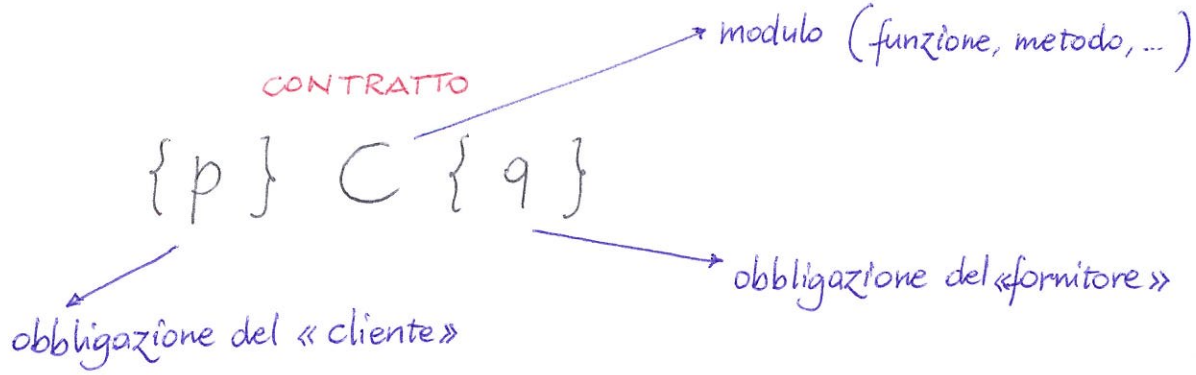
Design by contract

Programming by contract

Bertrand Meyer Design by Contract 1991



Eiffel

```
routine_name (arguments) is
    ...
    require
        precondition
    do
        body
    ensure
        postcondition
    end
```

```
class  class_name  feature
    ... declarations ...
    invariant
        ... invariant expression ...
end
```

Design by contract

JML Java Modeling Language

```
//@   requires  x >= 0.0
/*@   ensures   JMLDouble.approximatelyEqualTo
  @               (x, \result * \result, eps);
  @*/
public static double sqrt(double x) {
        ...
}
```

```
public class P {
    ...
    /*@ public invariant
      @     w >= 0;
      @*/
    ...
}
```

$> jmlc P.java il bytecode contiene istruzioni per verificare le asserzioni

Dimostrazioni di correttezza e semantica dei programmi

Fondamenti

R.W. Floyd *Assigning meanings to programs* 1967

C.A.R. Hoare *An axiomatic basis for computer programming* 1969

E.W. Dijkstra *A discipline of programming* 1976

Applicazioni pratiche

Eiffel

Java Modeling Language, Daikon

C, C++ : assert

SPARK

...

Applicazioni teoriche (?!)

Semantica dei programmi

- Qual è il "significato" di un programma ?

Dimostrazioni di correttezza

Estensioni del linguaggio di programmazione

- funzioni
- funzioni ricorsive
- tipi di dati strutturati
- classi e oggetti

Programmi concorrenti ?

La logica di Hoare e la sintesi di programmi iterativi

PROBLEMA Calcolare la radice quadrata intera di un numero K

a. Stabiliamo il criterio di correttezza (i "termini del contratto")

$$\{ K \geq 0 \} \; P \; \{ 0 \leq x^2 \leq K < (x+1)^2 \}$$

b. Ipotesi di soluzione: calcoliamo il valore richiesto per approssimazioni successive → iterazione fino a quando il valore calcolato soddisfa la postcondizione

Struttura del programma

```
x := E(K);      // preparazione di x
while  B(x,K)  do
    x := F(x,K)
endwhile
```

] W

c. Spezziamo la postcondizione per cercare un invariante di ciclo

$$0 \leq x^2 \qquad x^2 \leq K \qquad (x+1)^2 > K$$

Supponiamo di calcolare il risultato partendo da un'approssimazione per difetto, e incrementando

$\Rightarrow$ $0 \leq x^2 \wedge x^2 \leq K$ è un possibile invariante

$\Rightarrow$ $x := E(K)$ deve stabilire l'invariante

Poiché $K \geq 0$, scegliamo 0 come valore iniziale di x

d. Se $inv \equiv 0 \leq x^2 \leq K$ è un invariante per W, al termine dell'esecuzione varrà

$$inv \wedge \neg B(x,K)$$

$\Rightarrow$ dobbiamo scegliere $B(x,K)$ in modo che

$$\big(inv \wedge \neg B(x,K)\big) \rightarrow \big(inv \wedge K < (x+1)^2\big)$$

$\Rightarrow$ Poniamo $B(x,K) \equiv \boxed{(x+1)^2 \leq K}$

e. Se nel corpo dell'iterazione incrementiamo x, prima o poi $(x+1)^2$ diventerà maggiore di K

$\Rightarrow$

```
x := 0;
while (x+1)*(x+1) <= K do
    x := x + 1
endwhile
```

Esercizio: dimostrare che inv è un invariante
dimostrare la correttezza totale

Schema generale di dimostrazione

istruzione iterativa

$\{p\}\ V; W; Z\ \{q\}$

supponiamo che V e Z non contengano while

Da $Z\ \{q\}$ ricaviamo $wp(Z, q) \equiv s$ $\quad \{s\}\ Z\ \{q\}$

Cerchiamo un invariante i per W tale che $(i \wedge \neg B) \rightarrow s$ $\quad \{i\}\ W; Z\ \{q\}$

Cerchiamo una formula u tale che $\{p\}\ V\ \{u\}$ e $u \rightarrow i$ $\quad \{p\}\ V; W; Z\ \{q\}$