1. Proste operacje na wektorach

1 Zadanie

Uzupełnij załączony program o definicje funkcji operujących na wektorach:

1. `void linspace(double v[], double start, double stop, int n);`
   Funkcja wypełnia tablicę rzeczywistą `v n` wartościami równomiernie rozłożonymi w przedziale `[start, stop]`. Wartość n powinna być nieujemna; dla $n = 1$ funkcja wpisuje do początkowego elementu tablicy wartość `start`. Dla $n = 0$ funkcja nie zmienia zawartości tablicy.

2. `void add(double v1[], const double v2[], int n);`
   Funkcja dodaje wektory, których elementy są zapisane w początkowych `n` elementach rzeczywistych tablic `v1` i `v2`. Funkcja zapisuje wektor będący sumą na miejscu wektora `v1`.

3. `double dot_product(const double v1[], const double v2[], int n);`
   Funkcja oblicza i zwraca iloczyn skalarny wektorów `v1` i `v2` o długości n.

4. `void multiply_by_scalar(double v[], int n, double s);`
   Funkcja mnoży każdy element tablicy rzeczywistej `v` (o długości n) przez liczbę rzeczywistą `s`. Wynik mnożenia jest zapisywany w tablicy `v`.

5. `void range(double v[], double start, double step, int n);`
   Funkcja wpisuje do n początkowych elementów rzeczywistej tablicy `v` wartości ciągu arytmatycznego o postępie `step`. Postęp może mieć wartość ujemną - wtedy kolejne elementy tablicy będą stanowić ciąg malejący.

   Wartość n powinna być nieujemna. Dla $n = 1$ funkcja wpisuje do początkowego elementu tablicy wartość `start`. Dla $n = 0$ funkcja nie zmienia zawartości tablicy.

Uzupełnij też pomocniczą funkcję `void read_vector(double v[], int n)`, która czyta ze standardowego wejścia n elementową tablicę rzeczywistą `v`.

2 Wejście

Pierwszy wiersz standardowego wejścia zawiera jedną liczbę naturalną $1 \leqslant F \leqslant 5$, oznaczającą kod funkcji do wykonania, zgodny z numeracją z poprzedniego paragrafu. Kolejne wiersze są zależne od wartości F, i tak:

1. $F = 1$: druga linia wejścia zawiera dwie liczby rzeczywiste (`start` i `stop`) i jedną liczbę całkowitą (wartość $0 \leqslant n \leqslant 100$).

2. $F = 2$: druga linia zawiera jedną liczbę całkowitą $0 < n \leqslant 100$ (długość dodawanych wektorów). Kolejne dwie linie zawierają po n liczb rzeczywistych każda (elementy wektorów `v1` i `v2`).

3. $F = 3$: druga linia zawiera jedną liczbę całkowitą $0 < n \leqslant 100$ (długość mnożonych wektorów). Kolejne dwie linie zawierają po n liczb rzeczywistych każda (elementy wektorów `v1` i `v2`).

4. $F = 4$: druga linia zawiera jedną liczbę całkowitą $0 < n \leqslant 100$ (długość wektora) i jedną liczbę rzeczywistą s (przez którą mnożymy elementy wektora v). Kolejna linia zawiera n liczb rzeczywistych (elementy wektora `v`).

5. $F = 5$: druga linia wejścia zawiera dwie liczby rzeczywiste (`start` i `step`) i jedną liczbę całkowitą (wartość $0 \leqslant n \leqslant 100$).

3 Wyjście

Wyjście programu również zależy od użytej funkcji. Dla $F = 1, 2, 4, 5$ program wypisuje (w jednej linii) elementy wyznaczonego wektora. Dla $F = 3$ program wypisuje jedną liczbę rzeczywistą - iloczyn skalarny wektorów.

Wszystkie liczby rzeczywiste powinny być wyprowadzone z dwoma miejscami po kropce dziesiętnej (format "%.2f").

4 Przykłady

4.1 Wejście

```
1
-10 10 11
```

Wyjście

```
-10.00 -8.00 -6.00 -4.00 -2.00 0.00 2.00 4.00 6.00 8.00 10.00
```

4.2 Wejście

```
2
5
1 2 3 4 5.5
6 5 2 7.5 3
```

Wyjście

```
7.00 7.00 5.00 11.50 8.50
```

4.3 Wejście

```
3
5
1 2 3 4 5.5
6 5 2 7.5 3
```

Wyjście

```
68.50
```

4.4 Wejście

```
4
5 3.5
1 2 3 4 5
```

Wyjście

```
3.50 7.00 10.50 14.00 17.50
```

4.5 Wejście

```
5
1 -0.4 5
```

Wyjście

```
1.00 0.60 0.20 -0.20 -0.60
```