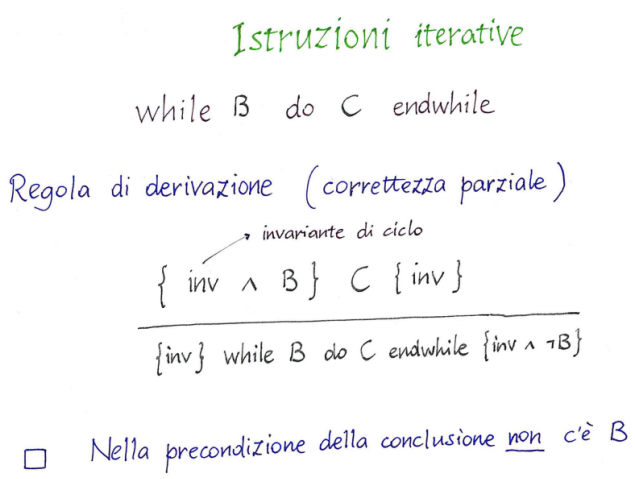
Lezione 15 27/11/2023



La regola ha una premessa, che ha come precondizione una formula (invariante) in congiunzione con la condizione di ciclo. L'invariante (se vera all'inizio) sarà vera alla fine. Durante l'esecuzione potrebbe diventare temporaneamente falsa.

Qui parliamo di correttezza parziale, ovvero supponiamo a priori che quest'istruzione iterativa termini in un tempo finito di passi.

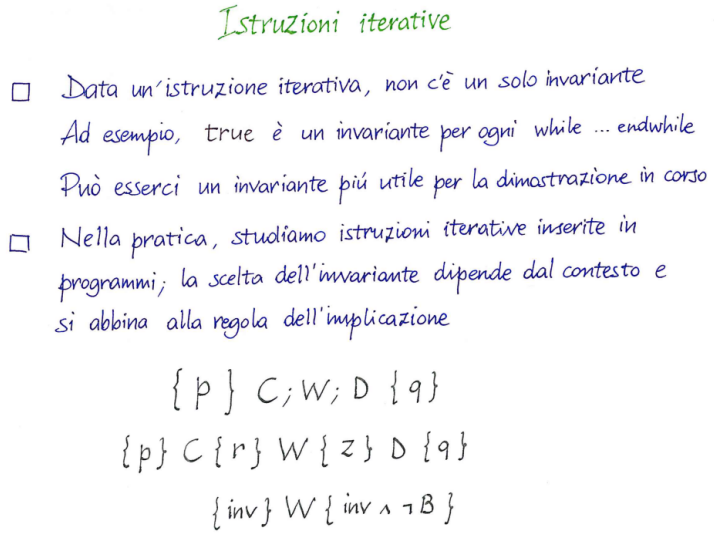
Nella conclusione la condizione di ciclo compare solo negata nella post condizione e non compare nella pre condizione.

Oggi facciamo esercizi sulla ricerca dell'invariante etc.

Non avrebbe senso dire "cerco l'invariante" perché non è solo una di solito.

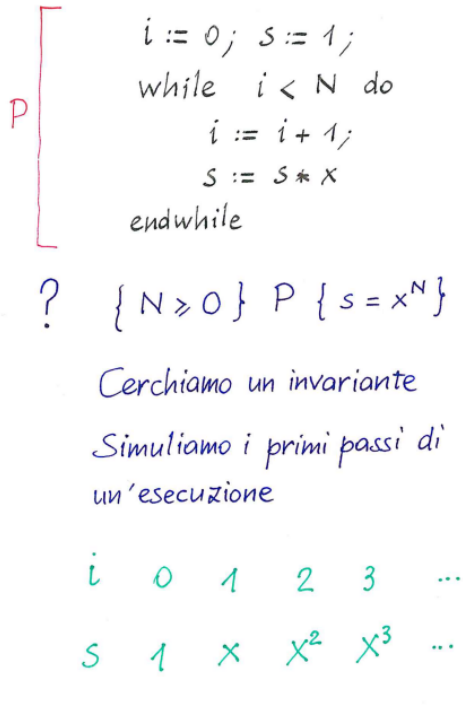
Avremo spesso lo schema in nero. L'istruzione iterativa immersa nel programma, e dovremo quindi combinare la regola di derivazione dell'iterazione con altri pezzi, come la regola per la sequenza e per l'implicazione.

In un'istruzione iterativa, quello che consideriamo



come “corpo” può essere un’istruzione complessa, potrebbe anche essere una sequenza di istruzione che include istruzioni di scelta ed eventualmente anche altre istruzioni iterative.

Esercizio



Ci sono due assegnamenti iniziali a *i* e *s*, e poi un’iterazione sulla variabile i. Questo programma calcola un’elevamento a potenza.

N è una costante che avrà un valore ben definito all’inizio.

Come dimostriamo questa ipotesi, ovvero come deriviamo questa tripla?

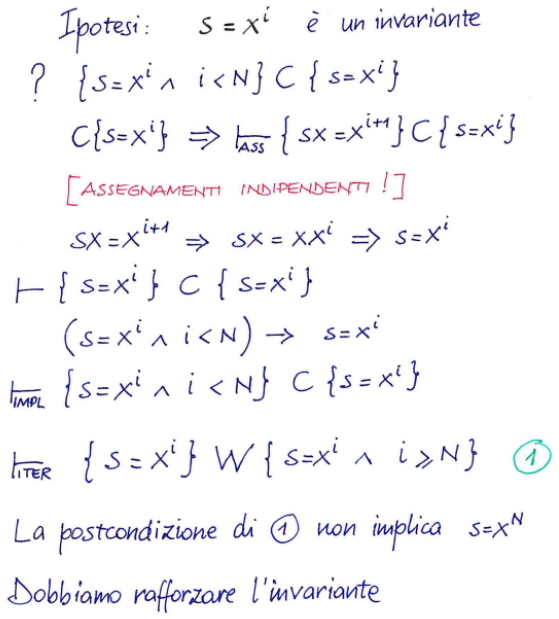
Il programma ha la sequenza di 3 istruzione: 2 iniziali e quella iterativa. Quella iterativa ha a sua volta un corpo che ha 2 istruzioni.

Cerchiamo un’invariante. La strategia consiste nel provare ad eseguire a mano qualche iterazione tenendo traccia dei valori delle variabili, costruendo una tabella che ci suggerirà l’invariante.

Costruiamo la tabella con *i* e *s*. Partiamo con *0* e *1* che sono i valori iniziali (sulla prima colonna), eseguiamo un’iterazione (manteniamo x visto che sarà un numero fisso) e avremo *1* e *x*. Continuiamo a fare altre iterazioni.

Troviamo quindi che *i* e *s* hanno una relazione fissa. Il valore di *s* è sempre *x* elevato al valore di *i*.

Per dimostrarlo dobbiamo derivare una tripla in cui come precondizione abbiamo questa presunta invariante in congiunzione con la condizione di ciclo (seconda riga prossima img).



Per derivare questa tripla, il corpo dell'iterazione è composta da 2 assegnamenti quindi useremo al regola per l'assegnamento e la regola per la sequenza. In questo caso i due assegnamenti sono indipendenti (perché toccano insiemi disgiunti di variabili) quindi potremmo anche eseguirle concorrentemente. L'indipendenza di alcuni assegnamenti eseguiti in sequenza, ci permette di eseguire in un passo solo le sostituzioni richieste dalla regola dell'assegnamento.

Per usare la regola dell'assegnamento dobbiamo partire dalla post condizione e sostituire dalle variabili che compaiono nella formula le espressioni che sono assegnate. Nella post condizione compaiono s e i, quindi a s sostituiamo sx e ad i sostituiamo i+1.

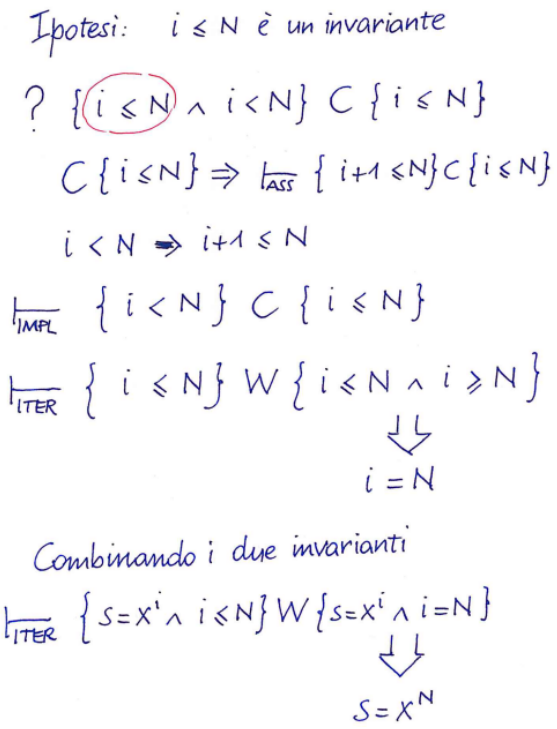
Ora confrontiamo questa tripla che abbiamo trovato (la nuova pre condizione, terza riga) con quella che è il nostro obiettivo (la pre condizione precedente, seconda riga).

Quindi alla riga 7 (prima di impl) abbiamo dimostrato che è un'invariante.

Possiamo dividere per x entrambi i lati e troviamo s=x^i. Dovremmo qui rispettare le regole della matematica e dire che x deve essere diverso da 0 ma in questo caso se

x fosse 0 (non ho capito cosa succederebbe al codice) quindi lo ignoriamo. La pre condizione che troviamo è quindi un rafforzamento rispetto a quella precedente

Dopo aver applicato le altre regole, trovo che la post condizione finale è diversa rispetto a quella che stavo cercando di dimostrare io. Dovrei quindi cercare di rafforzare l'invariante. Se potessimo dimostrare che alla fine dell'esecuzione i ≤ N (oltre a i ≥ N che già abbiamo) allora avremmo che a fine esecuzione i=N e questo farebbe diventare la post condizione uguale a quella che stiamo dimostrando.



Candidiamo come invariante i≤N.

Non consideriamo il secondo assegnamento perchè non intacca questa formula, quindi applichiamo la regola solo all'incremento di i. Otteniamo una tripla, che dice che se all'inizio i+1≤N allora alla fine i≤N.

i+1≤N equivale a dire che i<N.

Quindi per implicazione abbiamo dimostrato che i<N è precondizione di i≤N.

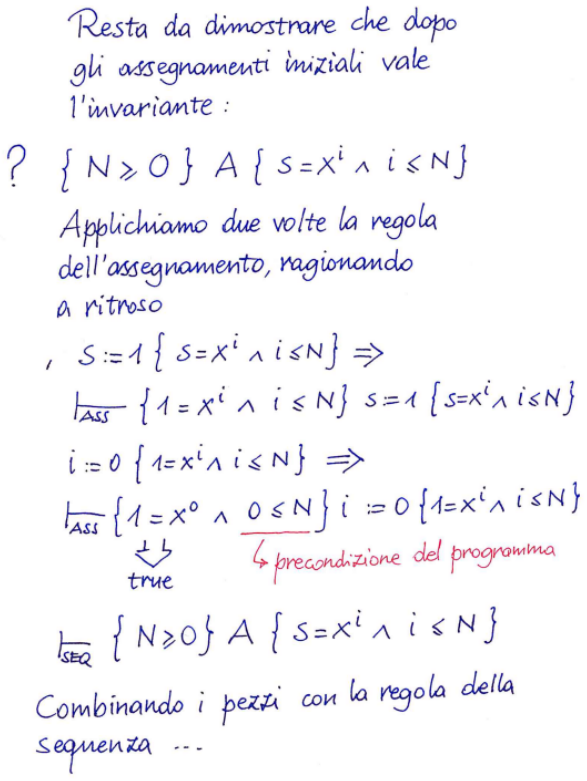
Abbiamo dimostrato che i≤N è un'invariante di ciclo.

Possiamo quindi applicare la regola dell'iterazione e affermare che vale la tripla. La post condizione si può riscrivere come i=N. Abbiamo dimostrato che al termine dell'istruzione iterativa i=N.

Abbiamo derivato separatamente due invarianti. Questo ci permette di concludere che la loro congiunzione logica è anch'essa un'invariante.

Quindi la post condizione finale che troviamo è uguale alla post condizione della tripla iniziale che volevamo dimostrare.

Però ci manca da dimostrare la pre condizione.

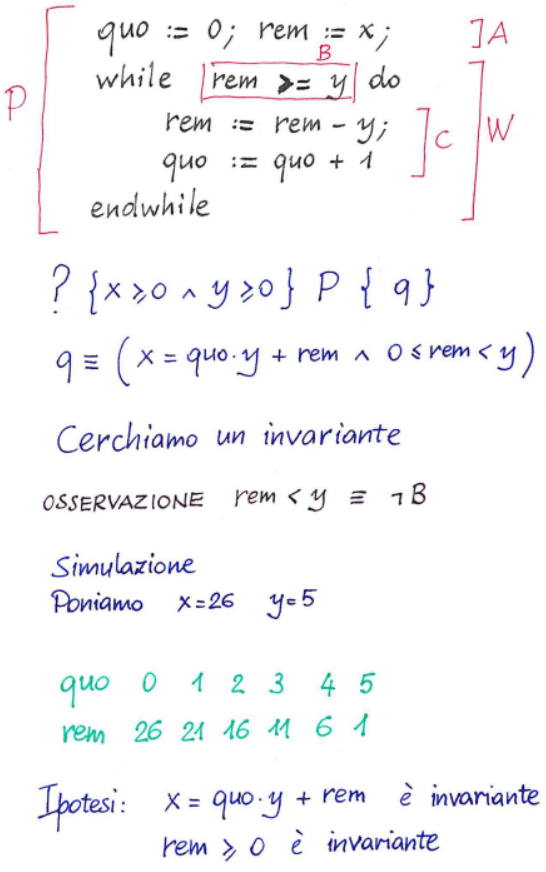


Dobbiamo dimostrare che se N≥0 all'inizio dell'esecuzione, dopo gli assegnamenti vale l'invariante.

Combinando la regola dell'assegnamento e della sequenza, partiamo dalla post condizione, applichiamo l'assegnamento, e sperare che alla fine dei passaggi possiamo applicare l'implicazione per agganciarci alla precondizione data.

Qui abbiamo fatto due assegnamenti staccati anche se li avremmo potuti fare insieme visto che sono indipendenti.

Esercizio



Il programma fa la divisione tra x e y, nel rem ci sarà il resto della divisione intera, se x e y sono ≥ 0 all'inizio.

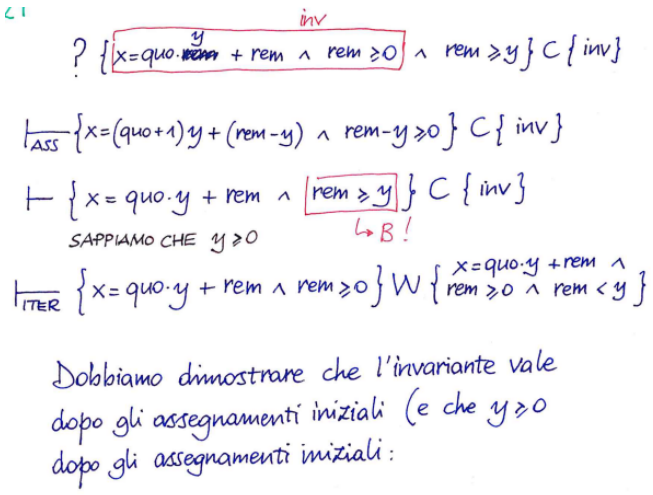
Nella post condizione che vogliamo ottenere, rem < y, è la negazione della condizione di ciclo.

Per il momento stiamo ragionando nel contesto della correttezza parziale, quindi supponiamo che il programma termini.

Cerchiamo l'invariante. Come prima facciamo la tabella, facendo qualche passo. Teniamo x=26 e y=5 inizialmente così possiamo fare qualche passaggio.

In questi casi (dove l'invariante non è immediata da trovare) per trovare l'invariante possiamo partire dalla post condizione, prendiamo un pezzo e controlliamo se quello è l'invariante, prendiamo x = quo * y + rem.

La pre condizione è data dall'invariante e la condizione di ciclo, e la post condizione è l'invariante. Per derivare questa tripla, vediamo che il corpo dell'iterazione è fatto da due assegnamenti in sequenza, quindi partiamo dalla post condizione e applichiamo la regola dell'assegnamento. I due assegnamenti sono indipendenti quindi li facciamo insieme.



Possiamo quindi applicare la regola dell'iterazione prendendo l'invariante come pre condizione e come post condizione l'invariante e la condizione di ciclo negata. Quid troviamo quindi la post condizione della tripla che stiamo dimostrando quindi va già bene.

Dobbiamo sostituire a rem, rem-y. Poi semplifichiamo, e troviamo come risultato è quello che cercavamo.

(pubblicherà altri esercizi svolti che magari aiutano in preparazione)