1. Operacje na macierzach

Uwagi ogólne

W segmencie głównym programu są zdefiniowane tablice tablic (tablice "dwuwymiarowe") `A[SIZE][SIZE]`, `B[SIZE][SIZE]`, `C[SIZE][SIZE]`, do których dane są wczytywane w `main()`. Funkcje, których definicje należy uzupełnić, wykonują obliczenia korzystając z tych tablic. Rozmiarów tych tablic nie należy zmieniać.

Wartości rzeczywiste (typu `double`) wypisujemy z dokładnością 4 miejsc po kropce dziesiętnej.

1 Mnożenie macierzy

Szablon programu należy uzupełnić o definicję funkcji `matrix_product()`, która oblicza iloczyn macierzy `A` i `B` i zapisuje go w macierzy `AB`.

- **Wejście**
  1
  liczba wierszy i liczba kolumn macierzy `A`, liczba kolumn macierzy `B`
  elementy macierzy `A`
  elementy macierzy `B`
- **Wyjście**
  elementy macierzy `AB`
- **Przykład:**

  Wejście:

  ```
  1
  2 3 2
  1 2 3
  10 20 30
  11 23
  1 1.5
  -2 0
  ```

  Wyjście:

  ```
  7.0000 26.0000
  70.0000 260.0000
  ```

2 Triangularyzacja macierzy i obliczanie wyznacznika - wersja uproszczona (bez zamiany wierszy)

Szablon programu należy uzupełnić o definicję funkcji `gauss_simplified()`, która przekształca macierz kwadratową `A` o wymiarach $n \times n$ do postaci trójkątnej górnej metodą Gaussa i zwraca wartość wyznacznika. W przypadku, gdy element na przekątnej głównej jest równy zeru, triangularyzacja nie jest kończona, a wyznacznik = NAN.

Funkcja może zmienić wartości elementów tablicy `A`.

- **Wejście**
  2
  n – liczba wierszy/kolumn macierzy `A`
  elementy macierzy `A`
- **Wyjście**
  wyznacznik macierzy

- **Przykład:**
  Wejście:
  ```
  2
  4
  1 1 0 3
  2 1 -1 1
  3 -1 -1 2
  -1 2 3 -1
  ```
  Wyjście:
  ```
  39.0000
  ```

3 Rozwiązywanie układu równań liniowych metodą Gaussa - wersja z rozszerzaną macierzą współczynników

Szablon programu należy uzupełnić o definicję funkcji `gauss(double A[][SIZE], const double b[], double x[], int n, double eps)`, która przekształca macierz kwadratową `A` do postaci trójkątnej górnej metodą Gaussa i zwraca wartość wyznacznika. Wiersze macierzy są zamieniane tak, aby wartość bezwzględna elementu głównego była największa. Zamiana wierszy nie jest realizowana poprzez przepisanie wierszy w tablicy, lecz z zastosowaniem wektora permutacji indeksów wierszy. W przypadku, gdy po zamianie wierszy element na przekątnej głównej jest mniejszy od `eps`, triangularyzacja nie jest kończona, a wyznacznik przyjmuje wartość 0.

Jeżeli argumenty funkcji `b` i `x` oraz wyznacznik nie są zerowe, funkcja rozwiązuje układ równań i rozwiązanie zapisuje w tablicy `x`.

Funkcja może zmienić wartości elementów tablicy `A`.

- **Wejście**
  3

n – liczba wierszy/kolumn macierzy `A`
elementy macierzy `A`
elementy wektora `b`

- **Wyjście**
  wyznacznik macierzy
  elementy wektora `x`
- **Przykład:**
  Wejście:

```
3
4
1 -1 2 -1
2 -2 3 -3
1 1 1 0
1 -1 4 3
-8 -20 -2 4
```

Wyjście:

```
4.0000
-7.0000 3.0000 2.0000 2.0000
```

4 Odwracanie macierzy kwadratowej metodą Gaussa - Jordana

Szablon programu należy uzupełnić o definicję funkcji `matrix_inv(double A[][SIZE], double B[][SIZE], size t n, double eps)`, która wyznacza macierz `B` - odwrotną do nieosobliwej macierzy `A`. Należy zastosować metodę Gaussa - Jordana z rozszerzaniem macierzy `A` o macierz jednostkową. Wiersze macierzy rozszerzonej są zamieniane analogicznie jak w zadaniu 3. Funkcja zwraca wyznacznik macierzy `A`. W przypadku, gdy po zamianie wierszy element na przekątnej głównej jest mniejszy od `eps`, to algorytm odwracania nie jest kończony, a wyznacznik przyjmuje wartość 0 (układ równań nie jest rozwiązywany).

Funkcja może zmienić wartości elementów tablicy `A`.

- **Wejście**
  4
  n – liczba wierszy i macierzy `A`
  elementy macierzy `A`
- **Wyjście**
  wyznacznik macierzy
  elementy macierzy odwrotnej `B`
- **Przykład:**
  Wejście:

```
4
3
1 2 -1
2 1 0
-1 1 2
```

Wyjście:

```
-9.000
-0.2222 0.5556 -0.1111
0.4444 -0.1111 0.2222
-0.3333 0.3333 0.3333
```