Linguaggi di Programmazione con Laboratorio

-CdS Matematica

Marco Bellia, Dip. Informatica, Università di Pisa

February 23, 2015

Overview

- Corso 2015
  - Programma
  - Obiettivi del corso
  - Organizzazione
- Linguaggi di Programmazione:
  Cosa sono? Perché più di uno? Perché studiarli?
- Due classi di base: Prescriptive vs. Descriptive
- Paradigmi di Linguaggi di Programmazione
- 50 years of Programming Languages: 1955-2005

Programma 2015 /1

- **Corso Principale:** Principi di Linguaggi di Programmazione
  - Linguaggi di Programmazione e Macchine Astratte
  - Metodi di definizione di LP
  - Struttura dei Linguaggi e influenza sulle Metodologie di Programmazione
  - Meccanismi di supporto, principali realizzazioni e loro influenza nell'uso dei costrutti
  - Studio di paradigmi e delle relazioni tra programmi sviluppati in essi
    - Linguaggi Imperativi: Struttura, Applicazioni, C vs. Pascal
    - Linguaggi Funzionali: Struttura, Applicazioni, OCaml
    - Linguaggi Object Oriented: Struttura, Applicazioni, Java
- **Laboratorio:** Esperienze di implementazione di LP

Programma 2015 /2

- **Corso Principale:** Principi di Linguaggi di Programmazione
- **Laboratorio:** Esperienze di implementazione di LP
  - Strumenti: formalismi per sintassi, e per semantica, OCaml
  - Attività:
    - Definizione di sotto/frammenti di linguaggi dei paradigmi trattati nel corso
    - Realizzazione di esecutori di programmi per tali linguaggi

- **Prerequisiti** Fondamenti di Programmazione e Laboratorio
- **Materiale**

• Gabrielli, M. and Martini, S. Programming Languages: Principles and Paradigms. Springer, 2010, 440p., ISBN 978-1-84882-913-8.
• Materiale distribuito e/o indicato durante il corso

Altre Letture

Mitchell, J.C. Concepts in Programming Languages. Cambridge Univ. Press, 2007, 529p., ISBN 978-0-52178-098-8.

Scott, M.L., Programming Language Pragmatics, Elsevier-Morgan Kaufmann Pub., 2009, 944p., ISBN 978-0-12374-514-9

Obiettivi 2015

Alla fine del corso:

- Maggiore consapevolezza dei limiti e delle possibilità del Linguaggio di Programmazione che si conosce, si sta usando, si deve usare
- Migliore conoscenza dell'uso di un Linguaggio di programmazione e del costo dei costrutti impiegati nel programmi scritti in tale linguaggio
- Esperienze di programmazione di una stessa applicazione in differenti paradigmi di programmazione
- Esperienza di programmazione nella realizzazione di esecutori di Linguaggi o sue parti.

Organizzazione 2015

- Docenti: Marco Bellia, Vincenzo Ciancia
- Orario Corso: Lunedì 11-13; Martedì 09-11; Mercoledì 14-16 – Aula P1
- Orario Laboratorio: Venerdì 11-13 – Aula M
- Contatti: bellia@di.unipi.it
- Pagina del corso:
- Orario di ricevimento:
- Modalità di esame. Media ottenuta su due prove
  Prova Preliminare:
    Esercizi sugli argomenti trattati (corso e laboratorio)
  Prova Finale:
    Test a risposta multipla sugli argomenti trattati

Linguaggi di Programmazione: Cosa sono?

Rispondiamo in 5 slides

Linguaggi di Programmazione: Cosa sono? /1

Gli **strumenti fondamentali** per esprimere *Computer Applications*

- Computer Applications sono:
  - Tutte le applicazioni che:
    - *sono state*, *sono* e *saranno* eseguibili su computers *isolati* e/o *interconnessi*
    - *pervadono ogni comparto e attività* della nostra esistenza: Information, Production, Education, Research, Culture, Health..
  - Tutte le **Funzioni Calcolabili**, $\mathcal{F}$

Linguaggi di Programmazione: Cosa sono? /2

Gli **strumenti fondamentali** per esprimere *Computer Applications*

- Computer Applications sono:
  - Tutte le applicazioni che ....
  - Tutte le **Funzioni Calcolabili** $\mathcal{F}$. Sia $\mathcal{D}$ insieme numerabile,
    - $\mathcal{F} \equiv \mathcal{D} \rightarrow \mathcal{D}$
    - $\mathcal{D} \cong^1 \mathcal{D} \rightarrow \mathcal{D}$
    - esprimibili [2] mediante *specifici formalismi* $\mathcal{L}$ che includono: Combinatory Logic, Lambda Calculus, Turing Machines, **Linguaggi di Programmazione, LP**

[1] $\cong$ sta per *isomorfo*

[2] esprimibili qui, significa *finitamente, completamente definite.*

Linguaggi di Programmazione: Cosa sono? /3

Gli **strumenti fondamentali** per esprimere *Computer Applications*

- Computer Applications sono:
  - Tutte le applicazioni che ....
  - Tutte le **Funzioni Calcolabili**, $\mathcal{F}$
    - $\mathcal{F} = \mathcal{D} \rightarrow \mathcal{D}$
    - $\mathcal{D} \cong \mathcal{D} \rightarrow \mathcal{D}$
  - Tutti i programmi $\mathcal{P}$ di un **LP** $\mathcal{L}$ [3]
    - (Universale) $\exists \mathcal{U} : \mathcal{P} \rightarrow \mathcal{F} \in \mathcal{F}$,
    - $\forall \mathtt{g} \in \mathcal{F}, \exists \mathtt{p} \in \mathcal{P}$,
      $$\mathcal{U}(\mathtt{p})(\mathtt{x}) = \mathtt{g}(\mathtt{x}) \ (\forall \mathtt{x} \in \mathcal{D})$$

[3] $\mathcal{P}$ dipende da $\mathcal{L}$ e $\mathcal{P}$, e scriveremo estesamente $\mathcal{P}_{\mathcal{L}}$ quando più linguaggi $\mathcal{L}$ sono considerati e potrebbe esserci ambiguità. Analogamente, è la funzione universale $\mathcal{U}$ di $\mathcal{L}$.

Linguaggi di Programmazione: Cosa sono? /4

- Un Linguaggio di Programmazione è un
  - **Formalismo**

    Sintassi per la forma dei termini del linguaggio

    Semantica per il significato da associare a ciascun termine
  - per definire **Programmi**
    - sono la forma principale dei termini esprimibili del linguaggio
    - esprimono tutte e solo le funzioni calcolabili
    - realizzano un **algoritmo**
  - dotato di un **esecutore** dei suoi programmi (Macchina Astratta)

Linguaggi di Programmazione: Cosa sono? /5

- **Algoritmo** è un astrazione:
  - con cui possiamo studiare proprietà di *problemi*, quali:
    - La loro appartenenza ad $\mathcal{F}$
    - La definizione di *procedimenti effettivi* di calcolo, cioè *soluzioni*
    - Il costo delle diverse soluzioni per funzioni di $\mathcal{F}$
    - Bounds e Trattabilià di funzioni di $\mathcal{F}$, ad esempio:
      $g \in \mathcal{F}$ non può essere calcolato in meno di un costo C_g
      $g \in \mathcal{F}$ può essere calcolato in un costo inferiore a C_g
      $g \in \mathcal{F}$ è calcolabile in costo esponenziale con l'input
  - senza far riferimento ad alcun specifico Linguaggio di Programmazione (o Formalismo di Calcolo)

- Quando il problema di cui vogliamo realizzare una Computer Application ha proprietà note (prima tra tutte, essere in $\mathcal{F}$), allora dobbiamo usare un

  Linguaggio di Programmazione

  Ma non uno qualunque: Uno adatto all'applicazione.

Linguaggi di Programmazione: Perché più d'uno?

Rispondiamo in 1 slide

Linguaggi di Programmazione: Perché più d'uno?

- Linguaggi di Programmazione sono **equipotenti**
  - esprimono tutto e solo $\mathcal{F}$, ovvero
    hanno programmi per tutte e sole le funzioni $g \in \mathcal{F}$
- Linguaggi di Programmazione **non hanno la stessa espressività**
  - Si differenziano nel modo di esprimere la *Struttura dei Programmi*
  - Ne consegue che, taluni *procedimenti effettivi* con cui $g \in \mathcal{F}$ può essere calcolata sono impraticabili in un dato linguaggio.
    - Mancanza di costrutti adeguati:
      Esempio. Ricorsione per algoritmi induttivi (quali quicksort, list-append, binary-search...)
    - Necessità di emulare strutture presenti in altri linguaggi:
      Esempio. memoria dinamica per algoritmi su stack (non limitati)

Linguaggi di Programmazione: Perché studiarli? /1

Rispondiamo in 2 slides e.... un corso di 9 crediti

Linguaggi di Programmazione: Perché studiarli? /1

Per conoscere in cosa e quanto sono diversi ed essere consapevoli di vantaggi e limiti del linguaggio che si sta usando.

- **Espressivitá**. Perchè esprimono funzioni calcolabili in modo anche, molto diverso
- **sviluppo** Permettono di realizzare Computer Applications con caratteristiche diverse per
  - risorse necessarie per la realizzazione
  - prestazioni del sistema realizzato (facilità di uso, tempi e risorse richieste dall'uso...)
  - manutenzione (correzione errori, interventi di modifica e ri-adeguamento...)
- **Metodologie** Supportano *Metodologie di Programmazione* diverse e/o usabili in modo maggiore o minore
- **Acquisire Competenza** (sintassi, semantica, implementazione, uso e metodologie supportare)

Linguaggi di Programmazione: Perché studiarli? /2

- **Acquisire Competenza** (sintassi, semantica, implementazione, uso e metodologie supportare)
  - Nella programmazione e nell'uso anche di linguaggi già noti ed usati. Attraverso:
    - conoscenza dell'effettiva implementazione delle strutture usate
    - confronto sul modo di utilizzare strutture e meccanismi simili in linguaggi diversi
    - esperienza nell'analisi di caratteristiche oscure o difficili da usare in un linguaggio
    - stesura in linguaggi diversi di programmi e/o problemi noti
    - ...
  - Nella scelta del Linguaggio più appropriato per ogni specifica Applicazione da sviluppare
  - Per *classificare* e apprendere gli ultimi Linguaggi di Programmazione definiti

Una prima Classificazione: Prescriptive vs. Descriptive

- **Prescriptive = Come il calcolo deve procedere**
  Imperative Languages: State + Mutable Value + Assignment + Sequence Control

- **Descriptive = Cosa il calcolo deve produrre**
  Declarative Languages: Immutable Value + Application + Composition

Una classificazione più fine

I principali:

- **Procedural**: Fortran, Cobol, Algol, Pascal, C, ADA, ...
- **Functional**: Lisp, Scheme, ML, Haskell, OCAML, ...
- **Algebraic**: Lucid, OBJ, OPAL, ActOne
- **Logic, Constraint-based**: Prolog, LogLisp, Datalog, Parlog (SQL, spreadsheets languages,)...
- **Object Oriented**: Simula67, SmallTalk, C++, OCAML, Java, C#, F#, ...
- **Scripting**: Perl, Python, PHP, Ruby, JavaScript...
- **Concurrent**: Lucid, OCCAM, C-Linda, PrologLinda, SPARK, Parlog, Java, C#, ...
- **Dataflow**: Lucid, C-Linda, PrologLinda,...
- **Multi-paradigms**: (i più recenti) F#, Scala, ...

50 years of Programming Languages

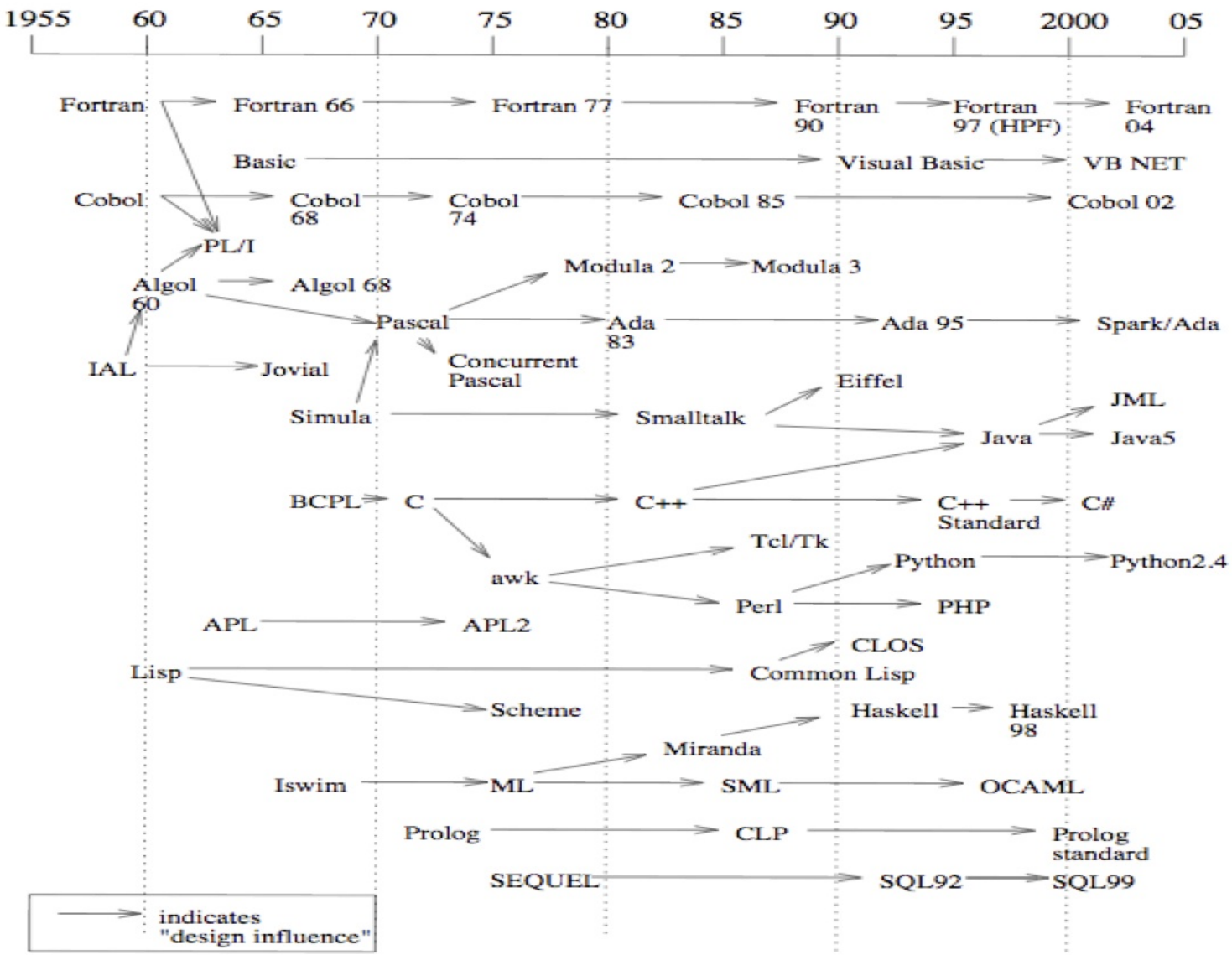

Figure 1.2: A Snapshot of Programming Language History