Llenguatges de Programació

Josep Carmona i Albert Rubio

Especialitat de Computació
Grau en Enginyeria Informàtica

FIB

Pla de la sessió

- Presentacció de l'assignatura
- Continguts
- Sessions sobre Compilació
- Introducció als llenguatges de programació

Presentació

Objectius

- Coneixer millor els lleguatges de programació
- Tenir coneixements bàsics sobre compilació
- Coneixer els llenguatges funcionals
- Coneixer construccions avançades
- Coneixer bàsicament el llenguatges d'scripting
- Millorar la capacitat d'apendre nous llenguatges
- Millorar la capacitat de triar el llenguatge

Presentació

Mètode d'avaluació
3 proves amb 3 treballs pràctics associats

1. Nota1 = 70%Prova CL + 30%Treball pràctic PCL
2. Nota2 = 70%Parcial EP + 30%Treball pràctic PEP
3. Nota3 = 90%Final EF + 10%Treball pràctic PEF

Treball dirigit: s'assignarà un llenguatge a cada estudiant

1. TC1 avaluarà els coneixements adquirits.
2. TC2 avaluarà la competència de G4.3 (Comunicació).
3. TC3 avaluarà la competència de G6.3 (Informació).

NOTA = 20%Nota1 + 30%Nota2 + 40%Nota3 + 10%Nota4

Continguts

1. Introducció als llenguatges de programació.
2. Introducció a la compilació.
3. Llenguatges funcionals.
4. Sistemes de tipus.
5. Programació d'ordre superior.
6. Especificació i modelat usant llenguatges funcionals.
7. Llenguatges de scripting.

Sessions Compilació

Les impartirà Josep Carmona

Aquest quadrimestre seran cap a final de curs.

Possiblement a partir del Dimecres 14 de Nvembre.

Ja confirmarem el calendari de sessions.

Introducció als LP

- Turing completesa.
- Paradigmes de la programació.
- Compilat vs Interpretat.
- Sistemes de tipus.

Turing completesa

Turing completeness en anglès.

- Un llenguatge de programació és *Turing complet* si pot simular una Màquina de Turing (d'una sola cinta).
- Model de càlcul molt simple creat per Allan Turing que pot realitzar qualsevol còmput que un computador digital pugui realitzar.
- Una sola cinta infinita i un capçal que llegeix i modifica el contingut de la cel·la en curs. El capçal es pot desplaçar a la dreta i a l'esquerra.

Aquest any és *l'any Turing*, centenari del seu nexament. (Més detalls a TC.)

Turing completesa

Model alternatiu *lambda càlcul* d'Alonzo Church.

Model funcional molt simple equivalent a la Màquina de Turing.

Tesi de Church-Turing mostrada pels dos independentment: Tot el que és algorísmicament computable és computable

- amb una Màquina de Turing o
- amb una funció en lambda càlcul.

Si tot es pot calcular amb un model tan simple, perquè hi ha tans lleguatges de programació?

- Eficiència
- Facilitat d'escriptura/llegibilitat
- ...

Turing completesa

Alguns autors consideren només com a lleguatges de programació els llenguatges Turing complets.

Exemples de llenguatges no Turing complets:

- expressions regulars (p.e. a Perl o a AWK).
- SQL.

Per ser Turing complet només cal tenir salts condicionals (bàsicament, **if** i **goto**).

Paradigmes de la programació

Hi ha una gran varietat de llenguatges.

- TIOBE index
- Language popularity lists

Com classificar-los?

Diferents *paradigmes o estils de programació*.

Paradigmes de la programació

Paradigmes més importants:

- Imperatiu o procedural
- Funcional
- Lògic (declaratiu)
- Orientat a objectes

Paradigmes de la programació

Paradigmes més importants:

- Imperatiu o procedural
- Funcional
- Lògic (declaratiu)
- Orientat a objectes

Altres paradigmes:

- Paral·lel/Concurrent
- Reflexiu
- .....

Paradigmes de la programació

Paradigmes més importants:

- Imperatiu o procedural
- Funcional
- Lògic (declaratiu)
- Orientat a objectes

Altres paradigmes:

- Paral·lel/Concurrent
- Reflexiu
- .....

Hi ha molts llenguatges multiparadigma.

Paradigmes de la programació

Imperatius

- noció d'estat
- canvi d'estat (efectes laterals).

Ja se n'han vist uns quants:
C, C++, Ensamblador, ...

Útils quan, per exemple, l'eficiència és clau.

Paradigmes de la programació

Funcionals

Són llenguatges procedurals però sense noció d'estat.

No hi ha efectes laterals.

- Més fàcil de raonar sobre corectesa.
- Útils per al prototipat, fases inicials de desenvolupament (especificacions executables i transformables).
- Tractament simbòlic.
- Sistemes de tipus potents.

Haskell, ML (Caml, OCaml), XSLT (tractament XML), Erlang,...

Paradigmes de la programació

Lògics

Llenguatges descriptius.

El programa diu que s'ha de fer, però no necessariament com.

- Prototipat d'aplicacions amb forta component simbòlica, problems combinatoris, etc.
- *Queries* en bases de dades relacionals o lògiques.
- Per especificació i raonament automàtic.

SQL (relacional), Prolog (lògica de primer ordre),...

(vistos en altres cursos)

Paradigmes de la programació

Orientats a objectes

- Es basa en *objectes* (camps + mètodes)
- Inclou principalment *polimorfisme* i *herència*
- Poden tenir sistemes de tipus complexos.

(vistos en cursos anteriors C++ i Java)

Paradigmes de la programació

Multiparadigma

Combinen diferents paradigmes:

- Python, Perl: Orientant a objectes + imperatiu + funcional
- Ocaml: funcional + Orientant a objectes + imperatiu

Alguns incorporen caràcteristique d'altres paradigmes

- Prolog: logic (+ imperatiu + funcional)

Altres combinacions:

- Erlang: funcional, concurrent, distribuït

Compilat vs Interpretat

- Compilats:
  el codi és transforma en codi objecte i després es monta en un executable.
  Exemples: C, C++, Ada, Haskell, ...

- Interpretat:
  el codi es transforma en codi d'una màquina virtual, que l'executa.
  Exemples: Python, JavaScript, Prolog (WAM),..

- Just-in-time compilation (JIT):
  entre els dos; es compila a un codi intermitg (bytecode) que és interpretat.
  Exemples: Java (JVM),..

Compilat vs Interpretat

Interpretar té aventatges i inconvenients:

++ Augmenten la portabilitat i l'expressabilitat (es poden fer més coses en temps d'execució).

- - Disminueix l'eficiència.

Amb el Just-in-time compilation s'intenta obtenir el millor dels dos mons.

Alguns interpretats, poden ser també compilats (per exemple, Basic, Lisp, Prolog, Haskell).

Sistemes de tipus

- Fortament tipats.
  Haskell, Java, Python, C++ (amb algunes conversions)
- Dèbilment tipats.
  Basic, JavaScript, Perl
  En Basic 2+″2″ és 4

- Comprovació estàtica de tipus (en compilació).
  Haskell, C++,. . .
- Comprovació dinàmica de tipus (en execució).
  Python, Ruby,. . .