Disseny modular II
Exemples

19 de març de 2018

1 Exemple d'especificació: classe `Cjt_estudiants`

```
#include "Estudiant.hh"

class Cjt_estudiants {

  // Representa un conjunt ordenat per DNI d'estudiants.
  // Es poden consultar i modificar els seus elements
  // de tipus Estudiant donat un DNI o per posició en l'ordre.

  private:

  public:

  // Constructores

  Cjt_estudiants();
  /* Pre: cert */
  /* Post: el resultat és un conjunt d'estudiants buit */

  // Destructora

  ~Cjt_estudiants();

  // Modificadores
```

```
void afegir_estudiant(const Estudiant &est);
/* Pre: el paràmetre implícit no conté cap estudiant amb el dni d'est; el
   nombre d'estudiants del p.i. és menor que la mida màxima permesa */
/* Post: s'ha afegit l'estudiant est al paràmetre implícit */

void modificar_estudiant(const Estudiant &est);
/* Pre: existeix un estudiant al paràmetre implícit amb el DNI d'est  */
/* Post: l'estudiant del paràmetre implícit original amb el dni
   d'est, ha quedat substituït per est */

void modificar_iessim(int i, const Estudiant &est);
/* Pre: 1 <= i <= nombre d'estudiants del paràmetre implícit,
   l'element i-èssim del conjunt en ordre creixent per DNI conté
   un estudiant amb el mateix DNI que est */
/* Post: l'estudiant est ha substituït l'estudiant i-èssim
   del paràmetre implícit */

// Consultores

int mida() const;
/* Pre: cert */
/* Post: el resultat és el nombre d'estudiants del paràmetre implícit */

static int mida_maxima();
/* Pre: cert */
/* Post: el resultat és el nombre màxim d'estudiants que pot arribar
   a tenir el paràmetre implícit */

bool existeix_estudiant(int dni) const;
/* Pre: dni > 0 */
/* Post: el resultat indica si existeix un estudiant al paràmetre implícit
   amb DNI = dni */

Estudiant consultar_estudiant(int dni) const;
/* Pre: existeix un estudiant al paràmetre implícit amb DNI = dni */
/* Post: el resultat és l'estudiant amb DNI = dni que conté el
   paràmetre implícit */
```

```
  Estudiant consultar_iessim(int i) const;
  /* Pre: 1 <= i <= nombre d'estudiants que conté el paràmetre implícit */
  /* Post: el resultat és l'estudiant i-èssim del paràmetre implícit
     en ordre creixent per DNI */

  // Lectura i escriptura

  void llegir();
  /* Pre: estan preparats al canal estàndard d'entrada un enter entre 0 i
     la mida maxima permesa, que representa el nombre d'elements que llegirem,
     i les dades de tal nombre d'estudiants diferents */
  /* Post: el paràmetre implícit conté el conjunt d'estudiants llegits
     del canal estàndard d'entrada */

  void escriure() const;
  /* Pre: cert */
  /* Post: s'han escrit pel canal estàndard de sortida els estudiants del
     paràmetre implícit en ordre ascendent per DNI */
};
```

2 Implementació classe Estudiant

2.1 Estudiant.hh

```
class Estudiant {

private:
  int dni;
  double nota;
  bool amb_nota;
  static const int MAX_NOTA = 10;

  /* Invariant de la representació:
     - 0 <= dni
     - si amb_nota, llavors 0 <= nota <= MAX_NOTA
  */
```

```
public:

  /* Constructores */
  Estudiant();
  Estudiant(int dni);

  /* Destructora por defecte */
  ~Estudiant();

  /* Modificadores */
  void afegir_nota(double nota);
  void modificar_nota(double nota);

 /* Consultores */
  bool te_nota() const;
  double consultar_nota() const;
  int consultar_DNI() const;
  static double nota_maxima();

  /* Entrada / Sortida */
  void llegir();
  void escriure() const;
};
```

2.2 Estudiant.cc

```
Estudiant::Estudiant()
/* Pre: cert */
{
    dni = 0;
    amb_nota = false;
}
/* Post: el resultat és un estudiant amb DNI=0 i sense nota */

Estudiant::Estudiant(int dni)
/* Pre: dni>=0 */
{
    this->dni = dni;
    amb_nota = false;
}
```

```
/* Post: el resultat és un estudiant amb DNI=dni i sense nota */

Estudiant::~Estudiant(){}
// esborra automàticament els objectes locals en sortir d'un àmbit de visibilitat

void Estudiant::afegir_nota(double nota)
/* Pre: el paràmetre implícit no té nota i 0<=nota<=nota_maxima() */
{
    (*this).nota = nota;
    /* una notació equivalent alternativa és this->nota = nota; */
    amb_nota = true;
}
/* Post: la nota del paràmetre implícit passa a ser nota */

void Estudiant::modificar_nota(double nota)
/* Pre: el paràmetre implícit té nota i 0<=nota<=nota_maxima() */
{
    this->nota = nota;
}
/* Post: la nota del paràmetre implícit passa a ser nota */

int Estudiant::consultar_DNI() const
/* Pre: cert  */
{
    return dni;
}
/* Post: el resultat és el DNI del paràmetre implícit  */

double Estudiant::consultar_nota() const
/* Pre: el paràmetre implícit té nota */
{
    return nota;
}
/* Post: el resultat és la nota del paràmetre implícit */

bool Estudiant::te_nota() const
/* Pre: cert  */
{
    return amb_nota;
}
```

```
/* Post: el resultat indica  si el paràmetre implícit té nota o no */

double Estudiant::nota_maxima() // la paraula clau static del .hh no s'ha de repetir a
/* Pre: cert */
{
return MAX_NOTA;
}
/* Post: el resultat és la nota màxima dels elements de la classe */

void Estudiant::llegir()
/* Pre: estan preparats al canal estandar d'entrada un enter no negatiu
   i un double */
{
    cin >> dni;
    double x;
    cin >> x;
    if (x >= 0 and x <= MAX_NOTA) afegir_nota(x);
    else amb_nota = false;
}
/* Post: el paràmetre implícit passa a tenir els atributs llegits
   del canal estàndard d'entrada; si el double no és una nota
   entre 0 i nota_maxima(), el p.i. es queda sense nota */

void Estudiant::escriure() const
/* Pre: cert */
{
    if (amb_nota) cout << dni << " " << nota << endl;
    else cout << dni << " NP" << endl;
}
/* Post: s'han escrit els atributs del paràmetre implícit
   al canal estàndard de sortida */
```