Linguaggi di Programmazione

Capitolo 6 del testo

Alberto Policriti

19 dicembre, 2019

La storia

Negli anni ’40 si scrivevano programmi usando il codice macchina:
“Muovi il contenuto del registro 5 al registro 6”

```
4056
```

oppure, usando *mnemonici*,

```
MOV R5 R6
```

La storia

Meglio, usando mnemonici e *identificatori*:

```
156C                      LD R5,Price
166D                      LD R6,ShippingCharge
5056                      ADDI R0,R5 R6
306E                      ST R0,TotalCost
C000                      HLT
```

assembler
programmi in grado di convertire linguaggio macchina in un linguaggio leggibile dall'umano: *assembly language*.

... ancora difficile da usare e, soprattutto, machine dependent.

La storia

prima generazione: linguaggio macchina

seconda generazione: assembly language

terza generazione: primitive di alto livello e istruzioni machine independent

Example
FORTRAN (FORmula TRANslator)

COBOL (COmmon Business-Oriented Language)

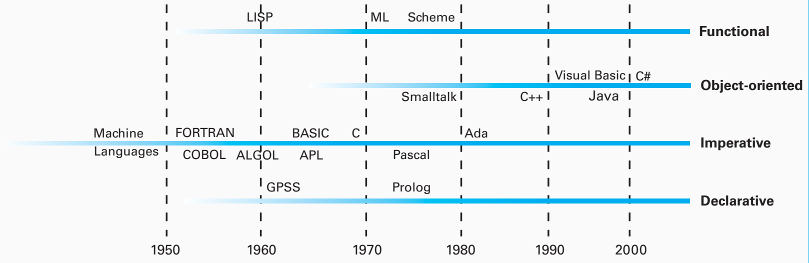
I traduttori di seconda generazione evolvono in

compilatori e interpreti

Una volta realizzato che si poteva programmare in modo indipendente ... *programming environments*: la macchina si adatta all'uomo.

La storia

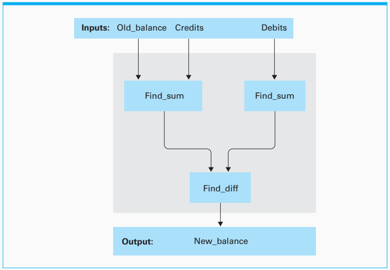
Programmng paradigms



La storia

Funzionale

```
(Find_diff (Find_sum Old_balance Credits) (Find_sum
Debits))
```



Imperativo

```
Total_credits ← sum of all Credits
Temp_balance ← Old_balance + Total_credits
Total_debits ← sum of all Debits
Balance ← Temp_balance − Total_debits
```

La storia

Logico/Dichiarativo
Prolog - Constraint programming

Object Oriented

- oggetti
- metodi
- classi
- istanze

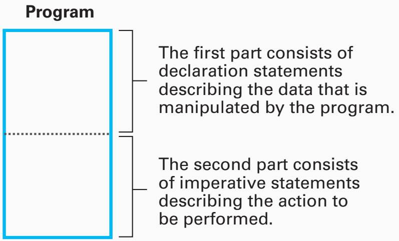
La storia

Scripting Languages

A subset of the imperative programming languages is the collection of languages known as **scripting languages.** These languages are typically used to perform administrative tasks rather than to develop complex programs. The expression of such a task is known as a **script,** which explains the term “scripting language.” For example, the administrator of a computer system might write a script to describe a sequence of record-keeping activities that should be performed every evening, or the user of a PC might write a script to direct the execution of a sequence of programs required to read pictures from a digital camera, index the pictures by date, and store copies of them in an archival storage system. The origin of scripting languages can be traced to the job control languages of the 1960s that were used to direct an operating system in the scheduling of batch processing jobs (see Section 3.1). Even today, many consider scripting languages to be languages for directing the execution of other programs, which is a rather restrictive view of current scripting languages. Examples of scripting languages include Perl and PHP, both of which are popular in controlling server-side Web applications (see Section 4.3), as well as VBScript, which is a dialect of Visual Basic that was developed by Microsoft and is used in Windows-specific situations.

Ada, C, C++, C#, FORTRAN, Java

Vecchio stile


Program
The first part consists of declaration statements describing the data that is manipulated by the program.
The second part consists of imperative statements describing the action to be performed.

- declarative statements
- imperative statements
- comments

Ada, C, C++, C#, FORTRAN, Java

Variabili

- (primitive) data type:

  ```
  float Length, Width;
  int Price, Tax, Total;
  char Symbol;
  ```

Strutture dati (elementari)

- array
- record
- field

```
struct {char Name[25];
        int Age;
        float SkillRating;}
       Employee;
```

Ada, C, C++, C#, FORTRAN, Java

Assegnamenti

$$Z = X + Y;$$
$$Z := X + Y;$$
$$Z \leftarrow X + Y;$$

Operatori e loro precedenze

Overloading

Ada, C, C++, C#, FORTRAN, Java

Istruzioni di controllo

```
    goto 40
20  Apply procedure Evade
    goto 70
40  if (KryptoniteLevel < LethalDose) then goto 60
    goto 20
60  Apply procedure RescueDamsel
70  ...
```

```
if (KryptoniteLevel < LethalDose)
    then (apply procedure RescueDamsel)
    else (apply procedure Evade)
```

programmazione strutturata

Ada, C, C++, C#, FORTRAN, Java

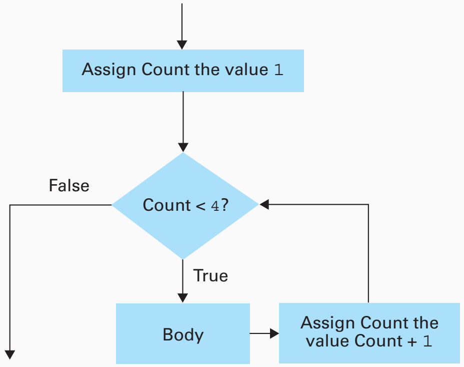
Istruzioni di controllo

```
switch (variable) {
  case 'A': statementA; break;
  case 'B': statementB; break;
  case 'C': statementC; break;
  default: statementD}
```

```
CASE variable IS
  WHEN 'A'=> statementA;
  WHEN 'B'=> statementB;
  WHEN 'C'=> statementC;
  WHEN OTHERS=> statementD;
END CASE
```

Ada, C, C++, C#, FORTRAN, Java

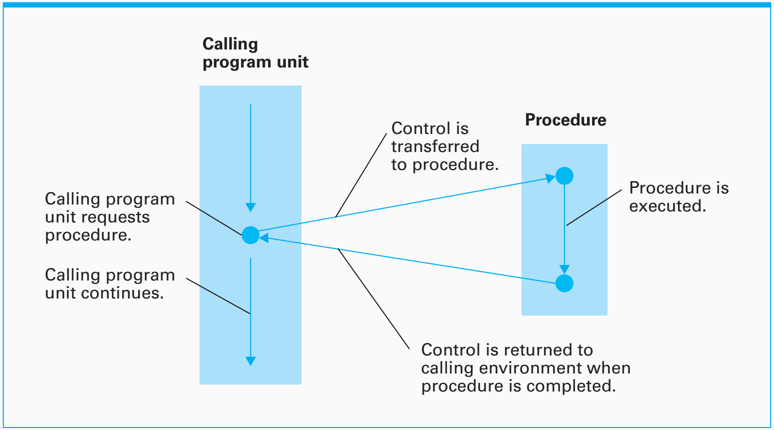
Istruzioni di controllo



```
for (int Count = 1; Count < 4; Count++)
     body ;
```

Ada, C, C++, C#, FORTRAN, Java

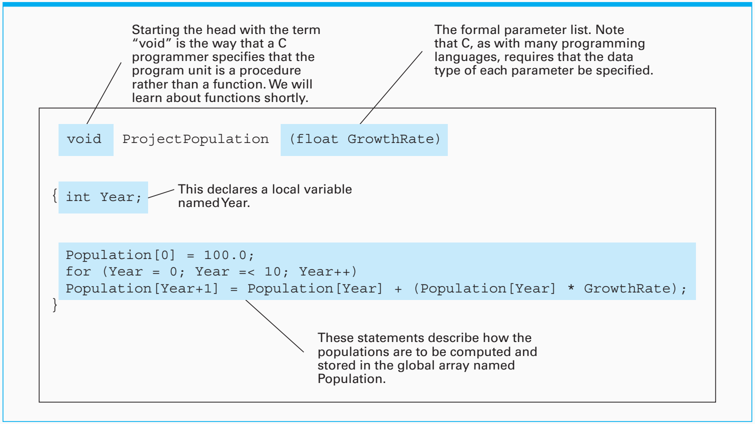
Procedure



procedure's header, (local/global variables, scopes)

Ada, C, C++, C#, FORTRAN, Java

Procedure e parametri



Starting the head with the term "void" is the way that a C programmer specifies that the program unit is a procedure rather than a function. We will learn about functions shortly.

The formal parameter list. Note that C, as with many programming languages, requires that the data type of each parameter be specified.

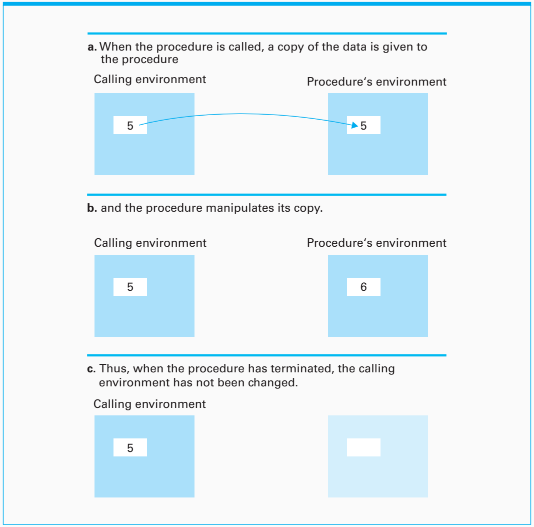
```
void  ProjectPopulation  (float GrowthRate)

{ int Year;

  Population[0] = 100.0;
  for (Year = 0; Year =< 10; Year++)
  Population[Year+1] = Population[Year] + (Population[Year] * GrowthRate);
}
```

This declares a local variable named Year.

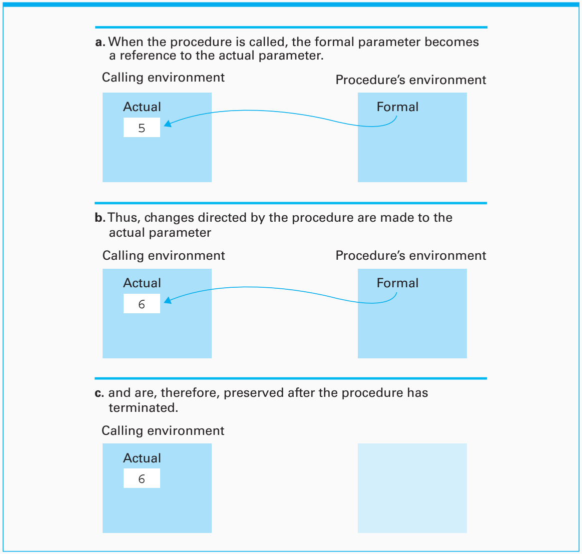
These statements describe how the populations are to be computed and stored in the global array named Population.

parametri formali/attuali

Procedure e parametri: by value

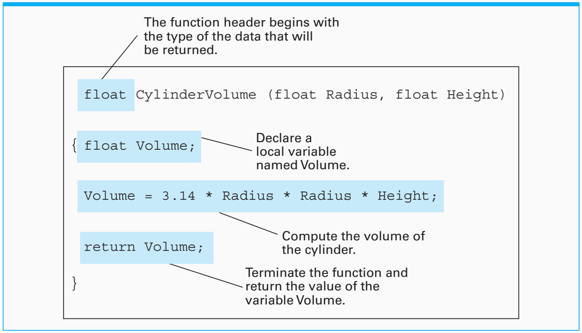

a. When the procedure is called, a copy of the data is given to the procedure
Calling environment
Procedure's environment
5
5
b. and the procedure manipulates its copy.
Calling environment
Procedure's environment
5
6
c. Thus, when the procedure has terminated, the calling environment has not been changed.
Calling environment
5

Procedure e parametri: by reference


a. When the procedure is called, the formal parameter becomes a reference to the actual parameter.
Calling environment
Procedure's environment
Actual
5
Formal
b. Thus, changes directed by the procedure are made to the actual parameter
Calling environment
Procedure's environment
Actual
6
Formal
c. and are, therefore, preserved after the procedure has terminated.
Calling environment
Actual
6

Ada, C, C++, C#, FORTRAN, Java

Funzioni

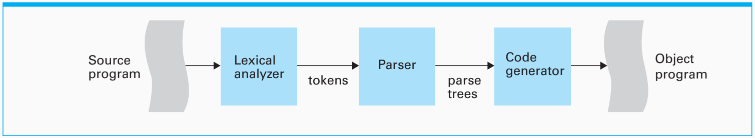


```
float CylinderVolume (float Radius, float Height)

{ float Volume;

  Volume = 3.14 * Radius * Radius * Height;

  return Volume;

}
```

The function header begins with the type of the data that will be returned.
Declare a local variable named Volume.
Compute the volume of the cylinder.
Terminate the function and return the value of the variable Volume.

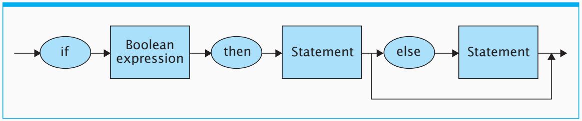
Implementazione (dei linguaggi di programmazione)

Traduzione: source to object



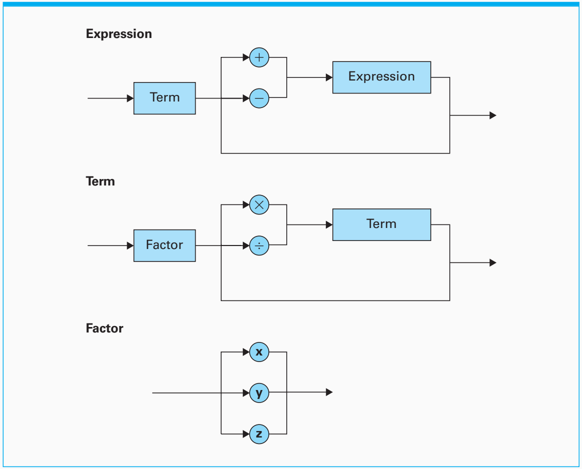
Implementazione (dei linguaggi di programmazione)

Sintassi: diagrammi



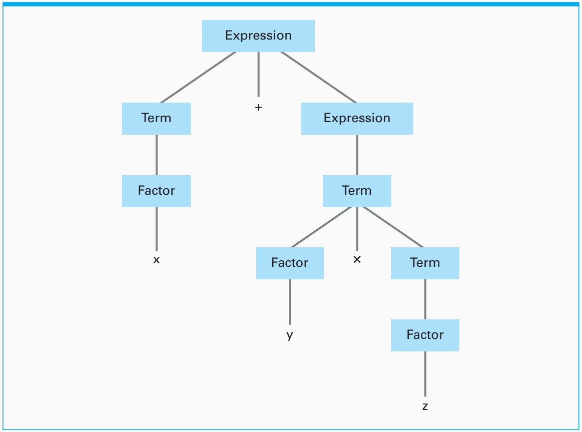
Implementazione (dei linguaggi di programmazione)

Sintassi: espressioni

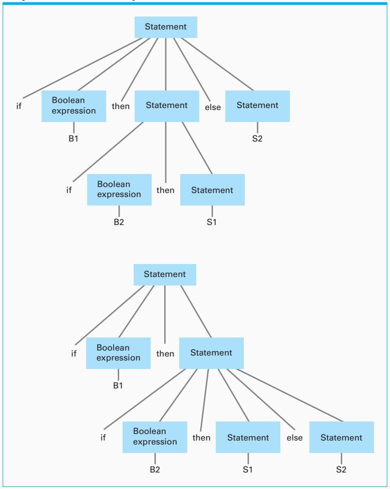


Implementazione (dei linguaggi di programmazione)

Sintassi: parsing

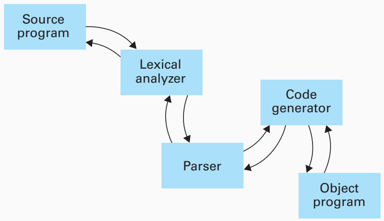


Sintassi: parsing (ambigui)



Implementazione (dei linguaggi di programmazione)

OO translation process


Source program
Lexical analyzer
Code generator
Parser
Object program