07. Traduttori

Corso di Informatica

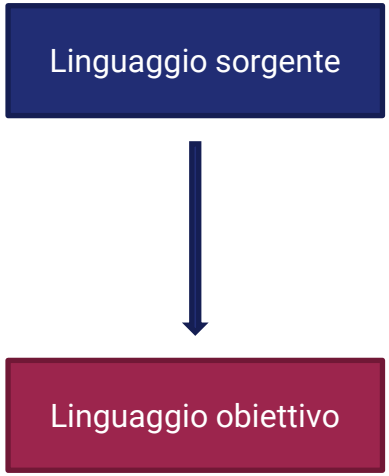
Corso di Laurea in Matematica (D.M. 270/04) - A.A. 2020/2021
Angelo Cardellicchio
angelo.cardellicchio@uniba.it
28/10/2020

Outline

- Traduttori
- Compilatori
  - Tipologie di compilatore
  - Operazioni di un compilatore
- Interpreti
- Compilatore vs. Interprete

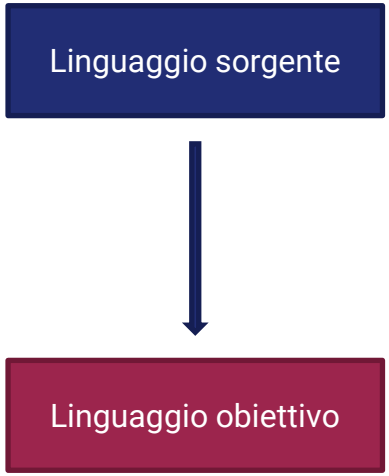
Traduttori

- Traducono il codice scritto in un *linguaggio sorgente* in codice scritto in un *linguaggio obiettivo*
- Valuta la *correttezza* di ciascuna istruzione
- Due categorie principali: *compilatori* ed *interpreti*



Compilatori – Tipologie di compilatore

- Compilatore ‘standard’: da linguaggio ad alto/medio livello a codice oggetto
- Ne esistono di altri tipi
  - ***Cross-compilatore***
  - ***Decompilatore***
  - ***Transcompilatore***
  - ***Etc.***

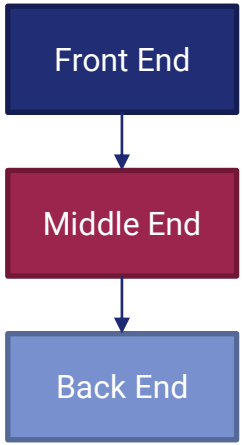


Compilatori – Tipologie di compilatore

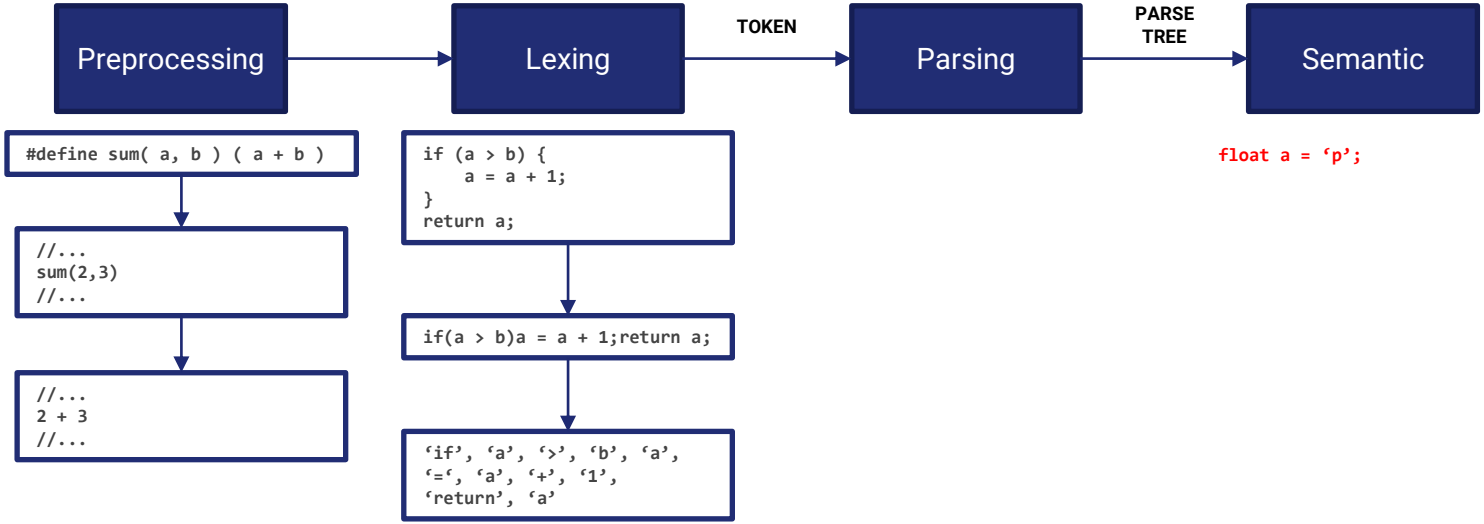
Tipologia	Linguaggio sorgente	Linguaggio obiettivo	Eseguito su
Standard	Alto/medio livello (es. C++)	Basso livello (es. assembly)	Stessa architettura
Cross-compilatore	Alto/medio livello (es. C++)	Basso livello (es. assembly)	Diverse architetture
Decompilatore	Basso livello (es. assembly)	Alto/medio livello (es. C++)	Stessa architettura
Trans-compilatore	Alto/medio livello	Alto/medio livello	Stessa architettura

Operazioni di un compilatore

- *Three-step compiler*
- *Front end*
  - analisi del sorgente
  - creazione di una rappresentazione intermedia (IR)
- *Middle end*
  - ottimizzazione IR (generica)
- *Back end*
  - ottimizzazione (specifica per architettura)
  - generazione del codice oggetto



Operazioni di un compilatore – Front End



Operazioni di un compilatore – Middle End

- *Middle end*
  - ottimizzazione IR (generica)
  - Esempio: eliminazione di ridondanze

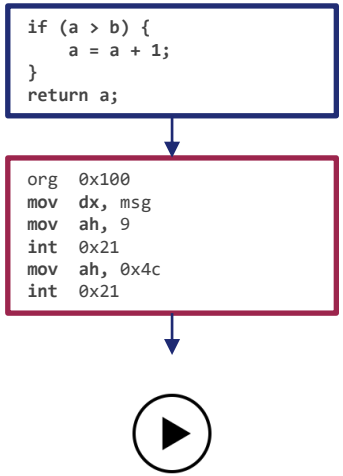
```
int main (void) {
    int a = 0;
    // Codice ridondante!
    if (1 == 0) {
        a = 1;
    }
    return a;
}
```

Operazioni di un compilatore – Back End

- *Back end*
  - Ottimizzazione (specifica per architettura)
  - Generazione del codice oggetto
- Una ottimizzazione specifica causa una ***minore portabilità del codice oggetto!***
- Il codice oggetto ***non è l'eseguibile!***
- Esiste uno step successivo delegato al ***linker***.

Interpreti

- Controllano la singola istruzione del sorgente
- La traducono in linguaggio macchina
- La eseguono



Compilatore vs. Interprete

	Compilatore	Interprete
Vantaggi	• Ottimizzazione del codice • Maggiore velocità di esecuzione • Controllo sintattico e semantico a compile time	• Compile time assente • Maggiore portabilità tra architettura • Possibilità di debug con granularità a livello di singola istruzione
Svantaggi	• Minore portabilità tra architetture • Debug complesso da effettuare	• Minore velocità di esecuzione • Performance non ottimizzate • Possibili errori imprevisti a run time

Domande?

42