Esercizi Blocco 2

Luca Oliveri
luca.olivieri-1@unitn.it

Università di Trento — October 31, 2020

Introduzione

Molti di questi esercizi richiedono aver affrontato l'argomento array, ma non per forza tutti. Quando si chiede di estendere le funzionalità di un altro programma si consiglia di non modificare quello vecchio, ma di iniziarne uno nuovo. Tenere a mente che abbiamo fatto le funzioni, quindi usarle quando ci sembra appropriato.

2.1

SUP che legge 10 numeri dall'utente e ne calcola la somma. Fermarsi a leggere numeri quando l'utente inserisce tutti i 10 numeri oppure quando inserisce 0, stabilendo un numero massimo a priori.

Info: Il limite massimo dipende da quanto grandi decidiamo che siano le strutture in fase di compilazione. Impareremo che è possibile scrivere programmi che non hanno questo tipo di limite.

2.2

Riscrivere l'esercizio 1.2 usando gli array, con lunghezza in cifre del numero variabile.

2.3

Modificare il programma 1.2 perché l'utente possa inserire frasi anziché numeri, lunghe fino a 100 caratteri.

Info: Una lista con cui fare dei test, tenendo a mente che la stessa lettera maiuscola e minuscola ha due codici ASCII diversi. Inoltre in queste frasi la punteggiatura non è palindroma. Usare questo e questo per modificare la stringa inserita dall'utente rendendo tutte le lettere minuscole e togliere la punteggiatura.

2.4

SUP che legge diversi numeri decimali e successivamente ristampa questi approssimati all'intero più vicino. Usare la funzione di libreria `round`. Stampare i numeri come interi, quindi non `42.0` ma `42`.

2.5

SUP che calcola l'ipotenusa dati i due cateti.

2.6

SUP un programma che usa due funzioni di conversione della temperatura, da Celsius a Fahrenheit e viceversa. Il programma stampa una tabella ben formattata con entrambe le scale da -40 a +250 gradi Celsius, a intervalli di 0.25 gradi.

2.7

SUP che calcola la media dei voti di un esame. Il programma si fa dare prima il numero totale di studenti, poi i voti. Supporre un limite massimo di studenti.

2.8

Trovare il numero più grande in un array è un problema ricorrente in programmazione. SUP che dati 10 numeri, inseriti dall'utente, restituisce il numero più grande.

2.9

Modificare il programma precedente in modo che trovi i due numeri più grandi.

2.10

Modificare il programma dei numeri primi cambiando il limite superiore della ricerca di divisori perfetti con la radice del numero sotto esame, anziché la sua metà. Confrontare l'output del nuovo algoritmo con quello vecchio.

Info: Provare a cercare online come si fa a temporizzare l'esecuzione di un programma e stimare il guadagno in velocità con questa modifica.

2.11

Un intero è detto perfetto quando la somma dei suoi fattori restituisce il numero stesso. Ad esempio 6 è numero perfetto essendo che $1+2+3=6$. Scrivere una funzione che determina se un numero è perfetto. Usarla per ricavare tutti i numeri perfetti da 1 a 1000.

Info: Se siete riusciti a temporizzare il programma precedente, provate ad alzare il limite della ricerca e confrontare i tempi di calcolo con i vostri compagni.

2.12

SUB che sceglie un numero a random tra 1 e 1000. Usare la funzione `rand` per fare questo. L'utente poi inserisce numeri fintanto che non azzecca la risposta corretta. Il programma guida l'utente rispondendo se il tentativo era maggiore o minore del numero estratto. È una ricerca di tipo binario.

2.13

SUP che prende in input (oppure scritti direttamente nel codice, nell'inizializzazione dell'array) 20 numeri e ristampa l'array ordinato per ordine ascendente. Usare due cicli innestati, dove quello più interno confronta due numeri adiacenti, scambiandoli se fuori ordine. Il ciclo esterno continua fintanto che durante un ciclo completo non avvengono più scambi, quando quindi tutti i numeri sono in ordine.

Warning: Modificare questo esercizio quando sarete in grado di scrivere funzioni che prendono in input un array. Nel frattempo risolvete il problema nel main.

Info: Riordinare un array, o *sorting*, è un problema estremamente ricorrente in programmazione. Per questo motivo è anche motivo di ricerca e quindi esistono molti algoritmi diversi. In questo video una bella visualizzazione mette a confronto come diversi algoritmi riordinano un array. L'audio è molesto ma ha il suo fascino. Notare come la dimensione degli array è diversa (colonnine più fitte) ma il tempo di esecuzione di ciascun algoritmo è simile. Nell'esercizio andiamo ad implementare il Bubble Sort, uno dei più lenti ma probabilmente il più semplice. Un esercizio avanzato, ma interessante, è implementare un altro algoritmo di sorting tra quelli della lista su Wikipedia e confrontare i tempi di calcolo usando array di grandi dimensioni riempiti con numeri randomici.

Esercizi `CodeStepByStep`

Tutti gli esercizi della categoria `array` saranno di vostra portata solamente quando saprete come fa una funzione a prendere come parametro un array.