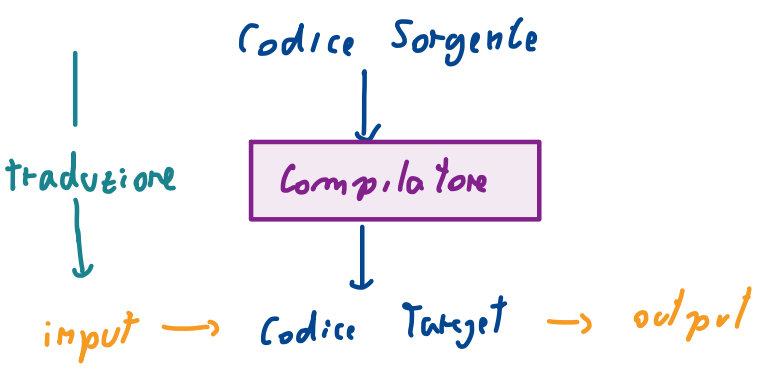
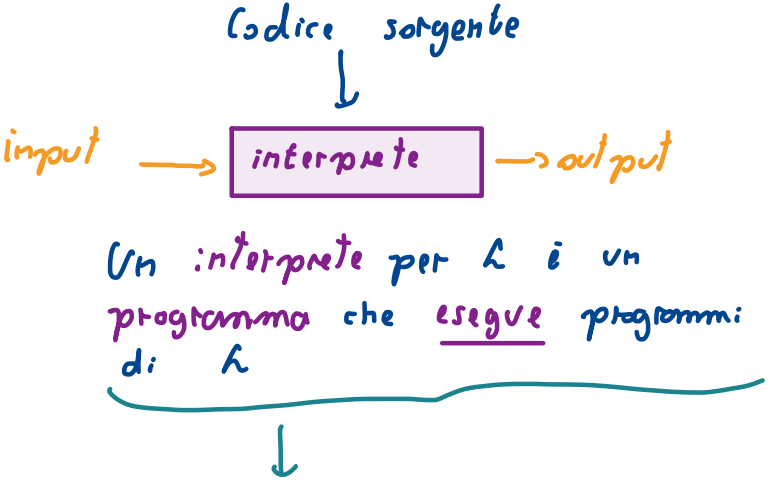
Interpreti e Compilatori

Obbiettivo: Capire le basi dell'implementazione di: OCaml





Un compilatore per $\mathcal{L}$ è un programma che traduce programmi di $\mathcal{L}$ } → codice come dato

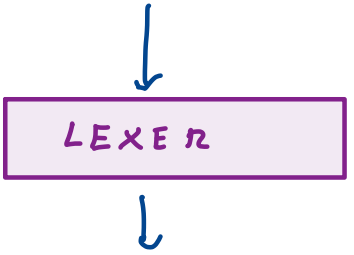
INTERPRETI
- "facili" da implementare ☺
- utili per capire come funziona un linguaggio ☺
  (→ definitional interpreter)
- poco efficienti ☹
- analisi estensionale (→ type checking) ☺/☹

COMPILATORI
- "difficili" (almeno un intero corso dedicato ai compilatori) ☹
- semanticamente oscuri ☹
- efficienti ☺
- analisi e ottimizzazione (intensionale) ☺

Fasi di Compliazione e Interpretazione

① Analisi lessicale

`if x = 0 then 1 else fact (x - 1)` ⇝ stringa di: caratteri

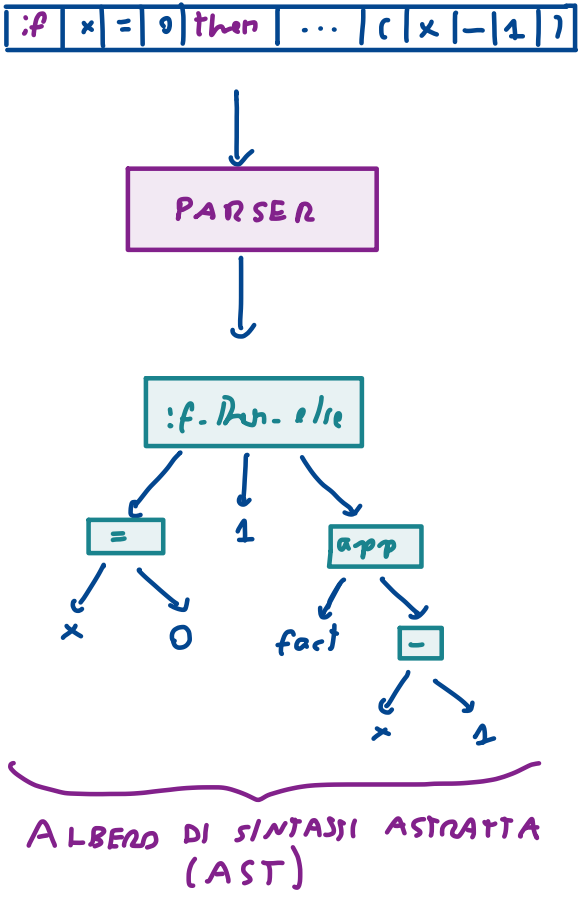


⇝ stream / lista di: token

Token = minima unità simbolica semanticamente sensata del linguaggio

⇝ parole

② Analisi Sintattica



AST catturano la struttura sintattica di una espressione

- dicono quali sono gli operatori
- quali e quanti: i loro argomenti

LA struttura è gerarchica e strutturata ~~~ no parentesi

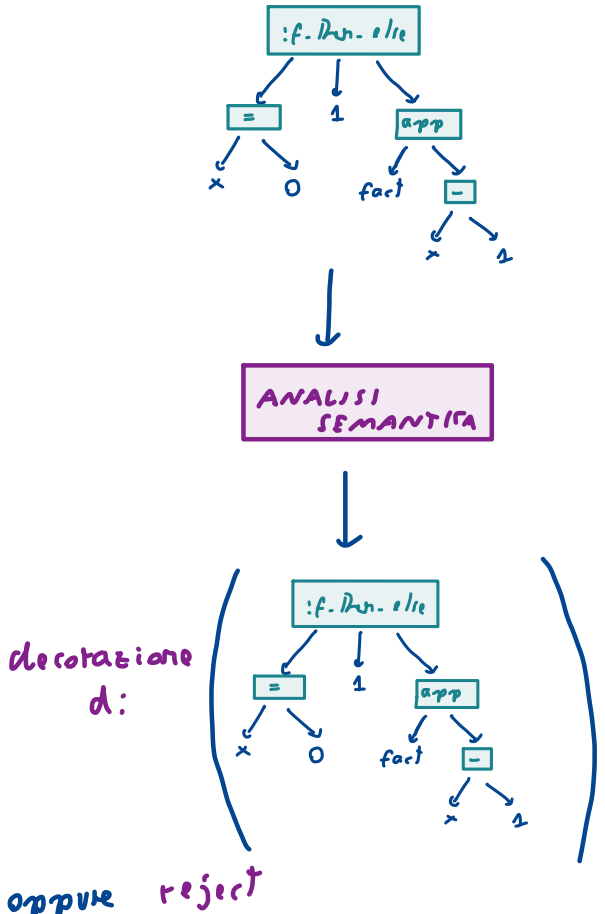
NB Sintassi è un tipo di dato

⇒ ricorsione

⇒ pattern matching

⋮

③ Analisi Semantica



Analisi semantica determina se il programma, visto come oggetto statico, è semanticamente sensato

- Creazione tabelle simboliche
  ident ↦ tipo
- Decorazione albero con tipi
- type inference e type checking
- Esaustività pattern matching
- ⋮

④ Interprete ⟶ esegue AST

Compilatore ⟶ traduce AST ... ⟶ codice macchina

Linguaggi funzionali ottimi per interpretazione e compilazione

- Base simbolica
- AST come tipo algebrico
- esecuzione / traduzione ricorsiva tramite pattern matching

WARM-UP: Un interprete per aritmetica
(~> calcolatrice)

① Linguaggio

$e ::= c_m \mid e + e \mid e * e$ } SINTASSI ASTRATTA

BNF

$\Sigma = \{ c_m, +, *, (,), - \}$

alfabeto

$\Upsilon = \{ c_m, +, *, \ldots, END \}$

tokens

} SINTASSI CONCRETA

In ocaml

string list $\xrightarrow{\text{parse}}$ exp

string list: Σ^*

exp: tipo delle espressioni: AST

Una espressione è

- un intero
- oppure un operatore $(+, *)$ applicato a due espressioni