

Nome: _____ Cognome: _____ Matr: _____ Postazione _____

Scrivere un programma in linguaggio C (chiamare il progetto con la propria <matricola>) che abbia il comportamento descritto nel seguito. Il tempo a disposizione è di **120 minuti**. Al termine del tempo gli elaborati salvati su **U:** verranno raccolti automaticamente dal sistema.

Eventuali documenti sono disponibili in **T:\Bertozzi**, si consiglia di usare **wordpad** per leggere i file di testo.

Il “Problema dello Zaino” è un problema di ottimizzazione. Si immagini di dover riempire uno zaino di volume **B** con oggetti scelti da un insieme di **k** oggetti, ciascuno dei quali con volume **a₁**, **a₂**, ..., **a_k** tali che $\sum_i a_i > B$. Chiaramente non si potranno infilare nello zaino tutti i **k** oggetti ma solo alcune combinazioni di questi. Si vuole trovare quale o quali combinazioni di oggetti riempie lo zaino il più possibile. Non esiste purtroppo un algoritmo efficiente per risolvere questo problema che, nella sua forma generale, richiede 2^k tentativi (problema NP-completo).

L'unica maniera efficiente di riempimento la si può ottenere se i **k** oggetti formano una sequenza “supercrescente” (definita nel seguito). In tal caso l'approccio possibile è quello “mettere” mano mano nello zaino gli elementi più grandi che ci stanno (*greedy algorithm*) fino ad esaurimento spazio o fino ad aver tentato di metterli tutti.

Ad esempio: consideriamo di avere 5 oggetti con volume 25, 51, 95, 190 e 384 e uno zaino di 555 (cm³). Inizialmente proveremo con l'oggetto di maggior volume (384), che ci sta. Si passa poi a considerare l'oggetto con volume 190 cm³ che però non è possibile inserire visto che rimangono solo 555-384=171 cm³. Quindi si prende in esame l'oggetto con volume pari a 95 cm³ che è possibile inserire e così via... Sviluppare un programma che:

1. Prende in ingresso il nome di un file binario (<numero>.dat) che contiene: nei primi 4 byte (un int) il volume dello zaino da riempire, nei secondi 4 byte il numero *n* di oggetti e, nei restanti *n**4 byte, il volume degli *n* oggetti ordinati per volume dal più piccolo. Legge e memorizza in opportuna struttura dati il contenuto di tale file
Stampare ai fini di debug i dati letti.
2. Verifica che la sequenza letta sia supercrescente, ovvero che ciascun numero nella sequenza ordinata sia maggiore della somma dei numeri precedenti. Il programma termina nel caso che questa condizione non sia soddisfatta
3. Usa il “*greedy algorithm*” discusso sopra per calcolare quali oggetti massimizzano l'uso dello zaino, stampa i loro volumi e stampa la percentuale di utilizzo dello zaino stesso.
4. Modificare il punto 1 di modo da poter gestire anche i file (u<numero>.dat) che contengono i volumi degli *n* oggetti non ordinati.

È obbligatorio usare, dove sensato, allocazione dinamica della memoria. La definizione e l'uso di funzioni per i primi 3 punti permette di ottenere un punteggio aggiuntivo.

Il codice va sviluppato nell'ordine indicato. Per ogni punto implementato si verifichi il corretto funzionamento. La correzione termina al primo punto non implementato correttamente.