Dallo scritto d'esame del 16 sett. 2009:

Scrivere un metodo Java che, ricevendo come parametri un intero positivo P e un array A di interi, stabilisca se si possa ottenere il valore P sommando parte delle componenti dell'array A (eventualmente anche tutte o una sola, senza obbligo di contiguità).

Ad esempio, il responso per

$$A = \boxed{2}\boxed{0}\boxed{5}\boxed{7}\boxed{5}\boxed{-1}\boxed{10},$$

dovrà essere:

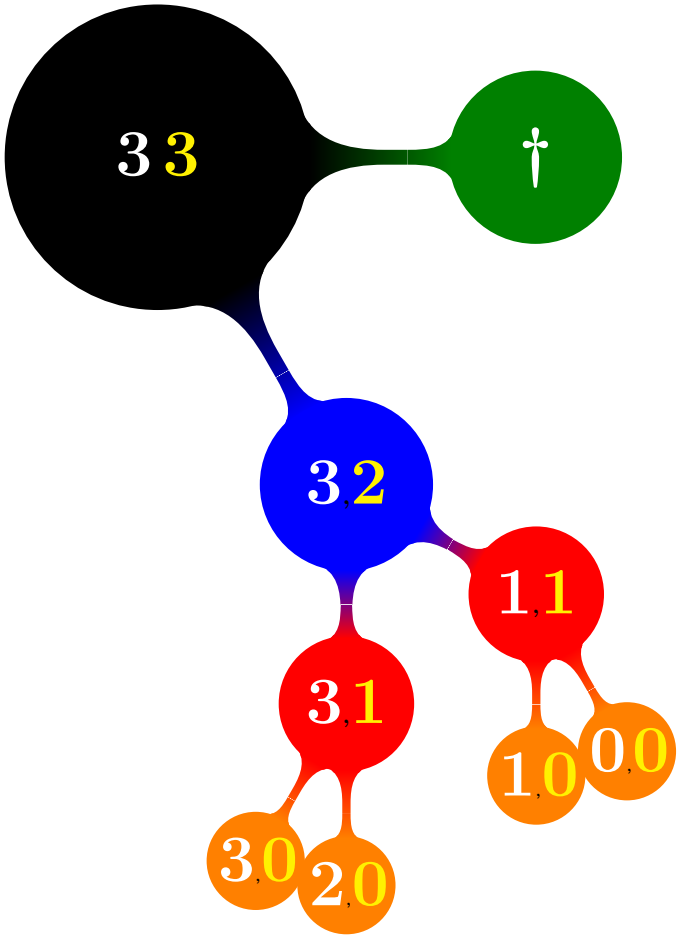
negativo se $P = 3$ oppure $P = 30$;

affermativo se $P = 2$, $P = 28$, oppure $P = 9$, in quanto ad es. $9 = 5+5+-1$.

Metodo risolutivo a responso booleano

```
public static boolean scomponInAddendi( int[] a, int p ){

    return scomponInAddendi( a, p, a.length );
}

private static boolean scomponInAddendi( int[] a, int p, int l ){

    if ( l == 0 ) return ( p == 0 );

    return scomponInAddendi( a, p, l-1 ) ||              // lascia
         ( a[ l-1 ] != 0 &&                              // pota
           scomponInAddendi( a, p - a[ l-1 ], l-1 ) ); // prendi
}
```

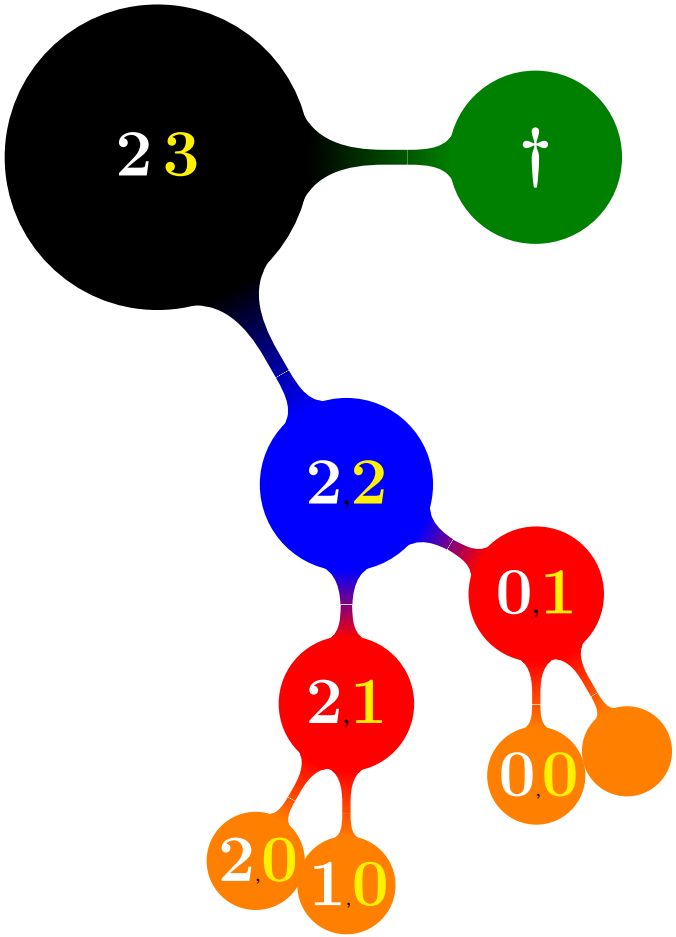
··· Seguono due esempi di esecuzione ···



array

1	2	0

goal $= 3$



array

1	2	0

goal = 2

Approfondimenti:

1. Come potare il ramo dell'albero di ricerca, non appena giunti a soluzione ?
2. Come restituire un *array* con gli addendi selezionati ?
3. Come restituire un *array*-soluzione quanto piú breve possibile ?
4. Come rappresentare e costruire la lista di tutti gli *array*-soluzione ?

Dallo scritto d'esame del 17 giugno 2009:

Una griglia di $N \times M$ caselle rappresenta un arcipelago. Le caselle contengono numeri, che dicono dove c'è mare (valore 0) e dove, invece, terra emersa (valore positivo). Tutt'intorno alla griglia, immaginiamo mare.

Un'isola è formata di caselle emerse collegate una all'altra in orizzontale e/o in verticale — intesa cosí, un'isola può essere circondata dal mare o toccarne altre in diagonale. Occorre individuare le isole e contarle.

· · · · · · segue esempio · · · · · ·

Dallo scritto del 17 giugno 2009

	0	1	2	3	4	5	6
0				•	•		
1			•		•		
2		•	•			•	
3			•	•	•	•	
4			•				
5					•		
6							•
7						•	•

Dallo scritto del 17 giugno 2009 Esempio: quattro isole.

	0	1	2	3	4	5	6
0				•	•		
1			•		•		
2		•	•			•	
3			•	•	•	•	
4			•				
5					•		
6							•
7						•	•

Dallo scritto del 17 giugno 2009 Esempio: quattro isole.

	0	1	2	3	4	5	6
0				•	•		
1			•		•		
2		•	•			•	
3			•	•	•	•	
4			•				
5					•		
6							•
7						•	•

	0	1	2	3	4	5	6
0				∗	∗		
1			⊙		∗		
2		⊙	⊙			⊙	
3			⊙	⊙	⊙	⊙	
4			⊙				
5					♣		
6							★
7						★	★

Dallo scritto del 17 giugno 2009 Richieste.

Scrivere metodi per:

1. riempire in modo casuale la griglia, rispettando (almeno approssimativamente) un rapporto indicato fra il numero di caselle emerse e il numero delle caselle di mare (ad esempio, una casella emersa ogni 4 di mare);

2. un metodo per costruire una copia di lavoro dell'arcipelago (cioè una seconda griglia inizialmente uguale alla prima, che potrà eventualmente subire modifiche durante lo svolgimento dei metodi che seguono);

3. un metodo che calcoli e stampi la superficie complessiva delle terre emerse (ogni casella conta per 2 Kmq);

4. un metodo che conti il numero totale di isole della laguna.

Per lo scritto del 17 giugno 2009: Idea per l'es. 1

```
/* Suggerimento di soluzione per l'esercizio 1 */

// Con il metodo che segue, ad esempio, per avere una
// casella emersa ogni 4 di mare, occorre porre k=0.2

public static short[][] riempi(short[][] A, double k)
{
    for (int i=0; i< A.length; i++)
        for (int j=0; j< A[i].length; j++)
            if (k<Math.random())
                A[i][j]=0;
            else
                A[i][j]=1;

    return A;
}
```

Per lo scritto del 17 giugno 2009: Idea per l'es. 4

```
/* Esercizio non richiesto: Individuare l'area dell'isola
(se esiste) di una cella di date coordinate  */

private static int celleIsola(short[][] A, int i, int j)
{
    if (!nellArcipelago(i,j,A))
        return 0;

    if (A[i][j]==0)
        return 0;

    A[i][j]=0;

    return 1+celleIsola(A,i,j+1)+celleIsola(A,i,j-1)+
             celleIsola(A,i+1,j)+celleIsola(A,i-1,j);
}

public static int areaIsola(short[][] A, int i, int j)
{
    return 2*celleIsola(clone(A),i,j);
}
```