*Greedy santa*

Corso: Algoritmi e Strutture Dati

Docente: Matteo Salani

Gruppo: Jari Näser, Mattia Toscanelli,

Matteo Cadoni, Antonio Fetta

Data: 26.12.2021

Descrizione del problema

Babbo Natale ama tutto del suo lavoro: la slitta, le renne, il clima freddo, i sorrisi dei bambini quando ricevono i loro regali... tutto tranne una cosa: odia davvero i suoi manager. È un dato di fatto, la North Pole Holdings LLC è una società molto esigente e i suoi alti dirigenti sono arroganti e difficili da accontentare. Ogni anno Babbo Natale fa del suo meglio per spendere il meno possibile per comprare i regali, essere il più veloce possibile per consegnarli in tempo, eppure c'è sempre qualche manager che si lamenta! Ecco perché, quest'anno, Babbo Natale si sente un po' "vendicativo": comprerà regali molto costosi (ad esempio, sta pensando di comprare una vacanza alle Hawaii come regalo per se stesso) per sperperare tutto il budget fornito dall'azienda, un totale di B euro.

Babbo Natale vuole usare tutto il budget dell'azienda anche se ciò significa dover pagare qualcosa di tasca propria! In altre parole, vuole scegliere un sottoinsieme dei regali disponibili che massimizzi i soldi spesi dal budget riducendo al minimo la quantità di denaro che deve spendere di tasca sua. Aiuta Babbo Natale calcolando il costo totale dei regali!

Input

La prima riga contiene due interi N e B, rispettivamente il numero di regali disponibili e il budget stabilito dai manager di Babbo Natale. La seconda riga contiene N interi , dove l'i-esimo intero rappresenta il costo dell'i-esimo dono.

Output

### È necessario scrivere una singola riga con un numero intero: il costo totale di alcuni sottoinsiemi dei regali in modo tale che il budget venga utilizzato il più possibile riducendo al minimo ciò che Babbo Natale dovrà pagare da solo.

### Assunzioni

Le assunzioni di questo progetto sono:

* Il budget è sempre sufficiente per comperare almeno uno dei regali

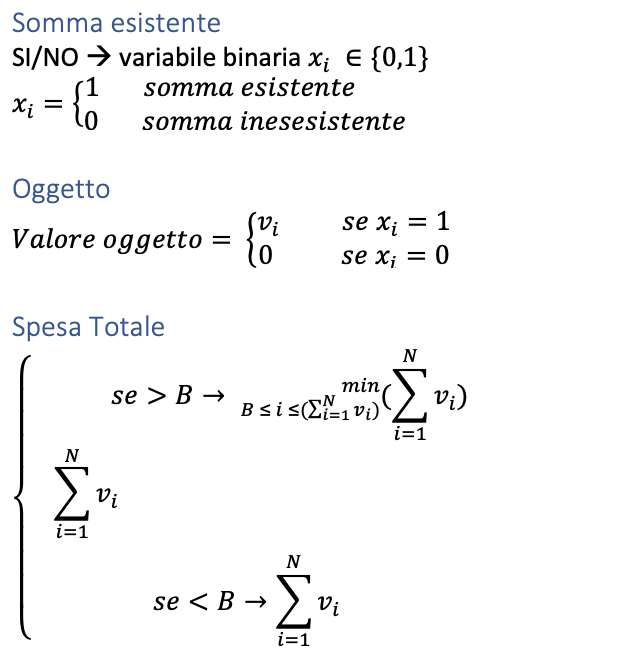
### 

### Assegnazione del punteggio

La valutazione del programma viene eseguito con 8 subtasks:

* Subtask 1 [0 punti]: Casi d’esempio
* Subtask 2 [30 punti]: Il costo di tutti i regali è di 1 o 2
* Subtask 3 [50 punti]:
* Subtask 4 [20 punti]: Nessuna limitazione aggiuntiva

## Formalizzazione



## Descrizione dell’algoritmo tramite pseudo-codice

### Versione non ricorsiva

//Metodo per calcolare il totale la spesa totale di babbo natale

function greedy\_santa()

//Array utilizzato per calcolare tutte le somme possibili.

//L'indice dell'array sono le somme e viene impostato a 1 a tutte le somme ammissibili.

var sums[MAXN] = fillArrayWithNumber(0);

//Variabile di supporto per cacolare la somma di tutti i regali.

int sum = 0;

//Variabile di ritorno della somma spesa da babbo natale.

int spesa = MAXN;

//Ciclo tutti i regali

for j = 0 to j < N do

//Incremento la somma totale dei regali.

sum += V[j];

//Eseguo un for che parte da Budget e arriva a 0.

//Parto da budget perché è inutile sommare altri regali già al di fuori del budget.

for i = B to i >= 0 do

//Controllo se è gia presente una somma calcolata precedentemente al di sotto

// del budget.

if sums[i] == 1 then

//Se è presente gia una somma aggiungo all'indice il valore dell'ultimo

// regalo 1.

sums[i + V[j]] = 1;

//Se trovo una somma che è uguale al budget è inutile continuare.

if i + V[j] == B then

return i + V[j];

//Se è una somma accettabile salvo il valore minimo sopra il budget.

else if i + V[j] > B then

spesa = min(spesa, i + V[j]);

end if

//Se il valore singolo del regalo non è ancora presente lo aggiungo all'array

// di somme.

else if sums[i] == 0 && i == V[j] then

sums[i] = 1;

end if

end for

end for

//Se la somma totale di tutti i regali non supera il budget, imposto questa come

// spesa di babbo natale.

if spesa == MAXN then

spesa = sum;

end if

return spesa;

end function

### Versione ricorsiva

// Dati in input

var N, B, V[N];

// Metodo per calcolare il totale la spesa totale di babbo natale

function greedySanta(var i, var sum){

// Se la somma del singolo regalo e quella già esistente è maggiore o uguale

// al budget essa viene ritornata

if V[i] + sum >= B then

return V[i] + sum;

end if

// Se arrive all’ultimo regalo ritorno la somma visto che è la combinazione

// più vicina al budget

if i == N then

return sommaTotale;

end if

// Se la somma totale è più piccola del budget oppure non sono ancora arrivato

// all'ultimo regalo richiamo ricorsivamente la funzione aggiornando i parametri.

return min(greedySanta(i+1, sum),greedySanta(i+1, sum+V[i]));

end function

## Analisi di complessità dell’algoritmo nel caso pessimo

### Metodo non ricorsivo

### A picture containing chart Description automatically generated

Operazioni atomiche

Complessità: B

Complessità: N

### 

Complessità soluzione non ricorsiva:

### Metodo ricorsivo

Chart

Description automatically generated

Operazioni atomiche

Complessità:

Complessità soluzione non ricorsiva:

## Screenshot della piattaforma di valutazione con i test passati o falliti

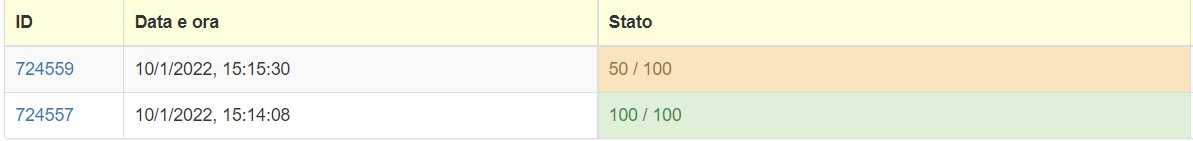
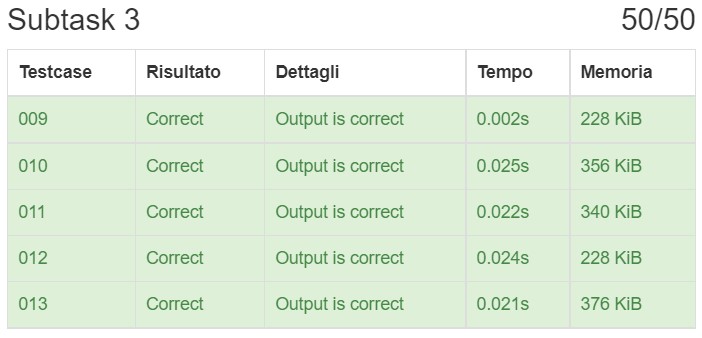


Figura 1 Il primo test è la versione ricorsiva, il secondo è la versione iterativa (Bottom-Up)

## Versione ricorsiva

Immagine che contiene tavolo

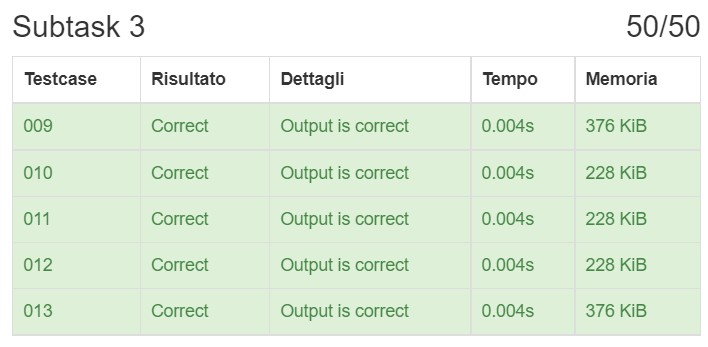
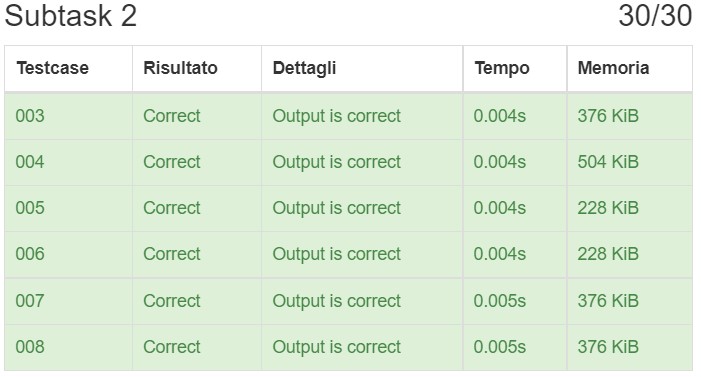
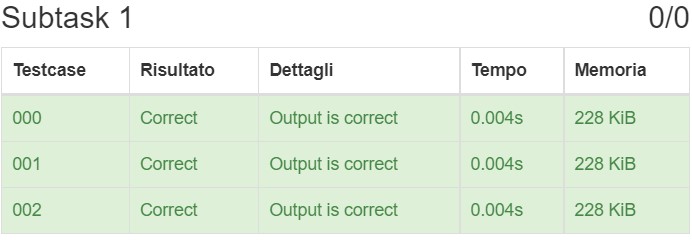
Descrizione generata automaticamenteImmagine che contiene tavolo

Descrizione generata automaticamenteTable

Description automatically generated

Figura Risultati dei test versione ricorsiva

## Bottom-Up

Table

Description automatically generated

Figura Risultati dei test versione bottom-up

La prima versione di codice, cioè la versione ricorsiva,

Descrizione delle procedure di test effettuate

Per il collaudo del sorgente sono stati utilizzati i test forniti dalla piattaforma. Durante lo sviluppo sono stati effettuati piccoli test in locale, attraverso la creazione di file di input appositi.

## Lista di problemi noti

Non sono stati riscontrati problemi durante lo sviluppo della soluzione

## Sitografia:

<https://training.olinfo.it/#/task/ois_christmas/statement>, 26.12.2021, Greedy Santa