Capítol 3

Estructures de dades lineals i arborescens

3.1 Estructures de dades lineals

Una estructura de dades és lineal si els seus elements formen una seqüència. Podem representar l'estructura com

$$e_1, e_2, \cdots, e_n$$

on $n \geq 0$. Si $n = 0$, l'estructura està buida. Si $n = 1$, l'estructura té un element que és al mateix temps el primer i l'últim, i no té ni antecessor ni successor. Si $n = 2$, hi ha dos elements, el primer element, sense antecessor i que té com a successor el segon, i el segon element, sense successor i que té com a antecessor el primer. Si $n > 2$, tots els elements a partir del segon fins al penúltim tenen antecessor i successor.

A classe de teoria i laboratori es presentaran les estructures lineals pila (`stack`), cua (`queue`) i llista (`list`). També farem servir la estructura lineal `vector`. Les diferents estructures de dades lineals es diferencien per com es pot accedir als seus elements.

3.2 Tipus genèrics o parametritzats de dades

El mecanisme de *templates* de C++ permet fer servir tipus de dades com a paràmetres. D'aquesta forma, un tipus de dades pot aparèixer com a paràmetre en la definició d'una classe o mètode. Es pot, per tant, treballar amb estructures de dades genèriques, on podem guardar qualsevol tipus d'element, donat que aquest tipus és un paràmetre. En el moment de crear l'estructura només caldrà *instanciar* el tipus d'element que hi guardarem.

En aquest capítol no implementarem estructures de dades genèriques, sino que farem servir estructures genèriques ja implementades (concretament, `vector<T>`, `pair<T1,T2>`, `stack<T>`, `queue<T>`, `list<T>` i `tree<T>` on `T`,`T1` i `T2` són paràmetres que representen tipus de dades). Per fer servir correctament aquestes estructures ens cal conèixer una sèrie de recomanacions i saber com fer la instanciació (e.g. `vector<Estudiant>` o `vector<bool>`).

Els vectors són un exemple d'estructura de dades genèrica que ja coneixem. En el seu cas, es pot consultar qualsevol element sense necessitat de modificar-lo, però és important tenir en

compte, com veurem més endavant als exemples, que per algunes d'aquestes estructures, consultar tots els elements implica buidar l'estructura d'elements (e.g. `stack` i `queue`). Per això, si volem conservar l'estructura inicial, caldrà fer una còpia. Normalment, per les estructures genèriques que farem servir als nostres programes, i també pels objectes continguts dintre de les estructures, podrem fer servir l'operador d'assignació = per fer còpies. En els casos en que no sigui així es farà constar i es proporcionarà algun mètode alternatiu per fer còpies.

Una conseqüència de treballar amb estructures genèriques és que no pot ser genèric cap mètode que depengui de la classe dels components. Cap mètode genèric pot tenir en el seu codi ni lectures, ni escriptures dels elements, perquè encara no se sap a quina classe pertanyeran aquests elements. Això obliga a crear operacions fora de la classe genèrica per fer les operacions de lectura i escriptura d'estructures que continguin elements d'una classe concreta. Per cada classe d'elements, haurem d'implementar operacions de lectura i escriptura per l'estructura de dades contenidora. Aquestes operacions no són de la classe i no tenen paràmetre implícit.