Problem solving: dalla matematica all'informatica

Alberto Montresor (Università di Trento)

Milano, 27 Maggio, 2022

Problemi e problem solving

A **problem** is an *unknown entity* in some situation
(the difference between a goal state and a current state)

D. Jonassen. *Toward a Design Theory of Problem Solving*.
Educational Technology Research and Development, 48(4):63–85 (2000).

Problem solving is
“any goal-directed sequence of cognitive operations”

J.R. Anderson. *Cognitive Psychology and its implication.*
Freeman, New York, 1980

Problemi e problem solving

Esercizi vs **Problemi**

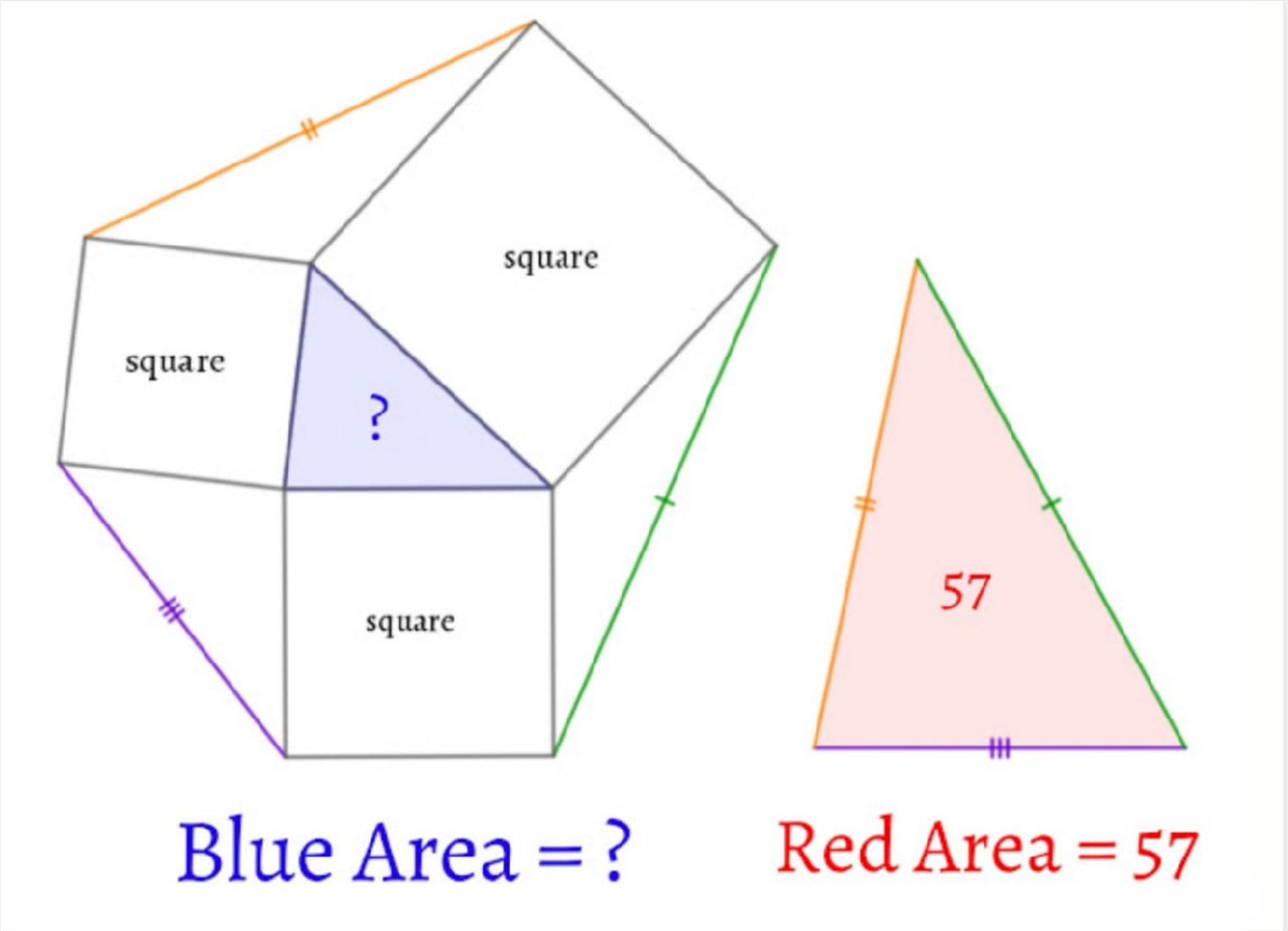
well-structured

ill-structured

applicazione di un
insieme ristretto di regole

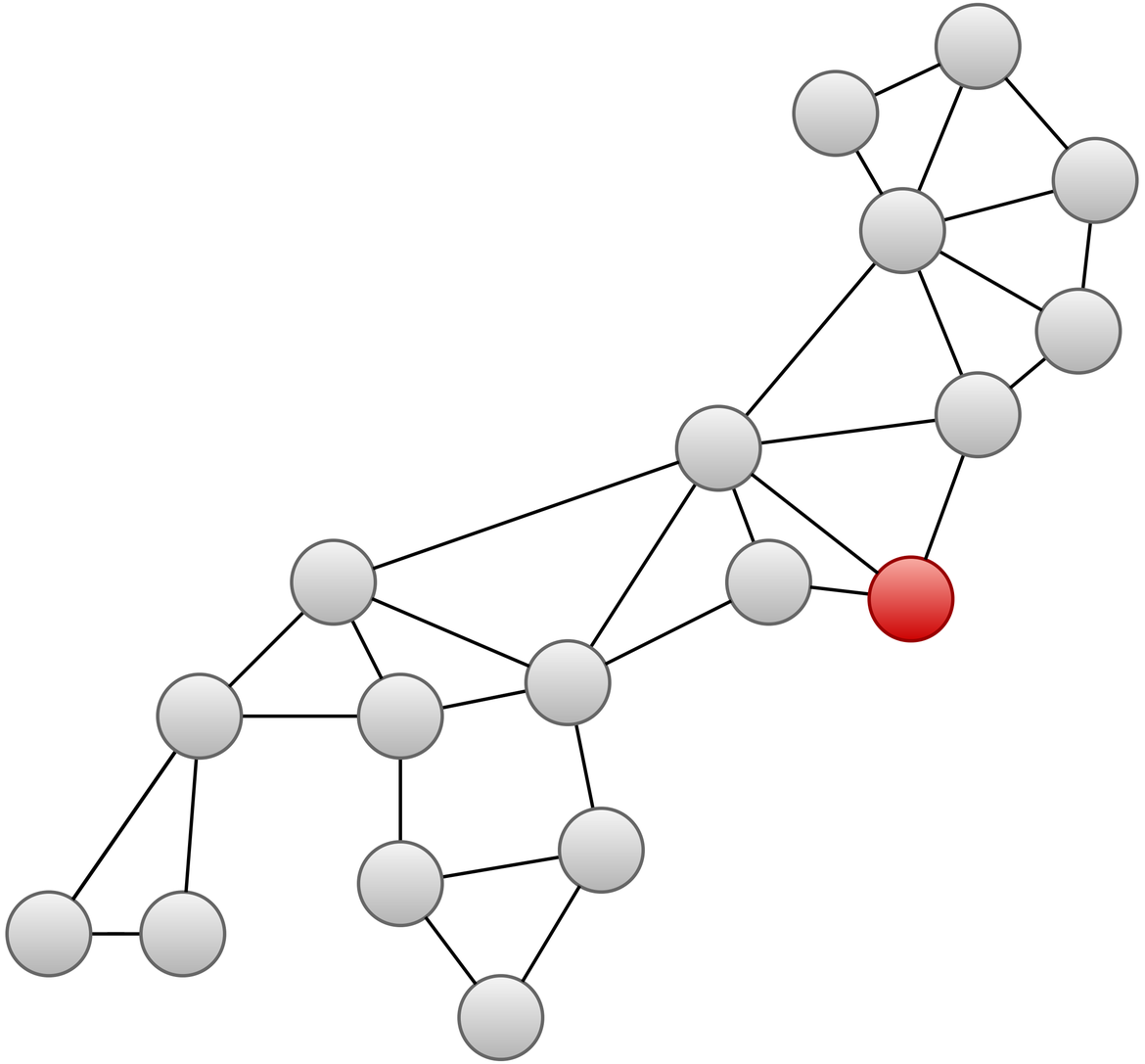
Possibili diversi
percorsi di soluzione

Risolvere problemi in matematica



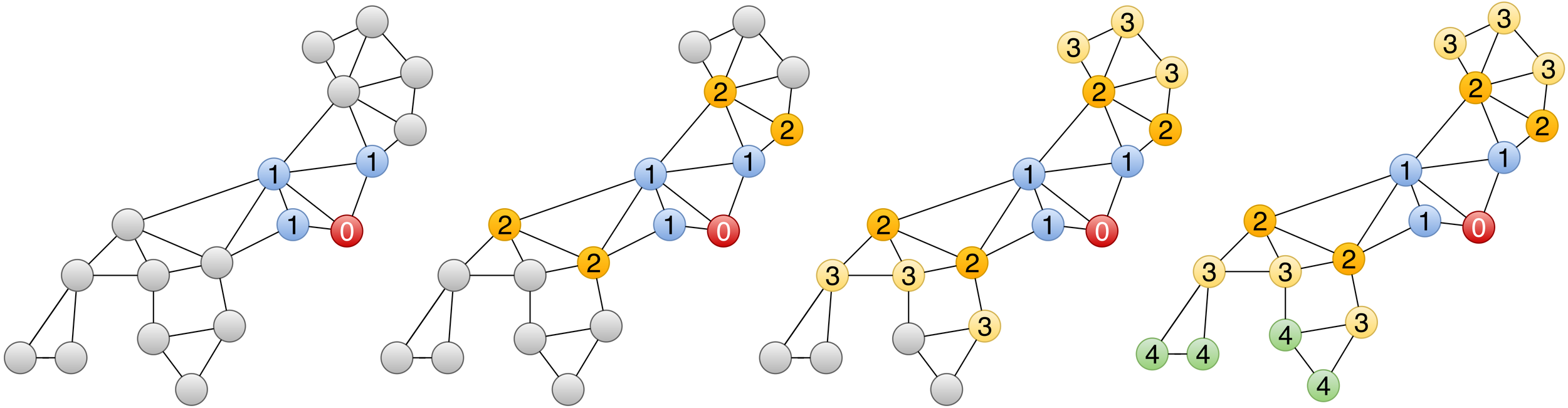
Segnali di fumo: Problema

- Comunicazione fra torri con segnali di fumo
- Due torri possono comunicare se sono visibili l'una con l'altra
- Accendere il fuoco richiede un minuto
- Quanto tempo per avvisare tutte le torri?



Fonte: Bebras

Segnali di fumo: Idea



1 minuto

2 minuti

3 minuti

4 minuti

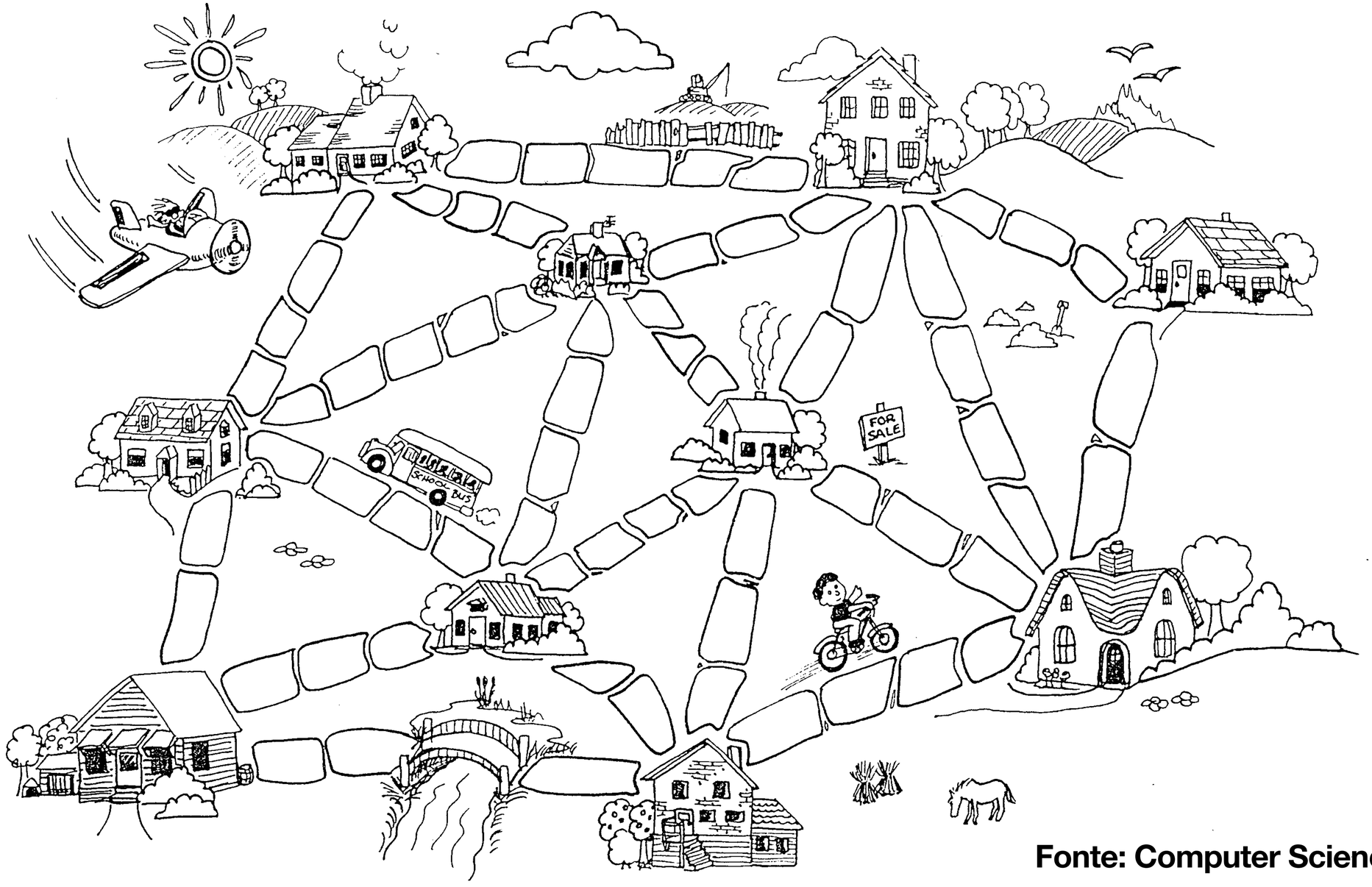
Fonte: Bebras

Segnali di fumo: Esprimere la soluzione

```
distance(Graph G, Node r, int[] distance)
Queue Q = Queue()
Q.enqueue(r)
foreach u ∈ G.V() − {r} do
    distance[u] = ∞
distance[r] = 0
while not Q.isEmpty() do
    Node u = Q.dequeue()
    foreach v ∈ G.adj(u) do
        if distance[v] == ∞ then % Se il nodo v non è stato scoperto
            distance[v] = distance[u] + 1
            Q.enqueue(v)
```

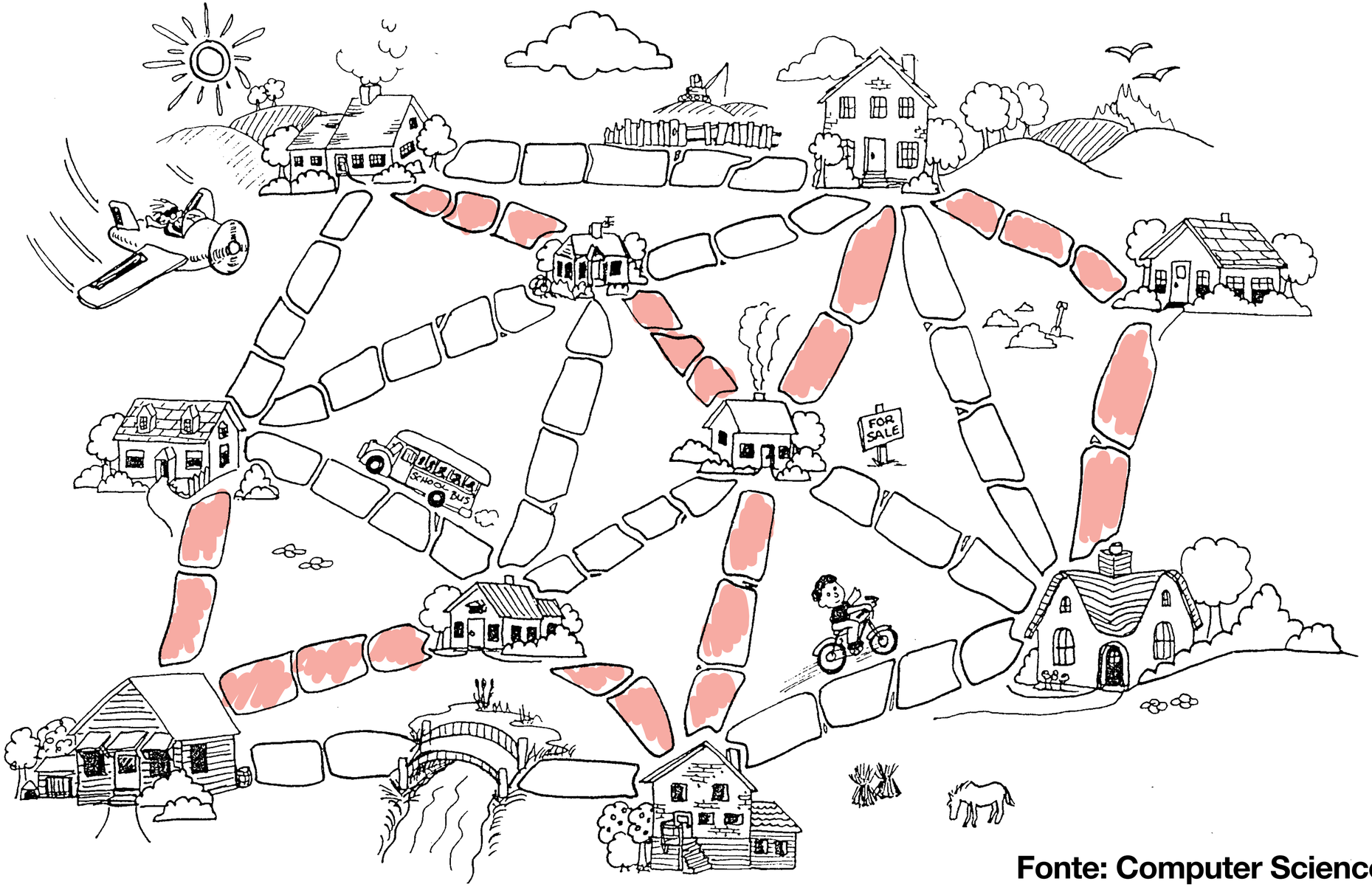
Visita in ampiezza

La città fangosa: Problema



Fonte: Computer Science Unplugged

La città fangosa: Idea

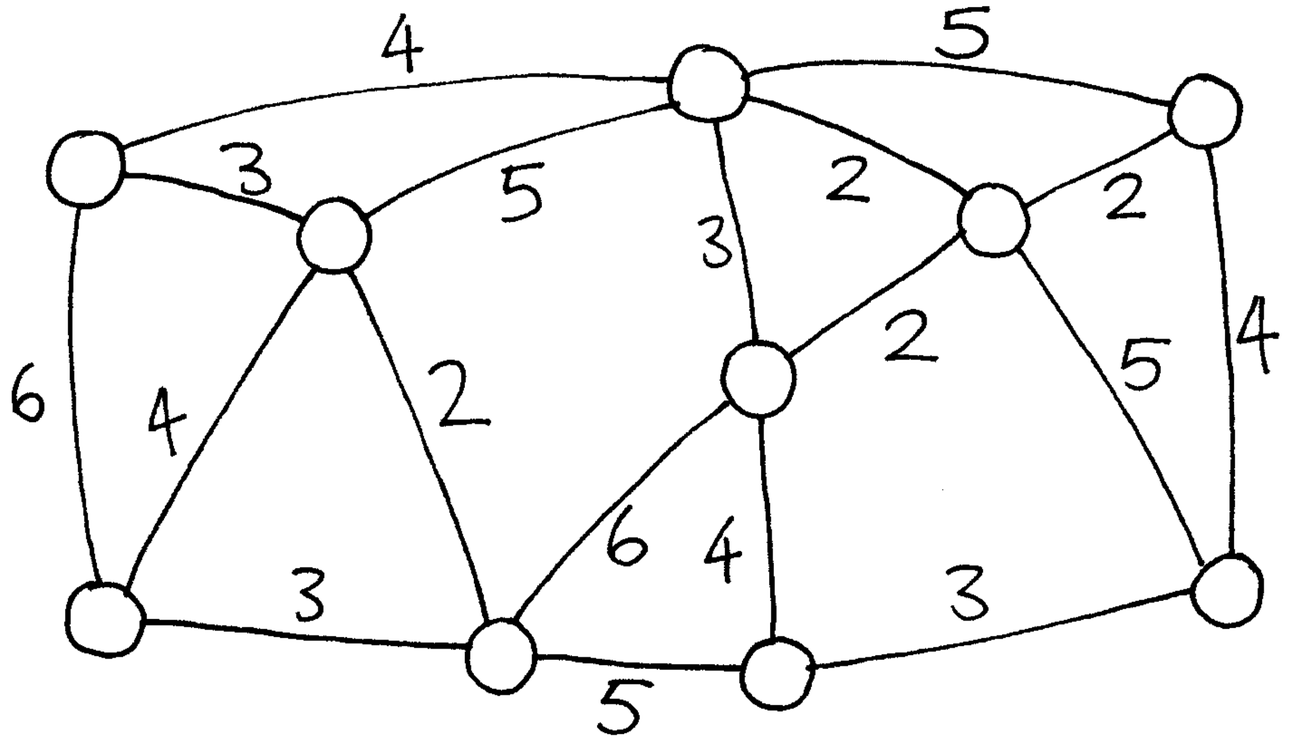


Fonte: Computer Science Unplugged

Città fangosa: generalizzazione

Albero di copertura di peso minimo

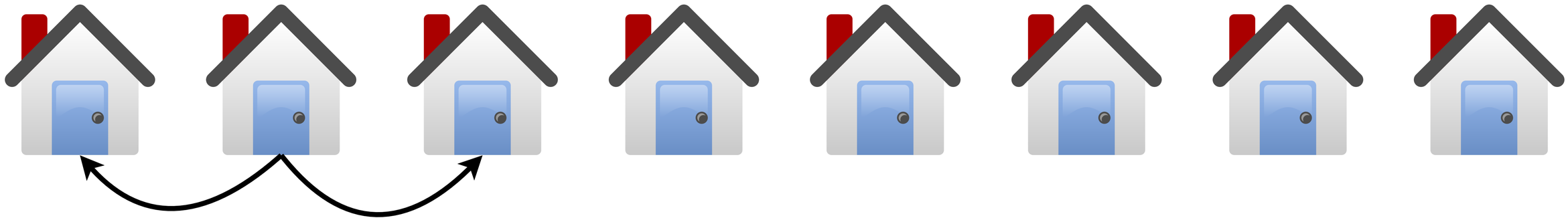
Algoritmo di Kruskal



Fonte: Computer Science Unplugged

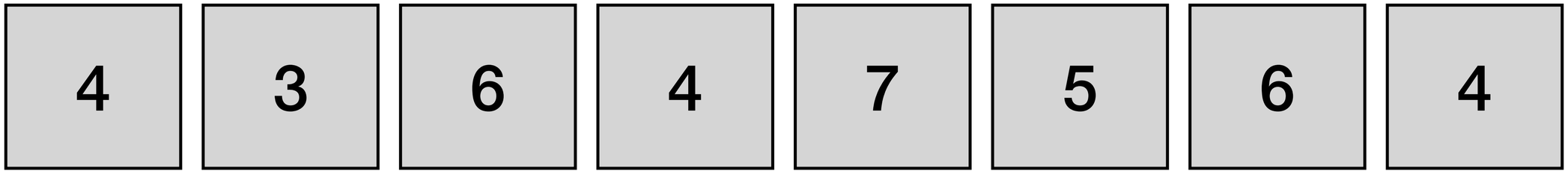
Hateville: Problema

- Hateville è un villaggio particolare, composto da *n* case, numerate da 1 a *n* lungo una singola strada.
- Ad Hateville ogni abitante *i*-esimo non sopporta i vicini *i-1* e *i+1*, da entrambi i lati (se esistenti)



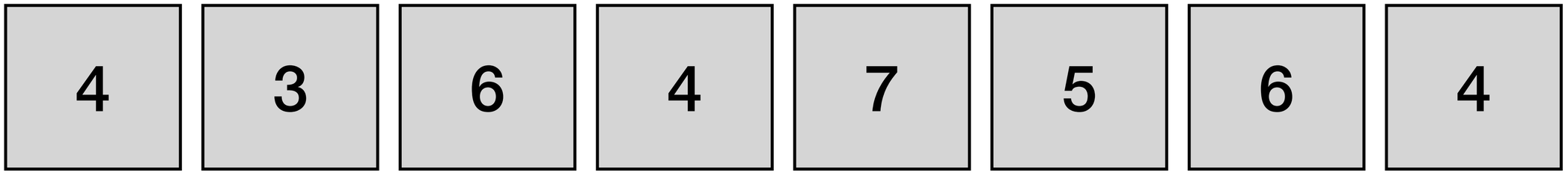
Hateville: Problema

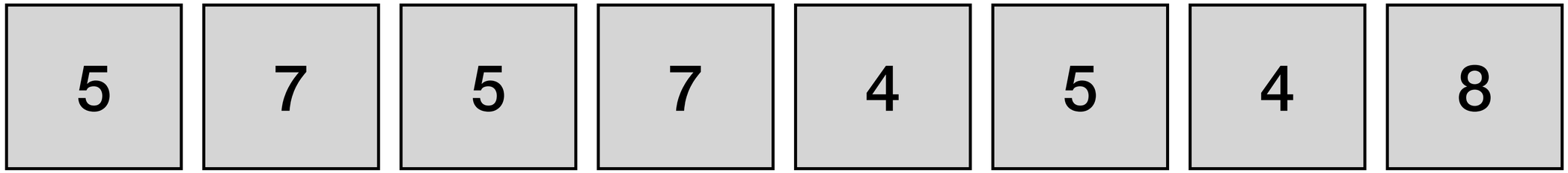
- Hateville vuole organizzare una sagra e vi ha affidato il compito di raccogliere i fondi.
- Ogni abitante *i* ha intenzione di donare una quantità D_i, ma non intende partecipare ad una raccolta fondi a cui partecipano uno o entrambi i propri vicini.



Hateville: Problema

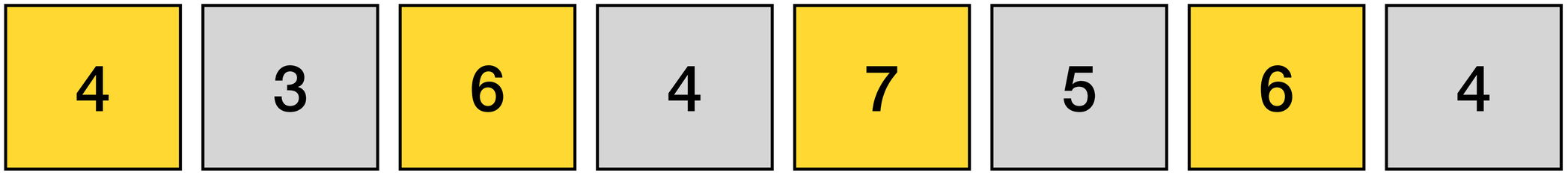
Qual è la somma massima ottenibile da questi due esempi?

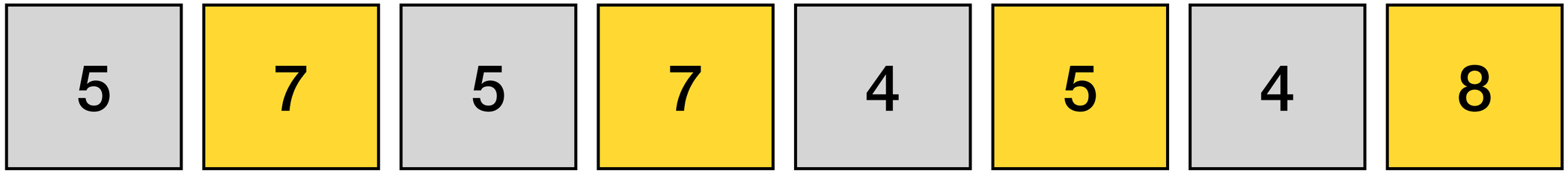




Hateville: Problema

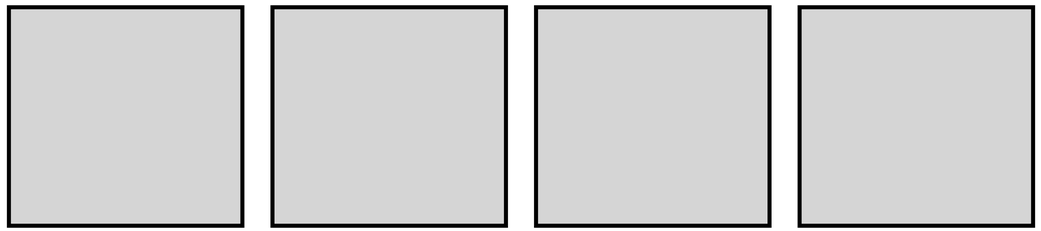
Qual è la somma massima ottenibile da questi due esempi?





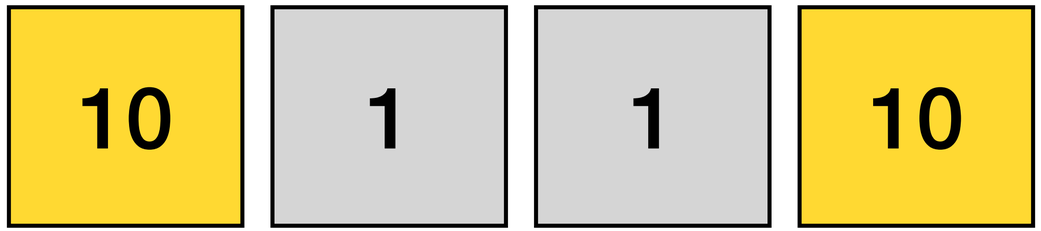
Hateville: Riflessione

Un esempio in cui non funziona?



Hateville: Riflessione

Un esempio in cui non funziona?



Come fare quindi?

Hateville: Soluzione

Sia S_i la somma massima che si può ottenere dalle prime *i* case

$$S_i = \begin{cases} 0 & i = 0 \\ D_1 & i = 1 \\ \max\{S_{i-1}, S_{i-2} + D_i\} & i > 1 \end{cases}$$

D_i	4	3	6	4	3	8	6	8
S_i								

Hateville: Soluzione

Sia S_i la somma massima che si può ottenere dalle prime *i* case

$$S_i = \begin{cases} 0 & i = 0 \\ D_1 & i = 1 \\ \max\{S_{i-1}, S_{i-2} + D_i\} & i > 1 \end{cases}$$

D_i	4	3	6	4	3	8	6	8
S_i	0	4	4	10	10	13	18	19	26

Hateville: Soluzione

Sia S_i la somma massima che si può ottenere dalle prime *i* case

$$S_i = \begin{cases} 0 & i = 0 \\ D_1 & i = 1 \\ \max\{S_{i-1}, S_{i-2} + D_i\} & i > 1 \end{cases}$$

D_i		4	3	6	4	3	8	6	8
S_i	0	4	4	10	10	13	18	19	26

Hateville: Esprimere la soluzione

```
1    def hateville(D):
2        S = [ 0, D[0] ]
3        for i in range(1, len(D)):
4            S.append(max(S[-1], S[-2] + D[i]))
5        return D[-1]
```

Pensiero computazionale

Il pensiero computazionale è il processo mentale coinvolto nella formulazione di un problema e nell'espressione delle sue soluzioni, in modo che possano essere effettivamente portate a termine da un esecutore – umano o artificiale.

Wing, Jeannette. "*Computational Thinking Benefits Society*".
40th Anniversary Blog of Social Issues in Computing. (2014)

Pensiero computazionale

Il pensiero computazionale è un modo di pensare, o processo mentale, per un particolare tipo di problem solving: quello che coinvolge un agente esterno per automatizzare l'esecuzione della soluzione

M. Lodi and S. Martini. Computational thinking, between Papert and Wing. Sci & Educ 30, 883–908 (2021).

Dal pensiero computazionale all’informatica

L’informatica costituisce l’ambiente ideale dove impostare un apprendimento basato sui problemi, che può concretizzare l’astratto, spostando il confine fra formale e concreto

Quadro di riferimento CISIA “Logica, ragionamento e problemi”

Osserviamo che per risolvere problemi è necessario avere atteggiamenti, convinzioni ed emozioni favorevoli.

Occorre che gli studenti imparino ad apprezzare il piacere intellettuale del problem solving e a controllare l’ansia che inevitabilmente si prova nelle situazioni nuove e incerte.

Riusciranno meglio a insistere con pazienza nel lavoro che occorre per risolvere un problema, anche se i primi sforzi dovessero risultare infruttuosi.

Quadro di riferimento CISIA “Logica, ragionamento e problemi”

A questi fini è utile che i docenti organizzino attività e discussioni di gruppo, lascino tempo per lavorare con calma, evitino di associare gli eventuali insuccessi nel problem solving a valutazioni scolastiche negative.

Così gli studenti potranno sviluppare la capacità di affrontare un problema con la tranquillità emotiva, la fiducia e la determinazione necessarie, nonché la disponibilità e l’interesse di cercare da sé problemi da risolvere.