Relazione del progetto relativo l'insegnamento di Programmazione Logica e Funzionale per la sessione estiva 2021/2022.

Relatori:
Di Fabrizio Giacomo
Montanari Matteo Marco

Docente:
prof. Marco Bernardo

Maggio 2022

1 Specifica del problema

Scrivere un programma Haskell e un programma Prolog che effettuano le seguenti operazioni:

1. Dati due insiemi finiti di numeri interi, verificano ricorsivamente se un insieme è sottoinsieme dell' altro.

2. Dati due insiemi finiti di numeri interi, verificano se i due insiemi sono uguali.

3. Dato un insieme finito di numeri interi, calcolano ricorsivamente l'insieme delle parti.

4. Dato un insieme finito di numeri interi, calcolano ricorsivamente la somma dei numeri pari all'interno dell'insieme.

5. Dato un insieme finito di numeri interi, calcolano ricorsivamente la somma dei numeri dispari all'interno dell'insieme.

2 Analisi del problema

2.1 Dati di ingresso del problema

L'input del problema è costituito da due insiemi finiti di numeri interi.

2.2 Dati di uscita del problema

L'output del problema è costituito da:

- Un valore booleano che esprime il fatto che il primo insieme è sottoinsieme del secondo.
- Un valore booleano che esprime il fatto che il secondo insieme è sottoinsieme del primo.
- Un valore booleano che esprime il fatto che il primo insieme è uguale al secondo.
- Un insieme finito che costituisce l'insieme delle parti del primo insieme.
- Un insieme finito che costituisce l'insieme delle parti del secondo insieme.
- Un valore intero che corrisponde alla somma dei numeri pari all'interno del primo insieme.
- Un valore intero che corrisponde alla somma dei numeri pari all'interno del secondo insieme.
- Un valore intero che corrisponde alla somma dei numeri dispari all'interno del primo insieme.
- Un valore intero che corrisponde alla somma dei numeri dispari all'interno del secondo insieme.

2.3 Relazioni intercorrenti tra i dati del problema

Dati due insiemi finiti A e B, allora:

$$A \subseteq B \iff \forall a \in A \implies a \in B \tag{1}$$

$$A = B \iff A \subseteq B \wedge B \subseteq A \tag{2}$$

$$\mathcal{P}(A) = \{A' \mid A' \subseteq A\} \tag{3}$$

$$\text{Somma Pari}(A) = \sum_{\{p \in A \,|\, p=2z,\, z \in \mathbb{Z}\}} p \tag{4}$$

$$\text{Somma Dispari}(A) = \sum_{\{d \in A \,|\, d=2z+1,\, z \in \mathbb{Z}\}} d \tag{5}$$

Inoltre, valgono i seguenti fatti utili alla risoluzione del problema:

- L'insieme delle parti di insiemi finiti di numeri interi può essere definito ricorsivamente come segue:

$$\mathcal{P}(A) = \begin{cases} \{\emptyset\} & \text{se } A = \emptyset \\ \{A\} \cup \bigcup_{x \in A} \mathcal{P}(A \setminus x) & \text{se } A \neq \emptyset \end{cases}$$

  Questa definizione è ben posta poiché se A è finito, sia $|A| = n \in \mathbb{N}$, allora:

$$|\mathcal{P}(A)| = 2^{|A|} \in \mathbb{N}$$

  Esempio:

$$\mathcal{P}(\{1,2\}) = \{\{1,2\}\} \cup \mathcal{P}(\{2\}) \cup \mathcal{P}(\{1\})$$

  dove:

$$\mathcal{P}(\{2\}) = \{\{2\}\} \cup \mathcal{P}(\emptyset) = \{\{2\}, \emptyset\}$$
$$\mathcal{P}(\{1\}) = \{\{1\}\} \cup \mathcal{P}(\emptyset) = \{\{1\}, \emptyset\}$$

  quindi:

$$\mathcal{P}(\{1,2\}) = \{\{1,2\}, \{2\}, \emptyset, \{1\}\}$$

- La funzione Somma Pari(A) ignora i numeri dispari in A, in simboli:

  $\forall x \in \{d \in A \,|\, d = 2z{+}1,\, z \in \mathbb{Z}\} \quad \text{Somma Pari}(A) = \text{Somma Pari}(A \backslash \{x\})$

- La funzione Somma Dispari(A) ignora i numeri pari in A, in simboli:

  $\forall x \in \{p \in A \,|\, p = 2z,\, z \in \mathbb{Z}\} \quad \text{Somma Dispari}(A) = \text{Somma Dispari}(A \backslash \{x\})$

3 Progettazione dell'algoritmo

3.1 Scelte di progetto

I vari insiemi di numeri interi utilizzati all'interno del programma per rappresentare i dati di input, output e per realizzare le computazioni necessarie a risolvere il problema, sono rappresentati come liste di numeri interi. Non conoscendo a priori le cardinalità dei vari insiemi, tali liste verranno allocate dinamicamente in modo da gestire insiemi di interi di dimensione arbitraria purché finita. Elementi duplicati all'interno dei due insiemi di input sono tollerati ma verranno eliminati prima di effettuare le computazioni necessarie alla risoluzione del problema in quanto, dato che essi rappresentano insiemi, la loro molteplicità non è rilevante per la corretta risoluzione del problema nonché mantenerla aumenterebbe la complessità computazionale dell'algoritmo. Tutte le computazioni espresse da specifica sono effettuate in maniera ricorsiva.

3.2 Passi dell'algoritmo

I passi dell'algoritmo differiscono in base al tipo di computazione effettuata.

1. Sottoinsieme:
    - *Caso base*: Se il primo insieme è vuoto, allora per ogni altro secondo insieme, il primo insieme sarà sempre sottoinsieme del secondo.
    - *Caso generale*: Se il primo insieme non è vuoto, si verifica se il primo elemento del primo insieme appartiene al secondo insieme e poi si procede ricorsivamente sulla restante parte del primo insieme e sul secondo insieme. Nel caso in cui un qualsiasi elemento del primo insieme non appartenga anche al secondo insieme allora il primo insieme non è sottoinsieme del secondo.
2. Uguaglianza:
    - Si verifica che il primo insieme è sottoinsieme del secondo insieme e poi che il secondo insieme è sottoinsieme del primo. Se entrambe le relazioni risultano vere allora l'ugualianza è verificata, altrimenti no. Per fare ciò si utilizza l'algoritmo ricorsivo per verificare se un insieme è sottoinsieme di un altro.
3. Insieme delle parti:

- *Caso base*: Se l'insieme è vuoto, allora l'insieme delle parti di tale insieme è il singoletto contenente l'insieme vuoto.
- *Caso generale*: Se l'insieme non è vuoto, sia A, allora si unisce il singoletto A con l'unione degli insiemi delle parti calcolati ricorsivamente eliminando progressivamente da A un elemento di A stesso. In formule:

$$\mathcal{P}(A) = \{A\} \cup \bigcup_{x \in A} \mathcal{P}(A \setminus x)$$

4. Somma Pari:

   - *Caso base*: Se l'insieme è vuoto, allora la somma degli elementi pari all'interno dell'insieme è 0.
   - *Caso generale*: Se l'insieme non è vuoto, si verifica se il primo elemento dell'insieme è pari e in tal caso lo si somma al risultato della chiamata ricorsiva sulla restante parte dell'insieme. Nel caso in cui il primo elemento dell'insieme sia dispari, questo elemento non viene considerato nella somma e si procede ricorsivamente sulla restante parte dell'insieme.

5. Somma Dispari:

   - *Caso base*: Se l'insieme è vuoto, allora la somma degli elementi dispari all'interno dell'insieme è 0.
   - *Caso generale*: Se l'insieme non è vuoto, si verifica se il primo elemento dell'insieme è dispari e in tal caso lo si somma al risultato della chiamata ricorsiva sulla restante parte dell'insieme. Nel caso in cui il primo elemento dell'insieme sia pari, questo elemento non viene considerato nella somma e si procede ricorsivamente sulla restante parte dell'insieme.

4 Implementazione dell'algoritmo

File sorgente: operazioni_insiemistiche.hs

```
-- Progetto di Programmazione Logica e Funzionale a.a. 2021/2022
-- Docente: prof. Marco Bernardo
-- Studenti: Giacomo Di Fabrizio, 301591; Matteo Marco Montanari, 299166.
-- Linguaggio Haskell

main :: IO ()
main = do putStrLn "Programma Haskell per effettuare le seguenti operazioni
                    su insiemi di numeri interi:"
          putStrLn "1) Dati due insiemi di numeri interi, verifica
                    ricorsivamente se un insieme e' sottoinsieme dell' altro."
          putStrLn "2) Dati due insiemi di numeri interi, verifica se i due
                    insiemi sono uguali."
          putStrLn "3) Dato un insieme di numeri interi, calcola ricorsivamente
                    l'insieme delle parti."
          putStrLn "4) Dato un insieme di numeri interi, calcola ricorsivamente
                    la somma dei numeri pari all'interno dell'insieme."
          putStrLn "5) Dato un insieme di numeri interi, calcola ricorsivamente
                    la somma dei numeri dispari all'interno dell'insieme."
          putStrLn "Inserire il primo insieme di numeri interi racchiuso tra
                    parentesi quadre:"
          i1 <- getLine
          let as = elimina_duplicati (read i1 :: [Int])
          putStr ">> Primo insieme privo di eventuali elementi ripetuti: "
          putStrLn $ show as
          putStrLn "Inserire il secondo insieme di numeri interi racchiuso tra
                    parentesi quadre:"
          i2 <- getLine
          let bs = elimina_duplicati (read i2 :: [Int])
          putStr ">> Secondo insieme privo di eventuali elementi ripetuti: "
          putStrLn $ show bs
          putStr "1) Il primo insieme e' sottoinsieme del secondo? "
          putStrLn $ show (sottoinsieme as bs)
          putStr "1) Il secondo insieme e' sottoinsieme del primo? "
          putStrLn $ show (sottoinsieme bs as)
          putStr "2) I due insiemi sono uguali? "
          putStrLn $ show (uguali as bs)
          putStr "3) Insieme delle parti del primo insieme: "
          putStrLn $ show (insieme_delle_parti as)
          putStr "3) Insieme delle parti del secondo insieme: "
          putStrLn $ show (insieme_delle_parti bs)
```

```
          putStr "4) Somma dei numeri pari del primo insieme: "
          putStrLn $ show (somma_valori_pari as)
          putStr "4) Somma dei numeri pari del secondo insieme: "
          putStrLn $ show (somma_valori_pari bs)
          putStr "5) Somma dei numeri dispari del primo insieme: "
          putStrLn $ show (somma_valori_dispari as)
          putStr "5) Somma dei numeri dispari del secondo insieme: "
          putStrLn $ show (somma_valori_dispari bs)

{- La funzione elimina_duplicati elimina i valori duplicati all'interno in
   un insieme:
   - il primo e unico argomento della funzione e' l'insieme del quale si
     vogliono eliminare i valori duplicati-}
elimina_duplicati :: [Int] -> [Int]
elimina_duplicati []       = []
elimina_duplicati (x : xs) = x : elimina_duplicati (filter (/= x) xs)

{- La funzione sottoinsieme verifica se un insieme e' sottoinsieme di un
   altro insieme:
   - il primo argomento e' il primo dei due insiemi;
   - il secondo argomento e' il secondo dei due insiemi.-}
sottoinsieme :: [Int] -> [Int] -> Bool
sottoinsieme [] _                    = True
sottoinsieme (x : xs) ys | elem x ys = sottoinsieme xs ys
                         | otherwise = False

{- La funzione uguali verifica l'ugualianza tra due insiemi:
    - il primo argomento e' il primo dei due insiemi;
    - il secondo argomento e' il secondo dei due insiemi.
    L'uso di sottoinsieme consente di verificare se gli insiemi sono
    rispettivamente l'uno il sottoinsieme dell'altro e l'uso di congiunzione
    consente di calcolare il valore di verita' della congiunzione delle due
    espressioni logiche risultato della funzione sottoinsieme-}
uguali :: [Int] -> [Int] -> Bool
uguali x y = congiunzione (sottoinsieme x y) (sottoinsieme y x)

{- La funzione congiunzione calcola la congiunzione logica tra due espressioni:
    - il primo argomento e' la prima espressione logica;
    - il secondo argomento e' la seconda espressione logica. -}
congiunzione :: Bool -> Bool -> Bool
congiunzione False _ = False
congiunzione True x  = x
```

```
{- La funzione insieme_delle_parti calcola l'insieme delle parti di un insieme:
   - il primo e unico argomento della funzione e' l'insieme del quale si vuole
   calcolare l'insieme delle parti-}
insieme_delle_parti :: [Int] -> [[Int]]
insieme_delle_parti []       = [[]]
insieme_delle_parti (x : xs) = [zs | ys <- insieme_delle_parti xs,
                                zs <- [ys, (x : ys)] ]

{- La funzione somma_valori_pari calcola la somma dei valori pari
   all'interno di un insieme:
   - il primo argomento e' l'insieme di cui si vuole calcolare la somma
     dei numeri pari al suo interno.
   L'uso di even consente di verificare se l'elemento preso in considerazione
   dalla funzione e' pari.-}
somma_valori_pari :: [Int] -> Int
somma_valori_pari [] = 0
somma_valori_pari (x : xs) | even x    =  x + somma_valori_pari(xs)
                           | otherwise = somma_valori_pari(xs)

{- La funzione somma_valori_dispari calcola la somma dei valori dispari
   all'interno di un insieme:
   - il primo argomento e' l'insieme di cui si vuole calcolare la somma
     dei numeri dispari al suo interno.
   L'uso di odd consente di verificare se l'elemento preso in considerazione
   dalla funzione e' dispari. -}
somma_valori_dispari :: [Int] -> Int
somma_valori_dispari [] = 0
somma_valori_dispari (x : xs) | odd x     = x + somma_valori_dispari(xs)
                              | otherwise = somma_valori_dispari(xs)
```

File sorgente: operazioni_insiemistiche.pro

```
/* Progetto di Programmazione Logica e Funzionale a.a. 2021/2022
Docente: prof. Marco Bernardo
Studenti: Giacomo Di Fabrizio, 301591; Matteo Marco Montanari, 299166.
Linguaggio Prolog
*/

main :- write('Programma Prolog per effettuare le seguenti operazioni su
               insiemi di numeri interi:'),nl,
        write('1) Dati due insiemi di numeri interi, verifica ricorsivamente se
               un insieme e' sottoinsieme dell altro.'),nl,
        write('2) Dati due insiemi di numeri interi, verifica se i due insiemi
               sono uguali.'),nl,
        write('3) Dato un insieme di numeri interi, calcola ricorsivamente
               l insieme delle parti.'),nl,
        write('4) Dato un insieme di numeri interi, calcola ricorsivamente
               la somma dei numeri pari all interno dell insieme.'),nl,
        write('5) Dato un insieme di numeri interi, calcola ricorsivamente
               la somma dei numeri dispari all interno dell insieme.'),nl,
        write('Inserire il primo insieme di numeri interi racchiuso tra
               parentesi quadre e terminato da punto:'),nl,
        read(I1), interi_non_duplicati(I1, AS),
        write('>> Primo insieme privo di eventuali elementi ripetuti: '),
        write(AS), nl,
        write('Inserire il secondo insieme di numeri interi racchiuso tra
               parentesi quadre e terminato da punto:'),nl,
        read(I2), interi_non_duplicati(I2, BS),
        write('>> Secondo insieme privo di eventuali elementi ripetuti: '),
        write(BS),nl,
        write('Il primo insieme e' sottoinsieme del secondo? '),
        risultato_sottoinsieme(AS, BS, R1), write(R1), nl,
        write('Il secondo insieme e' sottoinsieme del primo? '),
        risultato_sottoinsieme(BS, AS, R2), write(R2), nl,
        write('I due insiemi sono uguali? '),
        risultato_uguali(AS, BS, R3), write(R3), nl,
        write('Insieme delle parti del primo insieme: '),
        stampa_insieme_parti(AS), nl,
        write('Insieme delle parti del secondo insieme: '),
        stampa_insieme_parti(BS), nl,
        write('Somma dei numeri pari del primo insieme: '),
        somma_valori_pari(AS, R4), write(R4), nl,
        write('Somma dei numeri pari del secondo insieme: '),
        somma_valori_pari(BS, R5), write(R5), nl,
```

```
        write('Somma dei numeri dispari del primo insieme: '),
        somma_valori_dispari(AS, R6), write(R6), nl,
        write('Somma dei numeri dispari del secondo insieme: '),
        somma_valori_dispari(BS, R7), write(R7), nl.


/* Il predicato interi_non_duplicati rimuove i valori duplicati all'interno
di un insieme:
- il primo argomento e' l'insieme del quale si vogliono eliminare i valori
duplicati.
- il secondo argomento corrisponde al primo insieme privato dei duplicati.
L'uso del predicato rimuovi_elem  rimuove tutte le occorrenze di un elemento
da un insieme */

interi_non_duplicati([], []).
interi_non_duplicati([X | Xs], [X | L]) :- integer(X),
                                           rimuovi_elem(X, Xs, XsNoX),
                                           interi_non_duplicati(XsNoX, L).

/* Il predicato rimuovi_elem elimina tutte le occorrenze di un elemento da un
insieme:
- il primo argomento e' l'elemento da rimuovere.
- il secondo argomento e' l'insieme da cui rimuovere l'elemento specificato.
- il terzo argomento e' il risultato di tale operazione di rimozione. */

rimuovi_elem(_, [], []).
rimuovi_elem(X, [X | L], LNoX)       :- rimuovi_elem(X, L, LNoX).
rimuovi_elem(X, [Y | L], [Y | LNoX]) :- X \== Y,
                                        rimuovi_elem(X, L, LNoX).

/* Il predicato sottoinsieme verifica se un insieme e' sottoinsieme di un
altro insieme:
- il primo argomento e' il primo dei due insiemi;
- il secondo argomento e' il secondo dei due insiemi. */

sottoinsieme([], _).
sottoinsieme([X | Xs], Y) :- membro(X, Y),
                             sottoinsieme(Xs, Y).

/* il predicato risultato_sottoinsieme permette di stampare il risultato
del predicato sottoinsieme tramite l'utilizzo del predicato if_then_else:
- il primo argomento e' il primo dei due insiemi;
- il secondo argomento e' il secondo dei due insiemi.
```

```
- il terzo argomento e' il risultato. */

risultato_sottoinsieme(X, Y, R) :- if_then_else(sottoinsieme(X, Y),
                                                R is 1,
                                                R is 0).

/* il predicato if_then_else implementa il costrutto di selezione in
linguaggio prolog:
- il primo argomento e' la condizione dell'if.
- il secondo argomento e' il predicato da eseguire se la condizione e'
verificata.
- il terzo argomento e' il predicato da eseguire se la condizione non e'
verificata. */

if_then_else(I, T, _) :- I, !, T.
if_then_else(I, _, E) :- \+(I), !, E.

/* predicato per stabilire se un elemento appartiene ad una lista:
- Il primo argomento e' l'elemento di cui stabilire l'appartenenza.
- il secondo  argomento e' la lista in cui stabilire l'appartenenza. */

membro(X, [X | _]).
membro(X, [Y | Ys]) :- X \== Y,
                       membro(X, Ys).

/* Il predicato uguali verifica l'ugualianza tra due insiemi:
- il primo argomento e' il primo dei due insiemi;
- il secondo argomento e' il secondo dei due insiemi.
L'uso di sottoinsieme consente di verificare se gli insiemi sono
 rispettivamente l'uno il sottoinsieme dell'altro. */

uguali([], []).
uguali([X | Xs], [Y | Ys]) :- sottoinsieme([X | Xs], [Y | Ys]),
                               sottoinsieme([Y | Ys], [X | Xs]).

/* il predicato risultato_uguali permette di stampare il risultato
del predicato uguali tramite l'utilizzo del predicato if_then_else:
- il primo argomento e' il primo dei due insiemi;
- il secondo argomento e' il secondo dei due insiemi.
- il terzo argomento e' il risultato. */

risultato_uguali(X, Y, R) :- if_then_else(uguali(X, Y),
                                          R is 1,
```

```
                                        R is 0).

/* Il predicato insieme_delle_parti calcola l'insieme delle parti di
un insieme:
- il primo argomento e' l'insieme del quale si vuole calcolare l'insieme
  delle parti.
- il secondo argomento e' l'insieme delle parti. */

insieme_delle_parti([], []).
insieme_delle_parti([X | Xs], [X | PXs]) :- insieme_delle_parti(Xs, PXs).
insieme_delle_parti([_ | Xs], PXs)   :- insieme_delle_parti(Xs, PXs).


/* il predicato stampa_insieme_parti permette di stampare il risultato
del predicato insieme_delle_parti tramite l'utilizzo del predicato
setof:
- il primo e unico argomento e' l'insieme del quale si vuole calcolare
l'insieme delle parti. */

stampa_insieme_parti(X) :- setof(PX,
                                 insieme_delle_parti(X, PX),
                                 L),
                           write(L).

/* Il predicato pari verifica se un valore e' pari
- il primo e unico argomento e' il valore del quale si vuole verificare se sia
  pari. */

pari(X) :- Y is mod(X, 2),
           Y =:= 0.

/* Il predicato dispari verifica se un valore e' dispari
- il primo e unico argomento e' il valore del quale si vuole verificare se sia
  dispari
L'uso di pari viene utilizzato per verificare se il valore dispari preso in
considerazione sommato a uno risulta pari. */

dispari(X) :- Y is X + 1,
              pari(Y).

/* Il predicato somma_valori_pari calcola la somma dei valori pari all'interno
di un insieme:
- il primo argomento e' l'insieme di cui si vuole calcolare la somma dei numeri
```

```
  pari al suo interno;
- il secondo argomento e' la somma dei valori pari all'interno dell'insieme.
L'uso di pari consente di verificare se il termine del predicato e' pari.
L'uso di dispari consente di verificare se il termine del predicato e' dispari*/

somma_valori_pari([], 0).
somma_valori_pari([X | Xs], P) :- pari(X),
                                  somma_valori_pari(Xs, SP),
                                  P is X + SP.
somma_valori_pari([X | Xs], P) :- dispari(X),
                                  somma_valori_pari(Xs, P).

/* Il predicato somma_valori_dispari calcola la somma dei valori dispari
all'interno di un insieme:
- il primo argomento e' l'insieme di cui si vuole calcolare la somma
  dei numeri dispari al suo interno;
- il secondo argomento e' la somma dei valori dispari all'interno dell'insieme.
L'uso di dispari consente di verificare se il termine del predicato e' dispari.
L'uso di pari consente di verificare se il termine del predicato e' pari */

somma_valori_dispari([], 0).
somma_valori_dispari([X | Xs], D) :- dispari(X),
                                     somma_valori_dispari(Xs, SD),
                                     D is X + SD.
somma_valori_dispari([X | Xs], D) :- pari(X),
                                     somma_valori_dispari(Xs, D).
```

5 Testing del programma

Test Haskell 1

```
Programma Haskell per effettuare le seguenti operazioni su insiemi di numeri interi:
1) Dati due insiemi di numeri interi, verifica ricorsivamente se un insieme è sottoinsieme dell' altro.
2) Dati due insiemi di numeri interi, verifica se i due insiemi sono uguali.
3) Dato un insieme di numeri interi, calcola ricorsivamente l'insieme delle parti.
4) Dato un insieme di numeri interi, calcola ricorsivamente la somma dei numeri pari all'interno dell'insieme.
5) Dato un insieme di numeri interi, calcola ricorsivamente la somma dei numeri dispari all'interno dell'insieme.
Inserire il primo insieme di numeri interi racchiuso tra parentesi quadre:
[]
>> Primo insieme privo di eventuali elementi ripetuti: []
Inserire il secondo insieme di numeri interi racchiuso tra parentesi quadre:
[]
>> Secondo insieme privo di eventuali elementi ripetuti: []
1) Il primo insieme è sottoinsieme del secondo? True
1) Il secondo insieme è sottoinsieme del primo? True
2) I due insiemi sono uguali? True
3) Insieme delle parti del primo insieme: [[]]
3) Insieme delle parti del secondo insieme: [[]]
4) Somma dei numeri pari del primo insieme: 0
4) Somma dei numeri pari del secondo insieme: 0
5) Somma dei numeri dispari del primo insieme: 0
5) Somma dei numeri dispari del secondo insieme: 0
```

Test Haskell 2

```
Programma Haskell per effettuare le seguenti operazioni su insiemi di numeri interi:
1) Dati due insiemi di numeri interi, verifica ricorsivamente se un insieme è sottoinsieme dell' altro.
2) Dati due insiemi di numeri interi, verifica se i due insiemi sono uguali.
3) Dato un insieme di numeri interi, calcola ricorsivamente l'insieme delle parti.
4) Dato un insieme di numeri interi, calcola ricorsivamente la somma dei numeri pari all'interno dell'insieme.
5) Dato un insieme di numeri interi, calcola ricorsivamente la somma dei numeri dispari all'interno dell'insieme.
Inserire il primo insieme di numeri interi racchiuso tra parentesi quadre:
[1]
>> Primo insieme privo di eventuali elementi ripetuti: [1]
Inserire il secondo insieme di numeri interi racchiuso tra parentesi quadre:
[]
>> Secondo insieme privo di eventuali elementi ripetuti: []
1) Il primo insieme è sottoinsieme del secondo? False
1) Il secondo insieme è sottoinsieme del primo? True
2) I due insiemi sono uguali? False
3) Insieme delle parti del primo insieme: [[],[1]]
3) Insieme delle parti del secondo insieme: [[]]
4) Somma dei numeri pari del primo insieme: 0
4) Somma dei numeri pari del secondo insieme: 0
5) Somma dei numeri dispari del primo insieme: 1
5) Somma dei numeri dispari del secondo insieme: 0
```

Test Haskell 3

```
Programma Haskell per effettuare le seguenti operazioni su insiemi di numeri interi:
1) Dati due insiemi di numeri interi, verifica ricorsivamente se un insieme è sottoinsieme dell' altro.
2) Dati due insiemi di numeri interi, verifica se i due insiemi sono uguali.
3) Dato un insieme di numeri interi, calcola ricorsivamente l'insieme delle parti.
4) Dato un insieme di numeri interi, calcola ricorsivamente la somma dei numeri pari all'interno dell'insieme.
5) Dato un insieme di numeri interi, calcola ricorsivamente la somma dei numeri dispari all'interno dell'insieme.
Inserire il primo insieme di numeri interi racchiuso tra parentesi quadre:
[]
>> Primo insieme privo di eventuali elementi ripetuti: []
Inserire il secondo insieme di numeri interi racchiuso tra parentesi quadre:
[1]
>> Secondo insieme privo di eventuali elementi ripetuti: [1]
1) Il primo insieme è sottoinsieme del secondo? True
1) Il secondo insieme è sottoinsieme del primo? False
2) I due insiemi sono uguali? False
3) Insieme delle parti del primo insieme: [[]]
3) Insieme delle parti del secondo insieme: [[],[1]]
4) Somma dei numeri pari del primo insieme: 0
4) Somma dei numeri pari del secondo insieme: 0
5) Somma dei numeri dispari del primo insieme: 0
5) Somma dei numeri dispari del secondo insieme: 1
```

Test Haskell 4

```
Programma Haskell per effettuare le seguenti operazioni su insiemi di numeri interi:
1) Dati due insiemi di numeri interi, verifica ricorsivamente se un insieme è sottoinsieme dell' altro.
2) Dati due insiemi di numeri interi, verifica se i due insiemi sono uguali.
3) Dato un insieme di numeri interi, calcola ricorsivamente l'insieme delle parti.
4) Dato un insieme di numeri interi, calcola ricorsivamente la somma dei numeri pari all'interno dell'insieme.
5) Dato un insieme di numeri interi, calcola ricorsivamente la somma dei numeri dispari all'interno dell'insieme.
Inserire il primo insieme di numeri interi racchiuso tra parentesi quadre:
[1]
>> Primo insieme privo di eventuali elementi ripetuti: [1]
Inserire il secondo insieme di numeri interi racchiuso tra parentesi quadre:
[1]
>> Secondo insieme privo di eventuali elementi ripetuti: [1]
1) Il primo insieme è sottoinsieme del secondo? True
1) Il secondo insieme è sottoinsieme del primo? True
2) I due insiemi sono uguali? True
3) Insieme delle parti del primo insieme: [[],[1]]
3) Insieme delle parti del secondo insieme: [[],[1]]
4) Somma dei numeri pari del primo insieme: 0
4) Somma dei numeri pari del secondo insieme: 0
5) Somma dei numeri dispari del primo insieme: 1
5) Somma dei numeri dispari del secondo insieme: 1
```

Test Haskell 5

```
Programma Haskell per effettuare le seguenti operazioni su insiemi di numeri interi:
1) Dati due insiemi di numeri interi, verifica ricorsivamente se un insieme è sottoinsieme dell' altro.
2) Dati due insiemi di numeri interi, verifica se i due insiemi sono uguali.
3) Dato un insieme di numeri interi, calcola ricorsivamente l'insieme delle parti.
4) Dato un insieme di numeri interi, calcola ricorsivamente la somma dei numeri pari all'interno dell'insieme.
5) Dato un insieme di numeri interi, calcola ricorsivamente la somma dei numeri dispari all'interno dell'insieme.
Inserire il primo insieme di numeri interi racchiuso tra parentesi quadre:
[1]
>> Primo insieme privo di eventuali elementi ripetuti: [1]
Inserire il secondo insieme di numeri interi racchiuso tra parentesi quadre:
[2]
>> Secondo insieme privo di eventuali elementi ripetuti: [2]
1) Il primo insieme è sottoinsieme del secondo? False
1) Il secondo insieme è sottoinsieme del primo? False
2) I due insiemi sono uguali? False
3) Insieme delle parti del primo insieme: [[],[1]]
3) Insieme delle parti del secondo insieme: [[],[2]]
4) Somma dei numeri pari del primo insieme: 0
4) Somma dei numeri pari del secondo insieme: 2
5) Somma dei numeri dispari del primo insieme: 1
5) Somma dei numeri dispari del secondo insieme: 0
```

Test Haskell 6

```
Programma Haskell per effettuare le seguenti operazioni su insiemi di numeri interi:
1) Dati due insiemi di numeri interi, verifica ricorsivamente se un insieme è sottoinsieme dell' altro.
2) Dati due insiemi di numeri interi, verifica se i due insiemi sono uguali.
3) Dato un insieme di numeri interi, calcola ricorsivamente l'insieme delle parti.
4) Dato un insieme di numeri interi, calcola ricorsivamente la somma dei numeri pari all'interno dell'insieme.
5) Dato un insieme di numeri interi, calcola ricorsivamente la somma dei numeri dispari all'interno dell'insieme.
Inserire il primo insieme di numeri interi racchiuso tra parentesi quadre:
[1,2,3]
>> Primo insieme privo di eventuali elementi ripetuti: [1,2,3]
Inserire il secondo insieme di numeri interi racchiuso tra parentesi quadre:
[3,4,5]
>> Secondo insieme privo di eventuali elementi ripetuti: [3,4,5]
1) Il primo insieme è sottoinsieme del secondo? False
1) Il secondo insieme è sottoinsieme del primo? False
2) I due insiemi sono uguali? False
3) Insieme delle parti del primo insieme: [[],[1],[2],[1,2],[3],[1,3],[2,3],[1,2,3]]
3) Insieme delle parti del secondo insieme: [[],[3],[4],[3,4],[5],[3,5],[4,5],[3,4,5]]
4) Somma dei numeri pari del primo insieme: 2
4) Somma dei numeri pari del secondo insieme: 4
5) Somma dei numeri dispari del primo insieme: 4
5) Somma dei numeri dispari del secondo insieme: 8
```

Test Haskell 7

```
Programma Haskell per effettuare le seguenti operazioni su insiemi di numeri interi:
1) Dati due insiemi di numeri interi, verifica ricorsivamente se un insieme è sottoinsieme dell' altro.
2) Dati due insiemi di numeri interi, verifica se i due insiemi sono uguali.
3) Dato un insieme di numeri interi, calcola ricorsivamente l'insieme delle parti.
4) Dato un insieme di numeri interi, calcola ricorsivamente la somma dei numeri pari all'interno dell'insieme.
5) Dato un insieme di numeri interi, calcola ricorsivamente la somma dei numeri dispari all'interno dell'insieme.
Inserire il primo insieme di numeri interi racchiuso tra parentesi quadre:
[1,2,3]
>> Primo insieme privo di eventuali elementi ripetuti: [1,2,3]
Inserire il secondo insieme di numeri interi racchiuso tra parentesi quadre:
[3,2,1]
>> Secondo insieme privo di eventuali elementi ripetuti: [3,2,1]
1) Il primo insieme è sottoinsieme del secondo? True
1) Il secondo insieme è sottoinsieme del primo? True
2) I due insiemi sono uguali? True
3) Insieme delle parti del primo insieme: [[],[1],[2],[1,2],[3],[1,3],[2,3],[1,2,3]]
3) Insieme delle parti del secondo insieme: [[],[3],[2],[3,2],[1],[3,1],[2,1],[3,2,1]]
4) Somma dei numeri pari del primo insieme: 2
4) Somma dei numeri pari del secondo insieme: 2
5) Somma dei numeri dispari del primo insieme: 4
5) Somma dei numeri dispari del secondo insieme: 4
```

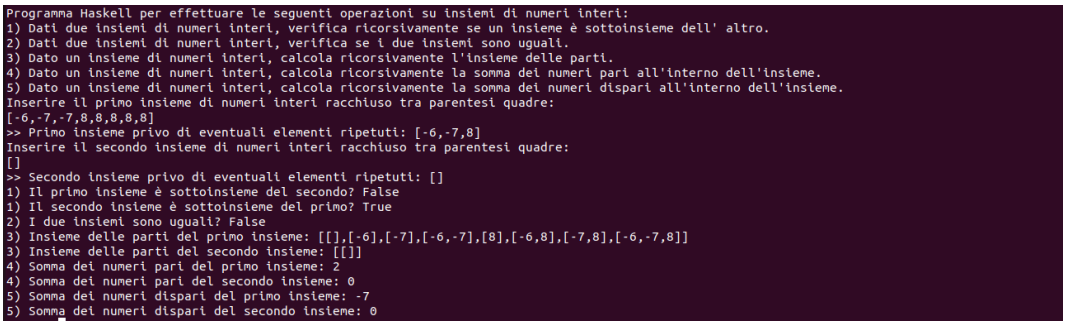
Test Haskell 8

```
1) Dati due insiemi di numeri interi, verifica ricorsivamente se un insieme è sottoinsieme dell' altro.
2) Dati due insiemi di numeri interi, verifica se i due insiemi sono uguali.
3) Dato un insieme di numeri interi, calcola ricorsivamente l'insieme delle parti.
4) Dato un insieme di numeri interi, calcola ricorsivamente la somma dei numeri pari all'interno dell'insieme.
5) Dato un insieme di numeri interi, calcola ricorsivamente la somma dei numeri dispari all'interno dell'insieme.
Inserire il primo insieme di numeri interi racchiuso tra parentesi quadre:
[-1,-2,-3]
>> Primo insieme privo di eventuali elementi ripetuti: [-1,-2,-3]
Inserire il secondo insieme di numeri interi racchiuso tra parentesi quadre:
[6,6,6,6,6,6,6,6,-5,-5,-5,-4,-5,-6,-4,-4]
>> Secondo insieme privo di eventuali elementi ripetuti: [6,-5,-4,-6]
1) Il primo insieme è sottoinsieme del secondo? False
1) Il secondo insieme è sottoinsieme del primo? False
2) I due insiemi sono uguali? False
3) Insieme delle parti del primo insieme: [[],[-1],[-2],[-1,-2],[-3],[-1,-3],[-2,-3],[-1,-2,-3]]
3) Insieme delle parti del secondo insieme: [[],[6],[-5],[6,-5],[-4],[6,-4],[-5,-4],[6,-5,-4],[-6],[6,-6],[-5,-6],[6,-5,-6],[-4,-6],[6,-4,-6],
[-5,-4,-6],[6,-5,-4,-6]]
4) Somma dei numeri pari del primo insieme: -2
4) Somma dei numeri pari del secondo insieme: -4
5) Somma dei numeri dispari del primo insieme: -4
5) Somma dei numeri dispari del secondo insieme: -5
```

Test Haskell 9

```
Programma Haskell per effettuare le seguenti operazioni su insiemi di numeri interi:
1) Dati due insiemi di numeri interi, verifica ricorsivamente se un insieme è sottoinsieme dell' altro.
2) Dati due insiemi di numeri interi, verifica se i due insiemi sono uguali.
3) Dato un insieme di numeri interi, calcola ricorsivamente l'insieme delle parti.
4) Dato un insieme di numeri interi, calcola ricorsivamente la somma dei numeri pari all'interno dell'insieme.
5) Dato un insieme di numeri interi, calcola ricorsivamente la somma dei numeri dispari all'interno dell'insieme.
Inserire il primo insieme di numeri interi racchiuso tra parentesi quadre:
[1,2,3]
>> Primo insieme privo di eventuali elementi ripetuti: [1,2,3]
Inserire il secondo insieme di numeri interi racchiuso tra parentesi quadre:
[2,3,0,0,0,0,0,-0]
>> Secondo insieme privo di eventuali elementi ripetuti: [2,3,0]
1) Il primo insieme è sottoinsieme del secondo? False
1) Il secondo insieme è sottoinsieme del primo? False
2) I due insiemi sono uguali? False
3) Insieme delle parti del primo insieme: [[],[1],[2],[1,2],[3],[1,3],[2,3],[1,2,3]]
3) Insieme delle parti del secondo insieme: [[],[2],[3],[2,3],[0],[2,0],[3,0],[2,3,0]]
4) Somma dei numeri pari del primo insieme: 2
4) Somma dei numeri pari del secondo insieme: 2
5) Somma dei numeri dispari del primo insieme: 4
5) Somma dei numeri dispari del secondo insieme: 3
```

Test Haskell 10



Test Prolog 1

```
Programma Prolog per effettuare le seguenti operazioni su insiemi di numeri interi:
1) Dati due insiemi di numeri interi, verifica ricorsivamente se un insieme è sottoinsieme dell altro.
2) Dati due insiemi di numeri interi, verifica se i due insiemi sono uguali.
3) Dato un insieme di numeri interi, calcola ricorsivamente l insieme delle parti.
4) Dato un insieme di numeri interi, calcola ricorsivamente la somma dei numeri pari all interno dell insieme.
5) Dato un insieme di numeri interi, calcola ricorsivamente la somma dei numeri dispari all interno dell insieme.
Inserire il primo insieme di numeri interi racchiuso tra parentesi quadre e terminato da punto:
[].
>> Primo insieme privo di eventuali elementi ripetuti: []
Inserire il secondo insieme di numeri interi racchiuso tra parentesi quadre e terminato da punto:
[].
>> Secondo insieme privo di eventuali elementi ripetuti: []
Il primo insieme è sottoinsieme del secondo? 1
Il secondo insieme è sottoinsieme del primo? 1
I due insiemi sono uguali? 1
Insieme delle parti del primo insieme: [[]]
Insieme delle parti del secondo insieme: [[]]
Somma dei numeri pari del primo insieme: 0
Somma dei numeri pari del secondo insieme: 0
Somma dei numeri dispari del primo insieme: 0
Somma dei numeri dispari del secondo insieme: 0

(1 ms) yes
```

Test Prolog 2

```
Programma Prolog per effettuare le seguenti operazioni su insiemi di numeri interi:
1) Dati due insiemi di numeri interi, verifica ricorsivamente se un insieme è sottoinsieme dell altro.
2) Dati due insiemi di numeri interi, verifica se i due insiemi sono uguali.
3) Dato un insieme di numeri interi, calcola ricorsivamente l insieme delle parti.
4) Dato un insieme di numeri interi, calcola ricorsivamente la somma dei numeri pari all interno dell insieme.
5) Dato un insieme di numeri interi, calcola ricorsivamente la somma dei numeri dispari all interno dell insieme.
Inserire il primo insieme di numeri interi racchiuso tra parentesi quadre e terminato da punto:
[1].
>> Primo insieme privo di eventuali elementi ripetuti: [1]
Inserire il secondo insieme di numeri interi racchiuso tra parentesi quadre e terminato da punto:
[].
>> Secondo insieme privo di eventuali elementi ripetuti: []
Il primo insieme è sottoinsieme del secondo? 0
Il secondo insieme è sottoinsieme del primo? 1
I due insiemi sono uguali? 0
Insieme delle parti del primo insieme: [[],[1]]
Insieme delle parti del secondo insieme: [[]]
Somma dei numeri pari del primo insieme: 0
Somma dei numeri pari del secondo insieme: 0
Somma dei numeri dispari del primo insieme: 1
Somma dei numeri dispari del secondo insieme: 0

true ?

(1 ms) yes
```

Test Prolog 3

```
Programma Prolog per effettuare le seguenti operazioni su insiemi di numeri interi:
1) Dati due insiemi di numeri interi, verifica ricorsivamente se un insieme è sottoinsieme dell altro.
2) Dati due insiemi di numeri interi, verifica se i due insiemi sono uguali.
3) Dato un insieme di numeri interi, calcola ricorsivamente l insieme delle parti.
4) Dato un insieme di numeri interi, calcola ricorsivamente la somma dei numeri pari all interno dell insieme.
5) Dato un insieme di numeri interi, calcola ricorsivamente la somma dei numeri dispari all interno dell insieme.
Inserire il primo insieme di numeri interi racchiuso tra parentesi quadre e terminato da punto:
[].
>> Primo insieme privo di eventuali elementi ripetuti: []
Inserire il secondo insieme di numeri interi racchiuso tra parentesi quadre e terminato da punto:
[1].
>> Secondo insieme privo di eventuali elementi ripetuti: [1]
Il primo insieme è sottoinsieme del secondo? 1
Il secondo insieme è sottoinsieme del primo? 0
I due insiemi sono uguali? 0
Insieme delle parti del primo insieme: [[]]
Insieme delle parti del secondo insieme: [[],[1]]
Somma dei numeri pari del primo insieme: 0
Somma dei numeri pari del secondo insieme: 0
Somma dei numeri dispari del primo insieme: 0
Somma dei numeri dispari del secondo insieme: 1

true ?

(1 ms) yes
```

Test Prolog 4

```
Programma Prolog per effettuare le seguenti operazioni su insiemi di numeri interi:
1) Dati due insiemi di numeri interi, verifica ricorsivamente se un insieme è sottoinsieme dell altro.
2) Dati due insiemi di numeri interi, verifica se i due insiemi sono uguali.
3) Dato un insieme di numeri interi, calcola ricorsivamente l insieme delle parti.
4) Dato un insieme di numeri interi, calcola ricorsivamente la somma dei numeri pari all interno dell insieme.
5) Dato un insieme di numeri interi, calcola ricorsivamente la somma dei numeri dispari all interno dell insieme.
Inserire il primo insieme di numeri interi racchiuso tra parentesi quadre e terminato da punto:
[1].
>> Primo insieme privo di eventuali elementi ripetuti: [1]
Inserire il secondo insieme di numeri interi racchiuso tra parentesi quadre e terminato da punto:
[1].
>> Secondo insieme privo di eventuali elementi ripetuti: [1]
Il primo insieme è sottoinsieme del secondo? 1
Il secondo insieme è sottoinsieme del primo? 1
I due insiemi sono uguali? 1
Insieme delle parti del primo insieme: [[],[1]]
Insieme delle parti del secondo insieme: [[],[1]]
Somma dei numeri pari del primo insieme: 0
Somma dei numeri pari del secondo insieme: 0
Somma dei numeri dispari del primo insieme: 1
Somma dei numeri dispari del secondo insieme: 1

true ?

yes
```

Test Prolog 5

```
Programma Prolog per effettuare le seguenti operazioni su insiemi di numeri interi:
1) Dati due insiemi di numeri interi, verifica ricorsivamente se un insieme è sottoinsieme dell altro.
2) Dati due insiemi di numeri interi, verifica se i due insiemi sono uguali.
3) Dato un insieme di numeri interi, calcola ricorsivamente l insieme delle parti.
4) Dato un insieme di numeri interi, calcola ricorsivamente la somma dei numeri pari all interno dell insieme.
5) Dato un insieme di numeri interi, calcola ricorsivamente la somma dei numeri dispari all interno dell insieme.
Inserire il primo insieme di numeri interi racchiuso tra parentesi quadre e terminato da punto:
[1].
>> Primo insieme privo di eventuali elementi ripetuti: [1]
Inserire il secondo insieme di numeri interi racchiuso tra parentesi quadre e terminato da punto:
[2].
>> Secondo insieme privo di eventuali elementi ripetuti: [2]
Il primo insieme è sottoinsieme del secondo? 0
Il secondo insieme è sottoinsieme del primo? 0
I due insiemi sono uguali? 0
Insieme delle parti del primo insieme: [[],[1]]
Insieme delle parti del secondo insieme: [[],[2]]
Somma dei numeri pari del primo insieme: 0
Somma dei numeri pari del secondo insieme: 2
Somma dei numeri dispari del primo insieme: 1
Somma dei numeri dispari del secondo insieme: 0

true ?

yes
```

Test Prolog 6

```
Programma Prolog per effettuare le seguenti operazioni su insiemi di numeri interi:
1) Dati due insiemi di numeri interi, verifica ricorsivamente se un insieme è sottoinsieme dell altro.
2) Dati due insiemi di numeri interi, verifica se i due insiemi sono uguali.
3) Dato un insieme di numeri interi, calcola ricorsivamente l insieme delle parti.
4) Dato un insieme di numeri interi, calcola ricorsivamente la somma dei numeri pari all interno dell insieme.
5) Dato un insieme di numeri interi, calcola ricorsivamente la somma dei numeri dispari all interno dell insieme.
Inserire il primo insieme di numeri interi racchiuso tra parentesi quadre e terminato da punto:
[1,2,3].
>> Primo insieme privo di eventuali elementi ripetuti: [1,2,3]
Inserire il secondo insieme di numeri interi racchiuso tra parentesi quadre e terminato da punto:
[3,4,5].
>> Secondo insieme privo di eventuali elementi ripetuti: [3,4,5]
Il primo insieme è sottoinsieme del secondo? 0
Il secondo insieme è sottoinsieme del primo? 0
I due insiemi sono uguali? 0
Insieme delle parti del primo insieme: [[],[1],[1,2],[1,2,3],[1,3],[2],[2,3],[3]]
Insieme delle parti del secondo insieme: [[],[3],[3,4],[3,4,5],[3,5],[4],[4,5],[5]]
Somma dei numeri pari del primo insieme: 2
Somma dei numeri pari del secondo insieme: 4
Somma dei numeri dispari del primo insieme: 4
Somma dei numeri dispari del secondo insieme: 8

true ?

yes
```

Test Prolog 7

```
Programma Prolog per effettuare le seguenti operazioni su insiemi di numeri interi:
1) Dati due insiemi di numeri interi, verifica ricorsivamente se un insieme è sottoinsieme dell altro.
2) Dati due insiemi di numeri interi, verifica se i due insiemi sono uguali.
3) Dato un insieme di numeri interi, calcola ricorsivamente l insieme delle parti.
4) Dato un insieme di numeri interi, calcola ricorsivamente la somma dei numeri pari all interno dell insieme.
5) Dato un insieme di numeri interi, calcola ricorsivamente la somma dei numeri dispari all interno dell insieme.
Inserire il primo insieme di numeri interi racchiuso tra parentesi quadre e terminato da punto:
[1,2,3].
>> Primo insieme privo di eventuali elementi ripetuti: [1,2,3]
Inserire il secondo insieme di numeri interi racchiuso tra parentesi quadre e terminato da punto:
[3,2,1].
>> Secondo insieme privo di eventuali elementi ripetuti: [3,2,1]
Il primo insieme è sottoinsieme del secondo? 1
Il secondo insieme è sottoinsieme del primo? 1
I due insiemi sono uguali? 1
Insieme delle parti del primo insieme: [[],[1],[1,2],[1,2,3],[1,3],[2],[2,3],[3]]
Insieme delle parti del secondo insieme: [[],[1],[2],[2,1],[3],[3,1],[3,2],[3,2,1]]
Somma dei numeri pari del primo insieme: 2
Somma dei numeri pari del secondo insieme: 2
Somma dei numeri dispari del primo insieme: 4
Somma dei numeri dispari del secondo insieme: 4

true ?

(2 ms) yes
```

Test Prolog 8

```
Programma Prolog per effettuare le seguenti operazioni su insiemi di numeri interi:
1) Dati due insiemi di numeri interi, verifica ricorsivamente se un insieme è sottoinsieme dell altro.
2) Dati due insiemi di numeri interi, verifica se i due insiemi sono uguali.
3) Dato un insieme di numeri interi, calcola ricorsivamente l insieme delle parti.
4) Dato un insieme di numeri interi, calcola ricorsivamente la somma dei numeri pari all interno dell insieme.
5) Dato un insieme di numeri interi, calcola ricorsivamente la somma dei numeri dispari all interno dell insieme.
Inserire il primo insieme di numeri interi racchiuso tra parentesi quadre e terminato da punto:
[-1,-2,-3,-3,-3,-2,-2,-2,-1,-1,-1].
>> Primo insieme privo di eventuali elementi ripetuti: [-1,-2,-3]
Inserire il secondo insieme di numeri interi racchiuso tra parentesi quadre e terminato da punto:
[-4,-6,1,1,1,1,1,1,1,-4].
>> Secondo insieme privo di eventuali elementi ripetuti: [-4,-6,1]
Il primo insieme è sottoinsieme del secondo? 0
Il secondo insieme è sottoinsieme del primo? 0
I due insiemi sono uguali? 0
Insieme delle parti del primo insieme: [[],[-3],[-2],[-2,-3],[-1],[-1,-3],[-1,-2],[-1,-2,-3]]
Insieme delle parti del secondo insieme: [[],[-6],[-6,1],[-4],[-4,-6],[-4,-6,1],[-4,1],[1]]
Somma dei numeri pari del primo insieme: -2
Somma dei numeri pari del secondo insieme: -10
Somma dei numeri dispari del primo insieme: -4
Somma dei numeri dispari del secondo insieme: 1

true ?

yes
```

Test Prolog 9

```
Programma Prolog per effettuare le seguenti operazioni su insiemi di numeri interi:
1) Dati due insiemi di numeri interi, verifica ricorsivamente se un insieme è sottoinsieme dell altro.
2) Dati due insiemi di numeri interi, verifica se i due insiemi sono uguali.
3) Dato un insieme di numeri interi, calcola ricorsivamente l insieme delle parti.
4) Dato un insieme di numeri interi, calcola ricorsivamente la somma dei numeri pari all interno dell insieme.
5) Dato un insieme di numeri interi, calcola ricorsivamente la somma dei numeri dispari all interno dell insieme.
Inserire il primo insieme di numeri interi racchiuso tra parentesi quadre e terminato da punto:
[1,2,3].
>> Primo insieme privo di eventuali elementi ripetuti: [1,2,3]
Inserire il secondo insieme di numeri interi racchiuso tra parentesi quadre e terminato da punto:
[].
>> Secondo insieme privo di eventuali elementi ripetuti: []
Il primo insieme è sottoinsieme del secondo? 0
Il secondo insieme è sottoinsieme del primo? 1
I due insiemi sono uguali? 0
Insieme delle parti del primo insieme: [[],[1],[1,2],[1,2,3],[1,3],[2],[2,3],[3]]
Insieme delle parti del secondo insieme: [[]]
Somma dei numeri pari del primo insieme: 2
Somma dei numeri pari del secondo insieme: 0
Somma dei numeri dispari del primo insieme: 4
Somma dei numeri dispari del secondo insieme: 0

true ?

yes
```

Test Prolog 10

```
Programma Prolog per effettuare le seguenti operazioni su insiemi di numeri interi:
1) Dati due insiemi di numeri interi, verifica ricorsivamente se un insieme è sottoinsieme dell altro.
2) Dati due insiemi di numeri interi, verifica se i due insiemi sono uguali.
3) Dato un insieme di numeri interi, calcola ricorsivamente l insieme delle parti.
4) Dato un insieme di numeri interi, calcola ricorsivamente la somma dei numeri pari all interno dell insieme.
5) Dato un insieme di numeri interi, calcola ricorsivamente la somma dei numeri dispari all interno dell insieme.
Inserire il primo insieme di numeri interi racchiuso tra parentesi quadre e terminato da punto:
[-6,-5,-4,-4,-4,1,1,1,1,1,1].
>> Primo insieme privo di eventuali elementi ripetuti: [-6,-5,-4,1]
Inserire il secondo insieme di numeri interi racchiuso tra parentesi quadre e terminato da punto:
[0,0,0,0,0,0,0,1,1,1,1,1,2,2,2,2,3,3,3,3,3,3].
>> Secondo insieme privo di eventuali elementi ripetuti: [0,1,2,3]
Il primo insieme è sottoinsieme del secondo? 0
Il secondo insieme è sottoinsieme del primo? 0
I due insiemi sono uguali? 0
Insieme delle parti del primo insieme: [[],[-6],[-6,-5],[-6,-5,-4],[-6,-5,-4,1],[-6,-5,1],[-6,-4],[-6,-4,1],[-6,1],[-5],[-5,-4],[-5,-4,1],[-5,
1],[-4],[-4,1],[1]]
Insieme delle parti del secondo insieme: [[],[0],[0,1],[0,1,2],[0,1,2,3],[0,1,3],[0,2],[0,2,3],[0,3],[1],[1,2],[1,2,3],[1,3],[2],[2,3],[3]]
Somma dei numeri pari del primo insieme: -10
Somma dei numeri pari del secondo insieme: 2
Somma dei numeri dispari del primo insieme: -4
Somma dei numeri dispari del secondo insieme: 4

true ?

(6 ms) yes
```