Linguaggi di programmazione

Esistono molti diversi linguaggi di programmazione. Sono stati pensati in base a diverse metodologie di lavoro e/ in riferimento a specifici campi di applicazione.

Nel nostro contesto ci occupiamo di linguaggi di uso generale (general purpose). Fra i linguaggi cito per esempio Java, C++, C#, Pascal, PHP, Fortran, Cobol, Basic, Ruby, Pyton... l'elenco può essere lunghissimo.

In questo corso faremo riferimento al linguaggo Java, che fa esplicito riferimento alla programmazione orientata ad oggetti.

Il codice sorgente

Il primo passo per realizzare un programma è di scrivere il "**sorgente**" (in inglese source). E' in sostanza il testo del programma, scritto seguendo le regole proprie del linguaggio utilizzando i normali caratteri alfabetici, numerici, di punteggiatura e quindi in qualche modo leggibili anche da noi.

Il programma così preparato non è ancora nella forma in cui il computer è in grado di eseguire, occorrono alcuni passaggi. Il tipo di passaggi può essere diverso tra un linguaggio e l'altro secondo tre grandi categorie:

- linguaggi compilati
- linguaggi interpretati
- linguaggi a codice intermedio

Linguaggi compilati

Caratteristica propria dei linguaggi di questo tipo è che il sorgente viene fatto elaborare da un particolare programma chiamato **compilatore** che svolge due funzioni: verifica la correttezza sintattica del programma scritto e se non incontra errori ne effettua la traduzione dal linguaggio del sorgente al linguaggio proprio del calcolatore (il linguaggio macchina). Ciò che si tiene viene chiamato **programma in versione oggetto**, ma in questo contesto il termine oggetto non assume lo stesso significato a cui abbiamo fatto riferimento parlando di oggetti e classi. Se il sorgente contiene uno o più errori sintattici non viene prodotto il file con la versione oggetto, ma viene fornito l'elenco degli errori riscontrati.

Il programma in versione oggetto è scritto in linguaggio macchina, ma non è ancora immediatamente eseguibile perché mancano alcune ulteriori informazioni e parti comuni che dovranno essere aggiunte. Per questo motivo la fase successiva è di far elaborare il programma in versione oggetto da un altro programma chiamato **linker** (termine inglese che significa 'collegatore') che appunto ha la funzione di collegare i riferimenti mancanti nel programma oggetto al fine di ottenere la versione definitiva chiamata **programma eseguibile**. A volte lo si trova indicata anche come versione binaria (binary).

In questa versione il programma può essere eseguito dal computer producendo gli effetti previsti dalle istruzioni che lo compongono.

Il programma eseguibile ottenuto è applicabile solamente ad un unico tipo di computer e tipo di sistema operativo. Quindi un programma compilato per un pc con sistema operativo windows potrà essere utilizzato dai compiter con queste caratteristiche e quindi non da

uno che abbia ad esempio il sistema operativo linux. Per questo ultimo occorre ri-effettuare i passaggi partendo dal sorgente e utilizzando compilatore e linker specifici per linux.

Appartengono a questa categoria ad esempio i linguaggi C++, Fortran, Cobol e molti altri

Linguaggi interpretati

Per linguaggi di questo tipo non esiste una fase di traduzione del sorgente. Si fa uso di un programma chiamato **interprete** a cui viene affidata l'elaborazione del sorgente. L'interprete esamina la prima istruzione, ne verifica la correttezza sintattica e in caso positivo esegue direttamente l'istruzione stessa, poi passa alla successiva e così via.

Se una istruzione contiene un errore l'interprete interrompe l'esecuzione e segnala a che punto si ferma e il motivo. Quindi un programma contenente uno o più errori potrà essere parzialmente eseguito.

Su computer e/o sistemi opertivi diversi lo stesso sorgente poà essere interpretato a patto di disporre del programma interprete adeguato all'ambiente specifico.

Fanno parte di questa categoria ad esempio i linguaggi Basic, PHP, Perl, Pyton e molti altri

Linguaggi a codice intermedio

Per questa categoria di linguaggi il programma sorgente deve subire una prima fase di verifica sintattica e traduzione producendo una versione detta **codice intermedio**. La differenza rispetto alla compilazione è che ciò che si è ottenuto non è ancora espresso secondo il linguaggio macchina, per essendo una forma non più comprensibile a noi. Il codice intermedio ottenuto deve essere affidato all'elaborazione di una '**macchina virtuale**' che è in grado di interpretare ed eseguire il suo contenuto. La macchina virtuale è in sostanza un programma specifico per tipo di computer e sistema operativo.

In sostanza ci troviamo in una situazione 'mista' delle due precedenti. La prima fase di traduzione è simile a quella del compilatore, ma produce un codice intermedio che è indipendente dal particolare tipo di computer/sistema operativo. Quindi un programma in versione codice intermedio è in sostanza eseguibile su piattaforme diverse. D'altra parte la macchina virtuale è in qualche modo simile ad un interprete, con la differenza che ha maggiore efficienza perché non deve occuparsi di verifica sintattica (già fatta nella traduzione verso il codice intermedio) e la traduzione finale stessa è più semplice visto che si parte da un codice molto più vicino alla macchina del linguaggio di programmazione.

Fanno parte di questa categoria ad esempio i linguaggi Java, C#, Pascal e altri

cc