 3) {
75             int fills;
76             if (not Conj.reproduir(Rank, fills)) {
77               fi = 2;
78               x = fills;
79             }
80             cout << fills << endl;
81     }
82
83     // Obtenir el rànking de reproducció dels organismes
84     else if (x == 4) {
85             Rank.ranking();
86     }
87
88     // Consultar l'estat d'un subconjunt d'organismes
89         else if (x == 5) {
90             int a;
91
92             cin >> a;
93             while(a != MARCA) {
94                 Conj.estat(a);
95                 cin >> a;
96             }
97     }
98         cin >> x;
```

```
99   }
100
101     // Instruccions per a la fi del programa
102     if (fi == 1) {
103         cout << "FI: Tots els organismes han mort" << endl;
104     }
105     else if (fi == 2) {
106         Rank.ranking();
107         Conj.escriure_ultims(x);
108     }
109 }
```

6.5 Referència del Fitxer Organisme.cpp

Implementació de la classe Organisme.

Inclou el graf de dependències per a Organisme.cpp:



6.5.1 Descripció Detallada

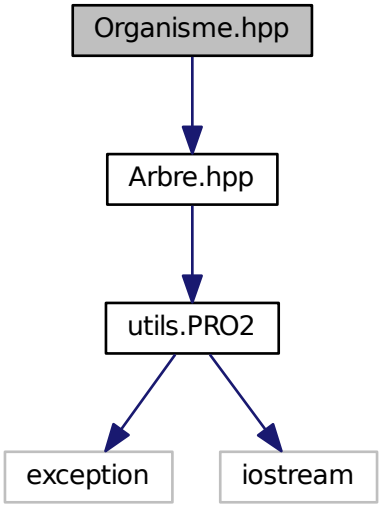
Implementació de la classe Organisme.

Definició al fitxer Organisme.cpp.

6.6 Referència del Fitxer Organisme.hpp

Especificació de la classe Organisme.

Inclou el graf de dependències per a Organisme.hpp:



Classes

- class Organisme

  És un conjunt de cèl·lules posades en un arbre.

- struct Organisme::Celula

  Element bàsic de cada organisme.

6.6.1 Descripció Detallada

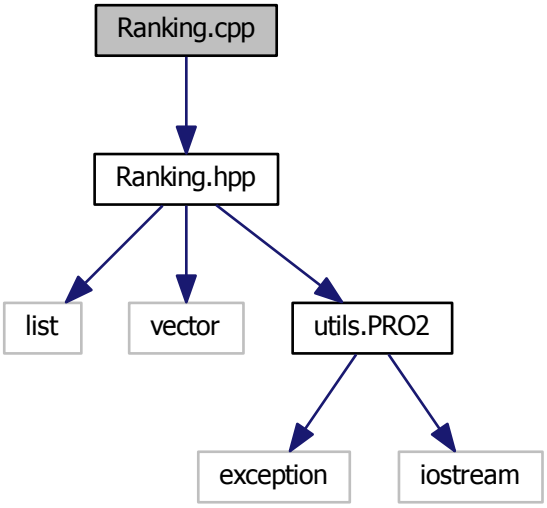
Especificació de la classe Organisme.

Definició al fitxer Organisme.hpp.

6.7 Referència del Fitxer Ranking.cpp

Implementació de la classe Ranking.

Inclou el graf de dependències per a Ranking.cpp:



6.7.1 Descripció Detallada

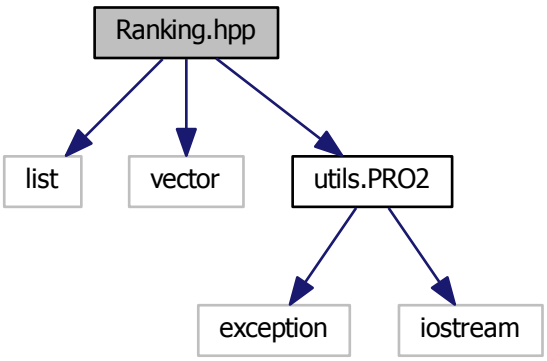
Implementació de la classe Ranking.

Definició al fitxer Ranking.cpp.

6.8 Referència del Fitxer Ranking.hpp

Especificació de la classe Ranking.

Inclou el graf de dependències per a Ranking.hpp:



Classes

- class Ranking

  Classe Ranking per poder imprimir el ranking dels organismes.
- struct Ranking::OrganRank

  Tipus de dades per poder fer el rking.
- struct Ranking::ParFill

  Estructura per poder saber quins fills ha tingut un organisme i amb qui els ha tingut.

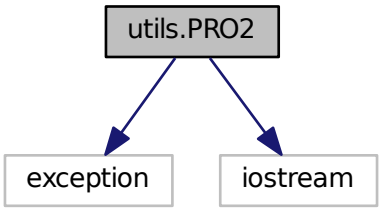
6.8.1 Descripció Detallada

Especificació de la classe Ranking.

Definició al fitxer Ranking.hpp.

6.9 Referència del Fitxer utils.PRO2

Inclou el graf de dependències per a utils.PRO2:



Classes

- class PRO2Excepcio

Definicions

- #define UTILS_PRO2

Funcions

- int readint ()
- char readchar ()
- bool readbool ()
- double readdouble ()
- string readstring ()

6.9.1 Documentació de les Definicions

6.9.1.1 #define UTILS_PRO2

Definició a la línia 2 del fitxer utils.PRO2.

6.9.2 Documentació de les Funcions

6.9.2.1 int readint ( )

Funcions per fer lectures de tipus basics.

Definició a la línia 23 del fitxer utils.PRO2.

```
24 {
25   int n;
26   cin >> n;
27   return n;
28 }
```

6.9.2.2 char readchar ( )

Definició a la línia 30 del fitxer utils.PRO2.

```
31 {
32   char n;
33   cin >> n;
34   return n;
35 }
```

6.9.2.3 bool readbool ( )

Definició a la línia 38 del fitxer utils.PRO2.

```
39 {
40   string n;
41   cin >> n;
42   if (n!="true" and n!="false") throw PRO2Excepcio("S'havia de llegir un boolea");
43   return (n=="true");
44 }
```

6.9.2.4 double readdouble ( )

Definició a la línia 46 del fitxer utils.PRO2.

```
47 {
48   double n;
49   cin >> n;
50   return n;
51 }
```

6.9.2.5 string readstring ( )

Definició a la línia 53 del fitxer utils.PRO2.

```
54 {
55   string s;
56   cin >> s;
57   return s;
58 }
```