Ciclo `do while` in FORTRAN

Vogliamo tradurre in linguaggio FORTRAN un algoritmo che risponde a questo tipo di struttura:

```
Fino a quando e' vera questa espressione logica
                    allora:
    fai questo
    fai quest'altro
    fai quest'altra cosa ancora
```

Esempio sul metodo del Punto Fisso

$$
\begin{aligned}
x_1 &= g(x_0) \\
x_2 &= g(x_1) \\
x_3 &= g(x_2) \\
&\vdots \\
x_{k+1} &= g(x_k)
\end{aligned}
$$

Da un punto di vista numerico, supponendo di volere una soluzione approssimata con una certa tolleranza `toll`, ci si ferma nel calcolo della successione dei valori, quando $|x_{k+1} - x_k| <$ `toll` per un certo valore di k.

In FORTRAN possiamo pensare di scrivere un programma in questo modo:
Parto da un'approssimazione iniziale x_0.
Pongo `xk` =x_0.
Assegno il valore della tolleranza `toll`.
Dichiaro la variabile `scarto`= | `xkp1 - xk`|.
`xkp1` ha il significato dell'iterazione successiva rispetto a `xk`.
Quando ho solo x_0, pongo `scarto` uguale ad un valore più grande della tolleranza `toll`, ad esempio `scarto= 2*toll`.
Conto il numero delle iterazioni che farò per arrivare a convergenza mediante la variabile `iter`.
Al passo iniziale, `iter`= 0.

Quindi

```
Fino a quando (scarto > =  toll) allora:
    iter = iter +1
    xkp1=g(xk)
    scarto = abs(xkp1 - xk)
    xk = xkp1
Fine di questo ciclo
```

Cosa succede?

Quando entro nel ciclo la variabile `scarto` è certamente maggiore di `toll` perchè ho assegnato a `scarto` il valore `2*toll`.

La proposizione `(scarto > = toll)` è vera.

Allora eseguo le istruzioni del ciclo, cioè:

```
iter = iter +1, vale  1
xkp1=g(xk)  assegno a xkp1 il valore
            della g in xk
scarto = abs(xkp1 - xk)  calcolo |xkp1-xk|
xk = xkp1   assegno il valore di xkp1 a xk
```

Una volta che arrivo alle parole `Fine di questo ciclo`, devo tornare indietro, all'inizio del ciclo

`Fino a quando (scarto > = toll) allora:`

La variabile `scarto` assume un valore diverso da quello precedente: se vale `(scarto > = toll)`, allora eseguirò di nuovo le istruzioni:

```
iter = iter +1  adesso sara' 1+1=2
xkp1=g(xk)      assegno a xkp1
                il valore della g in xk
scarto = abs(xkp1 - xk)  calcolo |xkp1-xk|
xk = xkp1       assegno il valore di xkp1 a xk
```

Di nuovo andrò a controllare se (`scarto > = toll`) e, se sarà così, di nuovo eseguirò le istruzioni che sono dentro al ciclo. Altrimenti, se cioè (`scarto < toll`), non eseguirò più quelle istruzioni.

In linguaggio FORTRAN...

Tradotto in linguaggio FORTRAN avrò:

```
do while  (scarto.ge.toll)
    iter = iter +1
    xkp1=g(xk)
    scarto = abs(xkp1 - xk)
    xk = xkp1
end do
```

Più in generale:

```
do while (predicato)
   una o piu' istruzioni
end do
```

Fino a quando è vero il predicato allorà verranno eseguite le istruzioni contenute all'interno del ciclo. Poi si valuta nuovamente il predicato: se è falso, si esce dal ciclo; invece, se è vero, si eseguono nuovamente le istruzioni del ciclo. Si va avanti in questo modo fino a quando il predicato risulta essere falso.

Se invece il predicato è sempre falso non si entra mai nel ciclo `do while` e quindi le istruzioni contenute all'interno del ciclo non sono mai eseguite.

E se il predicato è sempre vero? Il ciclo durerebbe all'infinito!!!!!!! Quindi bisogna prestare molta attenzione sul predicato che si scrive in modo che non possa essere sempre vero!

Esempio: nel calcolo del punto fisso il predicato `scarto > = toll` potrebbe essere sempre vero (ad esempio se non si arriva a convergenza).... Quindi non va bene mettere solo questo predicato nel ciclo `do while`.

Conviene controllare anche il numero delle iterazioni e fissare un numero massimo di iterazioni per il nostro ciclo, in genere un numero molto grande. Per esempio, ponendo una nuova variabile, che ha il significato di numero massimo di iterazioni, `itmax` = 100 (o 500, o 1000... a seconda dei problemi), possiamo scrivere:

Algoritmo del punto fisso

```
do while ((scarto.ge.toll).and.(iter.le.itmax))
    iter = iter +1
    xkp1=g(xk)
    scarto = abs(xkp1 - xk)
    xk = xkp1
end do
```

Ora il predicato del ciclo `do while` è:
`( (scarto.ge.toll).and.(iter.le.itmax) )`, cioè deve essere sia `scarto >= toll` e sia `iter <= itmax` perchè si eseguano le istruzioni contenute all'interno del ciclo `do while`.

Il predicato diventa falso quando una delle due proposizioni logiche non è più vera cioè: o `scarto < toll` o `iter > itmax`.

Se `scarto < toll` allora il metodo è arrivato a convergenza.
Se invece `iter > itmax` allora il numero delle iterazioni è troppo grande e il metodo o sta divergendo oppure non riesce ad arrivare a convergenza secondo la tolleranza prefissata entro quel numero massimo di iterazioni (il motivo esatto lo si capisce dalla stampa delle approssimazioni per ogni iterazione).

Siamo quindi in grado di scrivere un programma sul metodo del punto fisso utilizzando il ciclo `do while`.