# Linguaggi formali e compilazione

## Corso di Laurea in Informatica

A.A. 2012/2013

# Linguaggi formali e compilazione

# Informazioni generali sul corso



- Sito web: http://algo.ing.unimo.it/people/mauro/dida/2012-13_LFC/
- Ricevimento: mercoledì 14-16 (ma prima si consulti sempre la pagina web all'indirizzo http://algo.ing.unimo.it/people/mauro/dida/orario_ricevimento.shtml)
- Materiale didattico: si veda il sito web
- Modalità d'esame: scritto (orale solo su richiesta)
- CFU assegnati: 6 (circa 150 ore di lavoro)

# Obiettivi formativi



Sviluppo di conoscenze e competenze pratiche su:

- linguaggi formali;
- linguaggi formali nell'uso quotidiano (strumenti automatici come *grep*, *sed*, *AWK*, *Lex* e *YACC*;
- linguaggi di programmazione e compilatori.

# Un esempio motivante

- Raccolta e organizzazione di informazioni disponibili sul Web
- Passaggio da informazione non strutturata (o semi-strutturata), come:
  ... lo studente Marco Rossi, residente a Modena, in via Bianchi 10, tel. 059231818, ....
  a informazione con uno schema ben definito:

| Nome | Comune di residenza | Indirizzo | Telefono |
|---|---|---|---|
| Marco Rossi | Modena | Via Bianchi, 10 | 059231818 |

- Un esempio di filtraggio di informazione utilizzando Sed.

# Le fondamentali nozioni di alfabeto, stringa e linguaggio



- Un *alfabeto* è semplicemente un insieme finito di simboli, detti anche *caratteri*.
- I simboli di un alfabeto verranno scriti usando il font `typewriter`.
- Esempi di alfabeti importanti:
  - I set di caratteri ASCII e UNICODE;
  - $\mathcal{B} = \{\mathtt{0}, \mathtt{1}\}$, l'alfabeto binario;
  - $\mathcal{D} = \{\mathtt{A}, \mathtt{C}, \mathtt{G}, \mathtt{T}\}$, l'alfabeto del DNA.
- La lettera greca $\Sigma$ indicherà un generico alfabeto.
- Le lettere finali dell'alfabeto latino ($x$, $y$ e $z$) indicheranno simboli generici di un alfabeto.

# Stringhe

- Una stringa, su un dato alfabeto $\Sigma$, è una sequenza di simboli (giustapposti) di $\Sigma$.
- `010110` e `1111` sono esempi di stringhe sull'alfabeto binario.
- `ATGGTGCACCTGACT` è una stringa sull'alfabeto $\mathcal{D}$. (È l'inizio della sequenza che codifica la beta emoglobina, un componente del sangue.)
- `Linguaggi formali e compilazione` è una stringa sull'alfabeto ASCII.
- Conviene considerare anche la stringa vuota, cioè una stringa composta da nessun carattere.
- La stringa vuota è definita su qualsiasi alfabeto.

# Stringhe



- Per indicare stringhe generiche si utilizzeranno (a seconda dei casi) sia le ultime lettere dell'alfabeto latino ($X, Y, Z$), in maiuscolo, sia le prime lettere dell'alfabeto greco ($\alpha$, $\beta$, $\gamma$), in minuscolo.
- La lettera $\epsilon$ verrà riservata per denotare la stringa vuota.
- Una stringa formata da un solo carattere è un oggetto diverso dal carattere stesso.
- In tutti i casi nei quali possano sorgere ambiguità, ad esempio nel caso di stringhe composte da un solo carattere, le stringhe verranno rinchiuse fra doppi apici.
- ‘‘0’’ è quindi un oggetto diverso da 0.

# Stringhe (continua)



- Il numero di caratteri che compongono una stringa $X$ è detto lunghezza di $X$.
- La lunghezza di una stringa $X$ si indica spesso con $|X|$.
- Se $X = \text{"}\texttt{Questa e' una stringa}\text{"}$, allora $|X| = 21$.
- Si tratta di una funzione comunemente disponibile anche nei linguaggi di programmazione:
  - in C: $\texttt{strlen}(X)$
  - in Python: $\texttt{len}(X)$
  - in C++: $X.\texttt{length}()$

# Stringhe (continua)



- Se $|X| = n$, i singoli caratteri della stringa verranno individuati da $X_i$, $i = 1, \ldots, n$.
- Nel linguaggi di programmazione, si usa spesso la stessa notazione utilizzata per gli array: $X[i]$.
- Date due stringhe $X$ e $Y$, la stringa $Z$ ottenuta giustapponendo i caratteri di $Y$ a quelli di $X$ è detta *concatenazione* di $X$ e $Y$.
- Ad esempio se $X = \texttt{Linguaggi}$ e $Y = \texttt{formali}$, risulta $XY = \texttt{Linguaggiformali}$.
  - in C: $\texttt{strcat}(X, Y)$ ($X = \texttt{Linguaggiformali}$, $Y$ invariata)
  - in Python: $X + Y$ ($X$ e $Y$ invariate)
  - in C++: $X.\texttt{append}(Y)$ ($X = \texttt{Linguaggiformali}$, $Y$ invariata)
- Ovviamente, se $|X| = n$ e $|Y| = m$, risulta $|XY| = n + m$.

# Stringhe (continua)

- La concatenzione è associativa, ovvero

$$(XY)Z = X(YZ).$$

- La concatenazione *non* è commutativa: posto $X = \texttt{mela}$ e $Y = \texttt{verde}$, risulta $XY = \texttt{melaverde} \neq \texttt{verdemela} = YX$.
- La *potenza* $k$-esima di una stringa $X$, indicata con $X^k$, è la stringa ottenuta concatenando $X$ con se stessa $k \geq 0$ volte: $X^k = \overbrace{XX\ldots X}^{k}$.
- Si pone poi $X^0 = \epsilon$.
- La *riflessione* di una stringa $X = X_1X_2\ldots X_n$ è la stringa $X^R = X_nX_{n-1}\ldots X_0$ ottenuta rovesciando i caratteri di $X$.

# Stringhe (continua)



- $X$ è *sottostringa* di $Y$ se esistono $Z$ e $W$ (eventualmente vuote) tali che: $Y = ZXW$.
- Se almeno una fra $Z$ e $W$ è diversa dalla stringa vuota, allora $X$ è *sottostringa propria* di $Y$.
- Se $Z$ è vuota, $X$ è un *prefisso* di $Y$.
- Se $W$ è vuota, $X$ è un *suffisso* di $Y$.
- La notazione $X_{i..j}$ verrà utilizzata per indicare la sottostringa di $X$ formata dai caratteri dall'$i$-esimo al $j$-esimo.

# Esempi

- Se $X = \texttt{01}$, risulta $X^3 = \texttt{010101}$, se $X = \texttt{Linguaggi}$, allora $X^2 = \texttt{LinguaggiLinguaggi}$
- Se $X = \texttt{Linguaggi formali e compilatori}$, allora
  - $Y = \texttt{formali}$ è una sottostringa propria di $X$,
  - $Z = \texttt{Linguaggi}$ è un prefisso di $X$,
  - $W = \texttt{compilatori}$ è un suffisso di $X$,
  - $X_{10..17} =$ ‘‘ formali’’.
  - Si noti, nell'ultimo esempio, l'uso degli apici, necessario per evitare le possibili ambiguità dovute alla presenza di un spazio ad inizio stringa.
- Se $X = \texttt{Roma}$, allora $X^R = \texttt{amoR}$.

# Stringhe: ricapitolazione



- Le stringhe sono sequenze di caratteri (simboli) giustapposti, appartenenti ad un dato insieme, detto alfabeto.
- Sulle stringhe sono definite numerose operazioni.
- Tutti i linguaggi di programmazione prevedono meccanismi linguistici per definire e per manipolare stringhe.
- Il numero di applicazioni che richiedono manipolazione di stringhe è enorme e in continuo aumento (text processing, information retrieval, web mining, dna manipulation, ....)

# Linguaggi

- Un *linguaggio* su un dato alfabeto $\Sigma$ è un insieme di stringhe su $\Sigma$.
- Un linguaggio $L$ può essere specificato in molti modi (ne vedremo almeno 4).
  - Per enumerazione, cioè elencando le stringhe di $L$.
  - Descrivendo (matematicamente o anche in linguaggio naturale) come sono fatte le stringhe.
  - Mediante regole per generare le stringhe di $L$ (*formalismo generativo*).
  - Mediante algoritmi per riconoscere le stringhe di $L$ (es. *automi riconoscitori*).
- Buona parte di questo corso sarà dedicato alle grammatiche e agli automi riconoscitori, che rappresentano gli ultimi due modi di specifica.

# Esempi

- Esempi di linguaggi su $\mathcal{B}$ definiti per enumerazione:
  - $L_1 = \{00, 01, 10, 11\}$;
  - $L_2 = \{\epsilon, 0, 1, 00, 01, 10, 11, 000, 001, 010, 011, 100, 101, 110, 111\}$;
  - $L_3 = \{00, 010, 0110, 01110, \ldots\}$;
  - $L_4 = \{0, 1, 00, 11, 000, 010, 101, 111, \ldots\}$.
- Gli stessi linguaggi possono essere specificati in modo più sintetico:
  - $L_1 = \{X \in \mathcal{B}^* : |X| = 2\}$;
  - $L_2 = \{X \in \mathcal{B}^* : |X| \leq 3\}$;
  - $L_3 = \{X \in \mathcal{B}^* : \exists k \geq 0 \,\text{t.c.}\, X = 01^k0\}$;
  - $L_4 = \{X \in \mathcal{B}^* : X = X^R\}$.
- Proviamo a descrivere $L_1, \ldots, L_4$ usando la lingua italiana....

# Esempi (continua)

- $L_1$ ed $L_2$ sono esempi di linguaggi *finiti*, mentre $L_3$ ed $L_4$ sono linguaggi *infiniti*.
- Naturalmente i linguaggi infiniti rivestono maggior interesse, sia dal punto di vista teorico sia da quello applicativo.
- Dato un qualunque alfabeto $\Sigma$, due linguaggi su $\Sigma$ che rivestono particolare importanza sono l'insieme (o linguaggio) vuoto $\Phi$ e l'insieme di tutte le possibili stringhe su $\Sigma$ (denotato da $\Sigma^*$).

# Esempi (continua)

- Altri esempi di linguaggio, questa volta definiti sull'alfabeto di tre simboli $\Sigma = \{a, b, c\}$:
  - $L_5 = \{X \in \Sigma^* : X = a^n b^m, n, m \geq 0\}$; N.B. $n$ ed $m$ possono essere diversi
  - $L_6 = \{X \in \Sigma^* : X = a^n b^n, n \geq 0\}$; N.B. $n$ ed $m$ devono essere uguali
  - $L_7 = \{X \in \Sigma^* : X = b^n c^n, n \geq 0\}$;
  - $L_8 = \{X \in \Sigma^* : X = a^n b^n c^m, n, m \geq 0\}$;
  - $L_9 = \{X \in \Sigma^* : X = a^n b^m c^m, n, m \geq 0\}$.

# Operazioni con i linguaggi

- Innanzitutto, poiché i linguaggi sono insiemi (sono insiemi di stringhe), su di essi sono definite tutte le più comuni operazioni insiemistiche: unione, intersezione, differenza, ecc.
- Due linguaggi $L'$ ed $L''$ si possono poi *concatenare*:

$$L = L'L'' = \{X \in \Sigma^* : \exists Y \in L', \exists Z \in L'' \,\mathrm{t.c.}\, X = YZ\}$$

  In altre parole, $L$ è costituito da stringhe che sono la concatenazione, in tutti i modi possibili, di una stringa di $L'$ e di una stringa di $L''$.
- Più in generale, la seguente ricorrenza permette di definire, per ogni $m \geq 0$, la *potenza n-esima* di un linguaggio $L$:

$$\begin{aligned} L^0 &= \{\epsilon\} \\ L^n &= L^{n-1}L,\ n > 0. \end{aligned}$$

# Operazioni con i linguaggi (continua)

- La *chiusura* (*riflessiva*) di $L$ è il linguaggio

$$L^* = \bigcup_{n=0}^{\infty} L^n = L^0 \cup L^1 \cup L^2 \cup \dots$$

- Ad esempio:

$$\begin{aligned} \mathcal{B}^* &= \bigcup_{n=0}^{\infty} \{0,1\}^n \\ &= \{0,1\}^0 \cup \{0,1\}^1 \cup \{0,1\}^2 \cup \dots \\ &= \{\epsilon\} \cup \{0,1\} \cup \{00,01,10,11\} \dots \end{aligned}$$

e dunque $\mathcal{B}^*$ è l'insieme di tutte le possibili stringhe binarie.

- In generale, $\Sigma^*$ é l'insieme di tutte le possibili stringhe su un alfabeto $\Sigma$.

# Operazioni con i linguaggi (continua)



- La *chiusura* (non riflessiva) di $L$ è definita come $L^+ = LL^*$, ovvero:

$$L^+ = \bigcup_{n=1}^{\infty} L^n = L^1 \cup L^2 \cup \ldots$$

- La *riflessione* di un linguaggio $L$ su un alfabeto $\Sigma$ è il linguaggio:

$$L^R = \{X \in \Sigma^* : \exists Y \in L \,\text{t.c.}\, X = Y^R\}.$$

- Il numero di stringhe di un linguaggio finito verrà indicato con la notazione $|L|$. Se $L$ è infinito risulta $|L| = \mathbf{N}$, con ciò intendendo che $L$ ha la stessa cardinalità dell'insieme dei numeri naturali.

# Esempi

Ricordiamo che:

- $L_5 = \{X \in \Sigma^* : X = a^n b^m, n, m \geq 0\}$
- $L_6 = \{X \in \Sigma^* : X = a^n b^n, n \geq 0\}$
- $L_7 = \{X \in \Sigma^* : X = b^n c^n, n \geq 0\}$
- $L_5 \backslash L_6 = \{X \in \Sigma^* : X = a^n b^m, n, m \geq 0, n \neq m\}$
- $L_5 \cup L_6 = L_5$
- $L_5 \cap L_6 = L_6$
- $L^6\{c\}^* = L_8$
- $L_6 L_7 = \{X \in \Sigma^* : X = a^n b^m c^k, n, k \geq 0, m = n + k\}$
- $L\{\epsilon\} = \{\epsilon\}L = L$
- $|L_1| = 4$, $|L_2| = 15$, $|L_5| = |L_6| = |L_7| = \mathbf{N}$

# Esempi (continua)



- $\{ab, c\}^2 = \{abab, abc, cab, cc\}$
- $\{ab, c\}^3 = \{ababab, abcab, cabab, ccab, ababc, abcc, cabc, ccc\}$
- $\{ab, c\}^* = \{\epsilon, ab, c, abab, abc, cab, cc, ababab, abcab, cabab, ccab, ababc, abcc, cabc, ccc, \ldots\}$
- $\{ab, c\}^+ = \{ab, c\}^* \setminus \{\epsilon\}$
- $\left(\{ab, c\}^2\right)^R = \{baba, cba, bac, cc\}$