Pràctica 1: Fitxers d'accés directe

Bases de Dades — iTIC

Marta Tarrés-Puertas
Sebastià Vila-Marta

19 de febrer de 2015

Índex

1 Organització

1.1 Material necessari

Simplement us cal un computador amb sistema operatiu GNU/Linux i connexió a xarxa. Pel que fa a la documentació de referència, les pàgines de man i les dues referències següents us poden ser útils:

1.2 Lliurament

Cal lliurar la pràctica en un tarfile a través d'Atenea en la data fixada.

2 Introducció

En anteriors assignatures s'han introduït els fitxers i moltes de les seves peculiaritats però sempre en la seva modalitat d'accés seqüencial. Això no obstant segons el medi físic on s'emmagatzemen els fitxers a vegades són possibles altres modes d'accés a les seves dades. De particular importància és el mode d'*accés directe*. En aquest mode d'accés un fitxer es percep com una seqüència de dades amb la capacitat de situar el cursor de lectura/escriptura en qualsevol posició de la seqüència sense haver de passar obligatòriament per les anteriors. Sovint aquesta operació de posicionament s'anomena *seek*. Els fitxers emmagatzemats al disc tenen típicament la possibilitat de ser accedits en mode directe.

L'objectiu d'aquesta pràctica és explorar aquest règim d'accés als fitxers, que configura la tècnica bàsica per construir estructures de la informació complexes hostatjades en fitxers (i no en memòria central).

3 L'API Python pels fitxers d'accés directe

Els fitxers de `Python` disposen d'operacions per a poder accedir, si el medi físic ho permet, en mode directe. Entre aquestes operacions es troben: seek(), tell() i truncate().

TASCA 1 Llegiu la referència [con13][5.9 File Objects] i estudieu aquestes operacions. Noteu com, en un fitxer d'accés directe obert en mode lectura/escriptura és possible llegir i escriure en qualsevol lloc del fitxer.

Per exercitar l'ús d'aquestes operacions us proposem el següent exercici. Suposeu que una bateria de sensors d'un aparcament detecta les matrícules dels cotxes que s'aparquen a cada plaça. El sensor pot enviar dos tipus de missatges amb els següents formats:

102 EMPTY Indica que la plaça número 102 s'ha buidat i no l'ocupa ningú.

102 FULL 2310AMN Indica que la plaça número 102 l'ha ocupada un vehicle amb matrícula 2310AMN.

TASCA 2 Dissenyeu i implementeu un petit programa `Python` que interactua amb l'usuari i li permet:

- Rebre notificacions d'ocupació i buidat de places, en particular, ha de permetre,
  - Ocupar una plaça concreta
  - Ocupar la primera plaça buida
  - Sortir del pàrquing
- Consultar l'estat de qualsevol plaça.
- Llistar les places buides.
- Determinar si un vehicle concret és a l'aparcament i, si és el cas, quina plaça ocupa.
- Sortir de l'aplicació.

Assumiu que les places de l'aparcament han d'estar numerades entre 0 i 1000.

Naturalment l'aplicació ha de conservar l'estat entre execucions i per tant les dades d'ocupació han de ser emmagatzemades en un fitxer. L'estat d'ocupació de les places s'ha d'emmagatzemar en un fitxer de nom `'places.dat'` al que s'ha d'accedir en mode d'accés directe. Inicialment totes les places són buides.

L'organització interna del fitxer ha de ser tal que la matrícula de la plaça 0 ha d'ocupar els bytes $0, 1, \ldots, 6$ del fitxer, la matrícula de la plaça 1 els bytes $7, \ldots, 13$ i així successivament. Això us permet accedir directament a la matrícula d'una plaça concreta. En cas que una plaça estigui buida, hi emmagatzemeu la matrícula falsa `'XXXXXXX'`, que fa de marca.

Per a la interacció amb l'usuari implementeu un petit intèrpret d'ordres que faciliti la feina. L'arquitectura de l'aplicació és responsabilitat vostra i és un dels criteris de valoració del resultat.

4 Registres i empaquetat

Sovint el contingut d'un fitxer s'organitza en base a *registres*. Un registre podríem pensar que és una unitat de transferència lògica, i.e. el conjunt de dades interrelacionades que sempre s'escriuen o llegeixen al fitxer de forma atòmica[1]. En el cas del programa anterior el registre coincideix amb una matrícula. Moltes vegades, però, un registre conté diverses dades relacionades que anomenem *camps*. Imaginem, per exemple, que d'un vehicle aparcat no només se'n guarda la matrícula sinó que també se'n guarda el color i el model. Aleshores el registre fóra una agrupació d'aquests tres camps[2].

Les dades que formen part d'un registre moltes vegades tenen una mida variable. Aquest és el cas, per exemple dels camps de tipus cadena de caràcters. Això fa que el registre que les ha de contenir hagi de tenir mida variable. Una practica corrent consisteix a «empaquetar» les dades que formen un registre en un espai de mida fixa. Aquesta estratègia permet gestionar més fàcilment una col·lecció de registres emmagatzemats en un fitxer atès que tenen una longitud fixa.

En el cas de `Python` els registres es representen sobre una cadena de caràcters de mida fixa. Per tal de poder «desar» o «recuperar» els diferents camps encabits en un registre la llibreria de `Python` ofereix un servei per empaquetar/desempaquetar registres que és convenient conèixer.

Tasca 3 Estudieu aquesta referència [con13] [7.3. struct—Interpret strings as packed binary data] en la que s'explica el servei d'empaquetat que ofereix la llibreria de `Python` . Observeu que hi ha dues variants del mateix servei, una en base a funcions i una altra en base a la classe `struct`. Mireu d'entendre quines diferències hi ha i quan és útil una o altra variant.

Feu algunes proves amb petits programes que us ajudin a entendre com funciona l'empaquetat/desempaquetat de registres.

Amb el coneixements referents a l'empaquetat que heu adquirit ara podeu sofisticar una mica més el programa que heu escrit anteriorment.

Tasca 4 Suposeu ara que a l'exemple de l'aparcament els sensors envien més informació. Assumiu que el sensor pot enviar dos tipus de missatges amb els següents formats:

102 EMPTY Indica que la plaça número 102 s'ha buidat i no l'ocupa ningú.

102 FULL 2310AMN VERD TATA Indica que la plaça número 102 l'ha ocupada un vehicle amb matrícula 2310AMN, de color VERD i de marca TATA.

així, a més d'emmagatzemar dades sobre la matrícula del vehicle que ocupa una plaça, ara caldrà:

- Emmagatzemar el color del vehicle.
- Emmagatzemar la marca del vehicle.

Amplieu l'aplicació per encabir aquesta informació. Suposeu que tant el color com la marca es representen amb cadenes de caracters.

Naturalment cal que useu el concepte de registre i que emmagatzemeu les dades en el fitxer agrupades en registres.

[1] Fitxeu-vos que la unitat de transferència física és el bloc i té un sentit similar al registre però a nivell d'accés físic al perifèric

[2] Noteu que un registre també té una certa similitut amb el concepte de tuple a `Python`. Això no obstant es parla de registre quan ens referim específicament a les dades d'un fitxer.

Referències

[con13] Python project contributors. The Python Standard Library. Anglès. Versió 2.7.5. Python Software Foundation. 2015. url: http://docs.python.org/2/library (consultat 19 de feb. de 2015).