ESERCIZIO1

PREMESSA

Per risolvere problemi spesso esistono delle regole che, dai dati del problema, permettono di calcolare o *dedurre* la soluzione. Questa situazione si può descrivere col termine

regola(<sigla>,<lista antecedenti>,<conseguente>)

che indica una regola di nome <sigla> che consente di dedurre <conseguente> conoscendo tutti gli elementi contenuti nella <lista antecedenti>, detta anche *premessa*. Problemi "facili" possono essere risolti con una sola regola; per problemi "difficili" una sola regola non basta a risolverli, ma occorre applicarne diverse in successione.

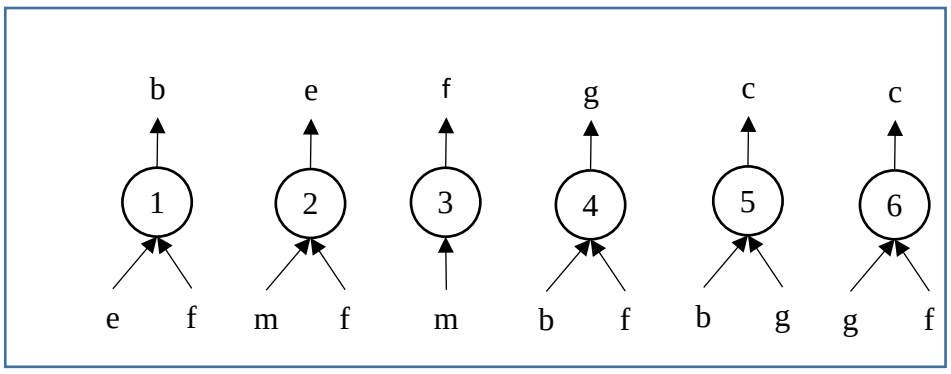
Si considerino le seguenti regole:

| | | |
|---|---|---|
| regola(1,[e,f],b) | regola(2,[m,f],e) | regola(3,[m],f) |
| regola(4,[b,f],g) | regola(5,[b,g],c) | regola(6,[g,f],c) |

Per esempio la regola 1 dice che si può calcolare (o dedurre) **b** conoscendo **e** ed **f** (cioè gli elementi della lista [e,f]); conoscendo **b** ed **f** (cioè gli elementi della lista [b,f]) è possibile dedurre **g** con la regola 4. Quindi, a partire da **e** ed **f** è possibile dedurre prima **b** (con la regola 1) e poi **g** (con la regola 4).

Un *procedimento di deduzione* (o deduttivo, o di calcolo) è rappresentato da un *insieme di regole da applicare in sequenza opportuna* per dedurre un certo elemento (incognito) a partire da certi dati: quindi può essere descritto dalla lista delle sigle di queste regole. Il procedimento [1,4] descrive la soluzione del problema: "dedurre **g** a partire da **e** ed **f**".

Una maniera grafica per rappresentare le regole è quella mostrata nella seguente figura: consiste nell'associare un albero (rovesciato) ad ogni regola: la radice (in alto) è il conseguente, le foglie (in basso) sono gli antecedenti.




Con questa rappresentazione grafica, risolvere il problema "dedurre **g** a partire da **e** ed **f**" è particolarmente facile; si cerca un "albero" (cioè una regola) che ha come radice l'incognita (cioè **g**): in questo caso ne esiste solo uno che è la regola 4: si veda la seguente figura 1 a sinistra.

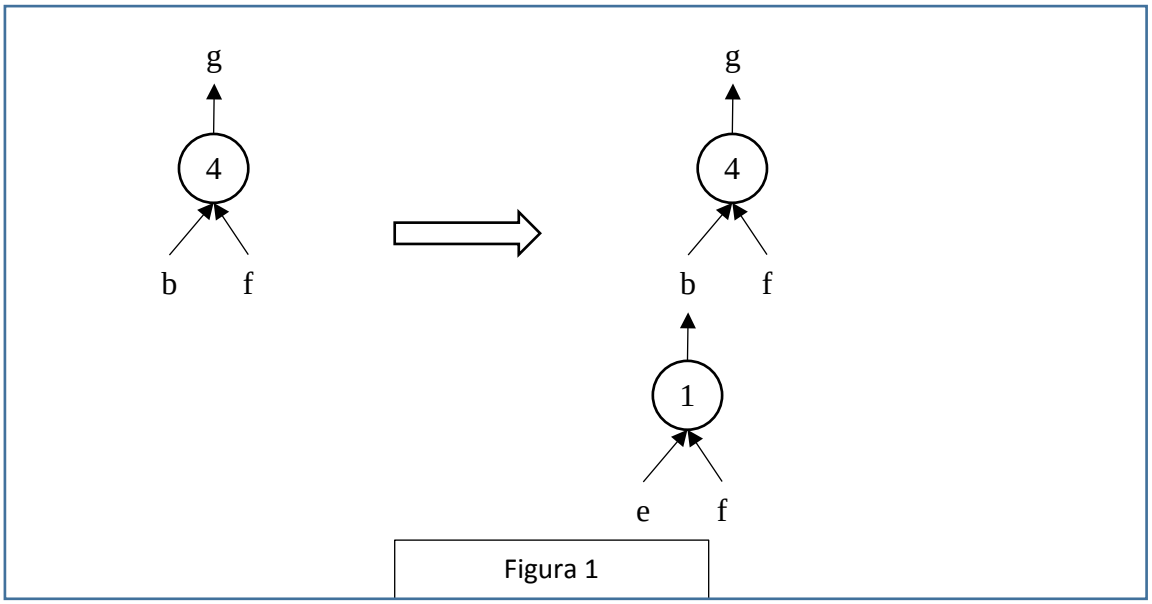


Figura 1

Le foglie di questo albero (**b** ed **f**) *non* sono tutte note: quelle note (**f** in questo caso) sono vere e proprie foglie, quelle incognite (**b** in questo caso) vanno considerati come "anelli" a cui "appendere" un altro albero; quindi bisogna continuare *sviluppando* la foglia incognita **b**, cioè "*appendendo*" a **b** l'albero rappresentato dalla regola 1, come illustrato nella figura 1 a destra.
Adesso tutte le foglie dell'albero così ottenuto (**e** ed **f**) sono note e il problema è risolto.
Si può anche dire che un albero le cui foglie sono tutte note rappresenta un procedimento per dedurre la "radice" a partire dalle "foglie". Per costruire la lista corrispondente occorre *partire dal basso*: prima si applica la regola 1, che utilizza solo i dati; poi si può applicare la regola 4. Il procedimento è quindi (individuato dalla lista) [1,4].

Come altro esempio, in figura 2 è illustrata la soluzione del problema: "dedurre **e** a partire da **m**". Tale soluzione si ottiene costruendo successivamente i due alberi mostrati; il procedimento è [3,2].

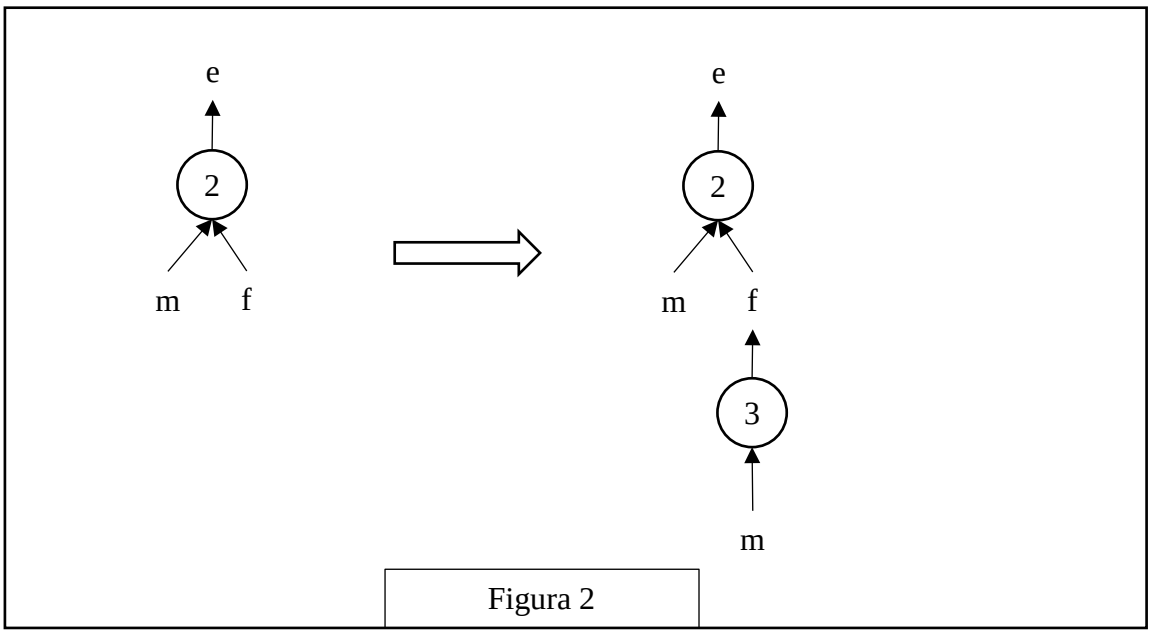


Figura 2

N.B. Nelle liste richieste occorre elencare le sigle delle regole nell'ordine che corrisponde alla sequenza di applicazione: la prima (a sinistra) della lista deve essere la sigla che corrisponde alla prima regola da applicare (che ha come antecedenti solo dati); l'ultima (a destra) deve essere la sigla della regola che ha come conseguente l'elemento incognito da dedurre.
Nella lista non ci sono regole *ripetute* (infatti un procedimento di deduzione è un *insieme* di regole da applicare in opportuna sequenza). L'applicazione di una regola rende disponibile il conseguente da utilizzare (come antecedente) nell'applicazione di regole successive.

La lista associata a un (ben preciso) procedimento si costruisce quindi per passi successivi a partire dal primo elemento che è la sigla della prima regola da applicare; ad ogni passo, se ci fossero più regole applicabili, occorre dare la precedenza (nella lista) a quella con sigla *inferiore* (questo per rendere *unica* la lista associata al procedimento).

N.B. In alcuni casi esistono più procedimenti deduttivi possibili che permettono di ricavare un certo elemento dagli stessi dati, in maniere diverse (cioè con alberi diversi e quindi con insiemi diversi di regole). Per esempio il problema "dedurre **c** a partire da **b** ed **f**" (dalle regole viste sopra) ha due distinti procedimenti risolutivi; gli alberi relativi ai due procedimenti sono mostrati nella seguente figura 3.

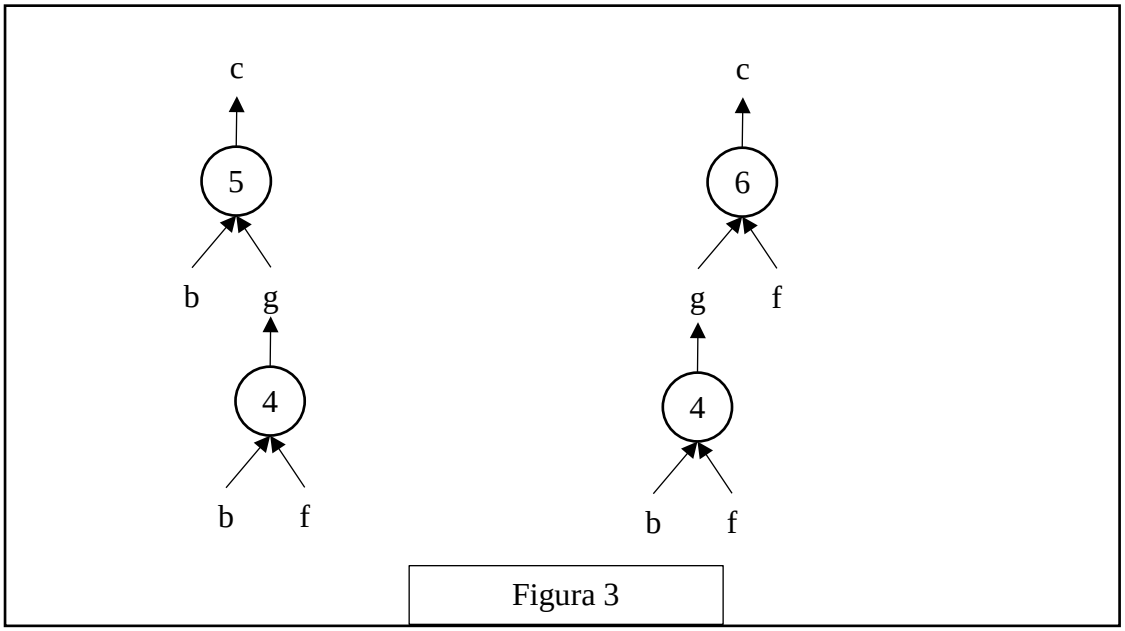



Figura 3

Le liste associate sono, rispettivamente, [4,5] e [4,6].

In un procedimento deduttivo, il numero di regole *differenti* coinvolte (e, quindi, anche il numero di elementi della lista corrispondente al procedimento) si dice *lunghezza* del procedimento.

PROBLEMA

Sono date le seguenti regole:

| | | |
|---|---|---|
| regola(1,[r,d],k) | regola(2,[r],q) | regola(3,[r,g],n) |
| regola(4,[f,u],x) | regola(5,[r,q],g) | regola(6,[w,f],u) |
| regola(7,[k,r],n) | regola(8,[d],r) | regola(9,[u,x],k) |
| regola(10,[n,g],w) | regola(11,[w,g],d) | regola(12,[y,p],d) |

Trovare:

1. la lista L1 che descrive il procedimento per dedurre **k** conoscendo **w** e **f**;
2. la lista L2 che descrive il procedimento per dedurre **n** con 4 regole conoscendo **y** e **p**;
3. la lista L3 che descrive il procedimento per dedurre **n** con 5 regole conoscendo **y** e **p**.

| | |
|---|---|
| L1 | [ ] |
| L2 | [ ] |
| L3 | [ ] |

ESERCIZIO 2

PREMESSA
In un deposito di minerali esistono esemplari di vario peso e valore individuati da sigle di riconoscimento. Ciascun minerale è descritto da una termine che contiene le seguenti informazioni:

tab(<sigla del minerale>, <valore in euro>, <peso in Kg>).

Il deposito contiene i seguenti minerali:

| | | |
|---|---|---|
| tab(m1,33,84) | tab(m2,36,86) | tab(m3,32,87) |
| tab(m4,31,83) | tab(m5,38,82) | tab(m6,37,88) |
| tab(m7,35,81) | tab(m8,34,88) | tab(m9,34,89) |

PROBLEMA
Disponendo di un autocarro con portata massima di 170 Kg, trovare la lista L1 delle sigle di 2 minerali diversi trasportabili con questo autocarro che consente di trasportare il massimo valore possibile.
Disponendo di un autocarro con portata massima di 250 Kg, trovare la lista L2 delle sigle di 3 minerali diversi trasportabili con questo autocarro che consente di trasportare il massimo valore possibile.
Disponendo di un autocarro con portata massima di 340 Kg, trovare la lista L3 delle sigle di 4 minerali diversi trasportabili con questo autocarro che consente di trasportare il massimo valore possibile.

N.B. Nelle liste, elencare le sigle in ordine crescente; per le sigle si ha il seguente ordine:
m1<m2<… <m9.

| | |
|---|---|
| L1 | [ ] |
| L2 | [ ] |
| L3 | [ ] |

ESERCIZIO 3

PROBLEMA
A visitor to a farm asked a farm worker how many fowl and how many oxen were in the farm. The worker replied: “There are 40 heads and 124 feet, and with that information you should be able to tell how many of each there are.” Help the visitor, putting your reply in the boxes below.

| | |
|---|---|
| fowl | |
| oxen | |

ESERCIZIO 4

PROBLEMA
Si consideri la seguente procedura PROVA1.

```
procedure PROVA1;
variables A, B, K, J integer;
A ← 0;
B ← 0;
for J from 1 to 4 step 1 do
        for K from 1 to 3 step 1 do
                A ← -J*(A+1);
                B ← -K*(A+B);
         endfor;
endfor;
output A,B;
endprocedure;
```

Determinare il valore di output di A e B.

| | |
|---|---|
| A | |
| B | |

ESERCIZIO 5

PREMESSA
La ripetizione di un gruppo di azioni può essere comandata non solo con la struttura "for" già vista, ma anche con la struttura "while", illustrata dal seguente esempio.

```
B ← 10;
A ←  0;
K  ← 0;
while A < B do
        K ← K + 1;
        A ← K × K + A;
endwhile;
output A;
```

Se il predicato A < B è vero, il ciclo viene ripetuto; quando diventa falso si passa alla esecuzione della istruzione successiva a "endwhile". In questo caso il valore di B rimane fisso a 10, mentre quello di A cambia dopo ogni iterazione assumendo i seguenti valori: 1, 5, 14. Dopo la terza iterazione il valore di A non è più minore di quello di B e il ciclo si arresta: in output si ha quindi 14.

PROBLEMA
Si consideri la seguente procedura PROVA2.

```
procedure PROVA2;
variables A, B, K, C integer;
A ←  1;
B ←  1;
C ← 100;
K ←  0;
while C > B do
        K ← K +1;
        A ← A + K**2;
        B ← B × K;
        C ← C+100;
endwhile;
output A;
endprocedure;
```

Determinare il valore di output di A.

| A | |
|---|---|

ESERCIZIO 6

PROBLEMA
Si consideri la seguente procedura PROVA3.

```
procedure PROVA3;
variables A, B, K, Z integer;
Z ← 0;
for K from 1 to 3 step 1 do
        input A;
        B ← 0;
        while B < 1000 do
                B ← (A+B) × (A+1);
        endwhile;
        Z ← Z + B;
endfor;
output Z;
endprocedure;
```

Se i valori di input per A sono 12, 9 e 15 calcolare il valore di output per Z.

| Z | |
|---|---|