ESERCIZI – VETTORI

1) Scrivere un metodo ***verifica*** che riceve un vettore **A** di **interi** di lunghezza **n**, e restituisce il valore **true** se **ogni** elemento di **A**, ad esclusione del primo e dell'ultimo, è **uguale** alla differenza tra l'elemento che lo segue e quello che lo precede, **false** altrimenti. Ad esempio, se **A**= [1, 2, 3, 5, 8, 13, 21] il metodo restituirà **true**.

2) Si scriva un metodo ***distinti*** che riceve un array di interi **V** e restituisce il numero di **elementi distinti** di **V**, cioè quelli che compaiono una sola volta in **V**.
Esempio: **V** = [15, 12, 3, 15, 7, 12]→ **n** = 2 (infatti, V[2] e V[4] compaiono una sola volta ciascuno).
Esempio: **V** = [15, 12, 15, 12]→ **n** = 0.

3) Scrivere un metodo ***verifica*** che riceve un vettore **L** di **interi** di lunghezza **n,** con **n** pari, e restituisce il valore **true** se la media della prima metà degli elementi di **L**, **è maggiore** del valore massimo della seconda metà degli elementi di L, **false** altrimenti. Ad esempio, se **L**= [8, 9, 7, 0, 6, 7] il metodo restituirà **true**.

4) Si scriva un metodo ***replicati*** che riceve un array di interi **V** e restituisce il numero di **elementi replicati** di **V**, cioè quelli che compaiono almeno due volte in **V**.
Esempio: **V** = [1, 2, 3, 2, 2, 3, 5]→ **n** = 5 (infatti, V[1] compare 3 volte, V[2] 2 volte, V[3] e V[4] 3 volte, V[5] 2 volte).
Esempio: **V** = [15, 12, 3, 7]→ **n** = 0.

5) Si scriva un metodo **esercizio** che riceve in input due vettori **V1** e **V2** di interi e restituisce un vettore **V3** di booleani così formato: il valore nell'i-sima posizione di **V3** sarà *true* se e solo se i valori nella **i**-sima posizione di **V1** e **V2** sono entrambi pari o entrambi dispari, *false* altrimenti.

Esempio: **V1** = [2, 5, 8, 3, 2] , **V2** = [4, 2, 3, 5, 0] allora **V3** = [true, false, false, true, true].

6) Si scriva un metodo **esercizio** che riceve in ingresso un vettore **v1**, e restituisce un vettore **v2** della stessa dimensione di **v1**. In particolare, **v2[i]** conterrà il primo elemento di **v1** che si incontra muovendosi dalla posizione **i+1** alla posizione **v1.length-1** ed il cui valore sia maggiore di **v1[i]**; se non esiste nessun elemento in **v1** che soddisfa la condizione, allora **v2[i]** sarà uguale a **v1[i]**. Si noti che l' ultimo elemento di **v2** sarà sempre uguale all'ultimo elemento di **v1**.

Ad esempio, se **v1** = [4, 5, 10, 6, 3, 8], il vettore restituito è **v2** = [5, 10, 10, 8, 8, 8].

7) Si scriva un metodo **esercizio** che riceve in input due vettori **A** e **B** di interi e restituisce un vettore **C** di booleani così formato: il valore nell'**i**-sima posizione di **C** sarà *true* se e solo se il valore nell'**i**-sima posizione di **A** è multiplo oppure sottomultiplo del valore contenuto nell'**i**-esima posizione di **B**, *false* altrimenti.

Esempio: **A** = [3, 4, 10, 13, 5] , **B** = [2, 2, 5, 7, 15] allora **C** = [false, true, true, false, true].

8) Si scriva un metodo **esercizio** che riceve in ingresso un vettore **v1**, e restituisce un vettore **v2** della stessa dimensione di **v1**. In particolare, **v2[i]** conterrà il primo elemento di **v1** che si incontra muovendosi dalla posizione **v1.length-1** alla posizione **i+1** ed il cui valore sia minore di **v1[i]**; se non esiste nessun elemento in **v1** che soddisfa la condizione, allora **v2[i]** sarà uguale a **0**. Si noti che l' ultimo elemento di **v2** sarà sempre uguale a **0**.

Ad esempio, se **v1** = [2, 6, 3, 1, 2, 9], il vettore restituito è **v2** = [1, 2, 2, 0, 0, 0].

9) Si scriva un metodo ***componivett*** che riceve un array di interi **V1** e restituisce un array di interi **V2** contenente gli elementi **V1[i]**, con **0 < i < V1.length**, che sono multipli di tutti gli elementi che li precedono nel vettore **V1.**
Ad esempio, se **V1** = [5, 2, 10, 30, 3,90] allora il risultato sarà **V2** = [10,30,90] .

10) Si scriva un metodo ***costruisciVettore*** che riceve in ingresso un array **V** di interi positivi ed un intero **k** e crea e restituisce un array di interi **W** in cui l'elemento **W[i]** è così calcolato**:**

W[i]=a + b dove **a** è la somma degli elementi che precedono l'elemento i-esimo di **V** e **b** è il prodotto degli elementi che seguono l'elemento i-esimo di **V,** se **a + b > k**; **W[i]=a** altrimenti (Se non c'e' nessun elemento alla destra dell'elemento i-esimo si assuma **b**=0, se non c'e' nessun elemento alla sinistra dell'elemento i-esimo si assuma **a**=0).

Ad esempio, se **k=10** e **V** = [2, 3, 1, 1, 7], allora il risultato sarà **W** = [21, 2, 12, 13, 7].

11) Si scriva un metodo ***costruisciVettore*** che riceve in ingresso un array **V** di interi positivi ed un intero **k** e crea e restituisce un array di interi **W** in cui l'elemento **W[i]** è così calcolato**:**

W[i]=a + b dove **a** è il prodotto degli elementi che precedono l'elemento i-esimo di **V** e **b** è la somma degli elementi che seguono l'elemento i-esimo di **V,** se **a + b < k**; **W[i]=a** altrimenti (Se non c'e' nessun elemento alla destra dell'elemento i-esimo si assuma **b**=0, se non c'e' nessun elemento alla sinistra dell'elemento i-esimo si assuma **a**=0).

Ad esempio, se **k=14** e **V** = [2, 3, 1, 1, 7], allora il risultato sarà **W** = [12, 11, 6, 13, 6].

12) Si scriva un metodo ***componivett*** che riceve un array di interi **V1** e restituisce un array di interi **V2** contenente gli elementi **V1[i]**, con **0 < i < V1.length**, che non sono multipli di alcun elemento che li precede nel vettore **V1.**
Ad esempio, se **V1** = [5, 2, 11, 15, 3, 90] allora il risultato sarà **V2** = [2, 11, 3] .

13) Si scriva un metodo ***costruisciVettore*** che riceve in ingresso un vettore di interi **v1**, e restituisce un vettore **v2** della stessa dimensione. In particolare, il vettore **v2** è così costruito:

- **v2[i]** è pari alla media degli elementi di **v1** con indice >= di **i**, se tale media è >= di **v1[i]**;
- altrimenti **v2[i]** è pari alla differenza tra la somma degli elementi alla sinistra di **v1[i]** e la somma degli elementi alla destra di **v1[i]** (ovviamente se non c'e' nessun elemento alla destra o alla sinistra tale somma vale zero).

Ad esempio, se **v1**= [20 , 9 , 4, 8 , 2] il vettore restituito è **v2** = [-23, 6, 4, 31, 2].

14) Si scriva un metodo ***generaVettore*** che riceve in ingresso un vettore di interi **v1**, e restituisce un vettore **v2** della stessa dimensione. In particolare, il vettore **v2** è così costruito:

- **v2[i]** è pari alla media degli elementi di **v1** con indice <= di **i**, se tale media è <= di **v1[i]**;
- altrimenti **v2[i]** è pari alla differenza tra la somma degli elementi alla destra di **v1[i]** e la somma degli elementi alla sinistra di **v1[i]** (ovviamente se non c'e' nessun elemento alla destra o alla sinistra tale somma vale zero).

Ad esempio, se **v1**= [20 , 9 , 4, 8 , 2] il vettore restituito è **v2** = [20, -6, -19, -31, -41].

15) Si scriva un metodo ***verificaCoppie*** che riceve un array di interi **V** e restituisce *true* se tutte le coppie non sovrapposte di elementi consecutivi in **V** hanno prodotto maggiore del massimo valore in **V**, *false* altrimenti. Ad esempio, se **V**=[3, 4, 5, 6, 2, 7, 6], il risultato sarà *true* perché, essendo il massimo valore in **V** pari a 7, si ha 3 * 4 > 7, 5 * 6 > 7 e 2 * 7 > 7. Si noti che, se la dimensione di **V** è dispari, un elemento di **V** non deve essere considerato nel calcolo dei prodotti.

16) Si scriva un metodo ***verificaSequenze*** che riceve un array di interi **V** ed un intero **k** e restituisce *true* se tutte le sequenze non sovrapposte di 3 elementi consecutivi in **V** hanno somma uguale a **k**, *false* altrimenti. Ad esempio, se **V**=[3, 4, 5, 6, -1, 7, 0, 1, 11, 2] e **k** = 12, il risultato sarà *true* perché 3 + 4 + 5 = 12, 6 – 1 + 7 = 12 e 0 + 1 + 11 =12. Si noti che, se la dimensione di **V** non è un multiplo di 3, alcuni degli ultimi elementi di **V** non devono essere considerati nel calcolo delle somme.

17) Si scriva un metodo ***verificaVettore*** che riceve in ingresso un vettore di interi **V** e restituisce *true* se ciascun elemento di V maggiore di 2 è uguale alla somma di 2 elementi di **V**; *false* altrimenti.

Ad esempio se **V**=[1,12,3,7,5,2] il risultato sarà *true,* in quanto 12=7+5, 3= 1+2; 7=5+2; 5=3+2.

18) Si scriva un metodo ***compattaRepliche*** che riceve in ingresso un vettore di interi **v**, e restituisce un vettore di interi **w** ottenuto riportando in esso i valori presenti una sola volta in **v** e sostituendo gli elementi replicati con la loro somma. Per eliminare correttamente le repliche, prima di inserire un elemento si suggerisce di controllare se è una replica di uno già considerato. Ad esempio, se **v** = [7, 3, 3, 4, 1, 8, 6, 6, 6, 7] , il vettore restituito è **w** = [14, 6, 4, 1, 8, 18].

19) Si scriva un metodo booleano ***verificaVettore*** che riceve in ingresso un vettore di interi **V** di dimensione pari ed un intero **k** e restituisce un vettore di boolean B la cui dimensione è pari a metà della dimensione di V ed il cui generico elemento B[i]=*true* se la somma del corrispondente elemento di V e del suo simmetrico è un multiplo di k; B[i]=*false* altrimenti. Ad esempio se **V**=[15,4,7,3,12,5] e **k**=4 B =[true,false,true]

20) Si scriva un metodo ***filtraVettore*** che riceve in ingresso un vettore di interi **V** e ne restituisce un copia senza gli elementi che violano l'ordinamento decrescente di **V**. Più precisamente un elemento di **V** viene inserito nel vettore risultato **W** se è minore di tutti gli elementi che lo precedono. Ad esempio se **V**=[16,12,13,5,4,4,6,2,1], il risultato sarà **W**=[16,12,5,4,2,1].