

Programmazione ad Oggetti Traccia Laboratorio 4 ottobre 2019

Utilizzando la classe `VectorInt` vista a lezione implementare in un file `main.cpp` le seguenti funzioni:

1. `void aggiungiElementi(VectorInt& v)`; genera un numero casuale di numeri interi e li inserisce all'interno di `v`.
2. `void rimuoviElementi(VectorInt& v, int n)`; elimina gli ultimi `n` elementi dal vettore.
3. `void scambiaVettori(VectorInt& v1, VectorInt& v2)`; scambia il contenuto dei vettori `v1` e `v2`.
4. `int massimo(VectorInt& v)`; restituisce il numero massimo in `v`. Se `v` è vuoto restituisce `INT_MIN`.
5. `int minimo(VectorInt& v)`; restituisce il numero minimo in `v`. Se `v` è vuoto restituisce `INT_MAX`.
6. `VectorInt sommaVettori(VectorInt& v1, VectorInt& v2)`; crea (e restituisce) un nuovo vettore sommando il contenuto dei vettori `v1` e `v2`. Si può assumere che `v1` e `v2` abbiano la stessa dimensione. Esempio: `v1=(0,2,4)`, `v2=(3,4,1)` si deve restituire `(3,6,5)`.
7. `int prodottoScalare(VectorInt& v1, VectorInt& v2)`; calcola il prodotto scalare tra `v1` e `v2`. Si può assumere che `v1` e `v2` abbiano la stessa dimensione. Si ricorda che il prodotto scalare tra due vettori $(u_1, u_2, u_3, \dots, u_N)$ e $(k_1, k_2, k_3, \dots, k_N)$ si calcola come $u_1*k_1 + u_2*k_2 + u_3*k_3 + \dots + u_N*k_N$.
8. `bool opposto(VectorInt& v1, VectorInt& v2)`; restituisce `true` se per ogni numero `N` in `v1`, il suo opposto `-N` è in `v2`.
9. `int sottosequenza(VectorInt& v)`; restituisce la dimensione della più grande sottosequenza di numeri pari in `v`.
10. `int frequente(VectorInt& v)`; restituisce il numero che occorre più spesso in `v`, in caso di parità, restituisce il primo. Si può assumere che `v` contenga almeno un elemento.