5 Operacje na macierzach

5.1 Funkcje „pomocnicze”

W tej części zadania będą definiowane podstawowe funkcje przeznaczone do wykorzystania w bardziej złożonych algorytmach przedstawionych w drugiej części tematu.

5.1.1 Permutacje wierszy tablicy

W zadaniu zastosowane są dwa sposoby implementacji tablicy łańcuchów znakowych:

- tablica wskaźników `char *keywords_ptab[N]`,
- dwuwymiarowa tablica znakowa (dokładniej: tablica tablic znakowych) `char keywords_t2D[N][STRLEN_MAX]`.

Zadanie polega na sortowaniu łańcuchów znakowych zapisanych w tych tablicach bez zmiany ich położenia w pamięci, tj. w czasie sortowania każdy łańcuch pozostaje w pamięci pod niezmienionym adresem. Należy uzupełnić definicje funkcji:

1. `ptab_sort(char *ptab[], size_t n)`. W tablicy `ptab` są zapisane adresy łańcuchów znakowych. Funkcja sortuje elementy tej tablicy (adresy) w kolejności alfabetycznej łańcuchów, na które te elementy wskazują. Do sortowania należy użyć – podobnie jak w zadaniu 4.1 – biblioteczną funkcję `qsort(...)`. Funkcja ta wywołuje funkcję `compar(...)`, którą też należy zdefiniować.

2. `t2D_sort(const char t2D[][STRLEN_MAX], size_t indices[], size_t n)`. W `n` wierszach tablicy `t2D` są zapisane łańcuchy znaków. Stosując algorytm sortowania bąbelkowego, funkcja ma uporządkować elementy tablicy tej w kolejności odpowiadającej odwrotnemu porządkowi alfabetycznemu wskazywanych wierszy[1]. Istotnym warunkiem jest pozostawienie łańcuchów w tym samym miejscu w pamięci (w tych samych wierszach tablicy), w jakim były przed sortowaniem. Zadany porządek łańcuchów ma być zapisany w wektorze permutacji indeksów `indices`.
   W tablicy `indices` mają być zapisane indeksy wierszy tablicy `t2D`.

3. `n_str_copy(char t2D[][STRLEN_MAX],char *ptab[],size_t n)`, która kopiuje łańcuchy wskazywane przez elementy tablicy wskaźników `ptab` do tablicy `t2D`.
   Uwaga: Założenie o pozostawaniu łańcuchów w tym samym miejscu pamięci dotyczy tylko sortowania – w tej funkcji kopiowanie jest dozwolone.

4. `print_ptab(char *ptab[], size_t n)`, która pisze łańcuchy znakowe wskazywane przez `n` pierwszych elementów tablicy `ptab`.

5. `print_t2D_ind(const char (*ptr)[STRLEN_MAX],const size_t *pindices,size_t n)`. Funkcja wyprowadza na ekran `n` łańcuchów znakowych zapisanych w tablicy tablic. Kolejność wypisywanych łańcuchów jest określona tablicą permutacji indeksów `indices`.
   Pierwszy parametr (argument formalny) funkcji jest zmienną typu wskaźnikowego do tablicy o `STRLEN_MAX` elementach typu `char`. W porównaniu z definicją pierwszego parametru funkcji `t2D_sort` (używającą operatora `[][]`), taka definicja tego parametru jest bardziej „naturalna” dla języka C – jawnie pokazuje, że do funkcji jest przekazywany adres pierwszego elementu tablicy (tym elementem jest tablica `STRLEN_MAX` znaków).
   Definicję funkcji należy zapisać wybierając jeden z 4 sposobów: z użyciem dwóch operatorów `[]`, dwóch operatorów dereferencji `*`, dwóch możliwości „mieszanych” (z jednym `[]` i jednym *). Poprawność pozostałych 3 wariantów też należy sprawdzić.

Test 1
Łańcuchy znaków (słowa kluczowe języka C) są definiowane w momencie inicjowania elementów tablicy wskaźników `keywords_ptab[]`. Funkcja `n_str_copy()` kopiuje je do tablicy znakowej `keywords_t2D[]`. Test wywołuje funkcje `ptab_sort()` i `t2D_sort()`, wczytuje liczbę łańcuchów `n` i wypisuje w uporządkowanej kolejności `n` łańcuchów z tablic `ptab` i `t2D`.

- **Wejście**
  Numer testu, liczba wypisywanych łańcuchów n
- **Wyjście**
  n początkowych łańcuchów uporządkowanej tablicy wskaźników
  n początkowych łańcuchów uporządkowanej tablicy tablic znaków

[1] Do określania alfabetycznej kolejności łańcuchów należy korzystać z bibliotecznej funkcji `strcmp(...)` lub `strncmp(...)`.

- **Przykład:**
  Wejście:
  1 3
  Wyjście:
  auto
  break
  case
  while
  volatile
  void

5.1.2 Mnożenie macierzy

Szablon programu należy uzupełnić o definicję funkcji `mac_product(double A[][SIZE], double B[][SIZE], double AB[][SIZE], size_t m, size_t p, size_t n)`. Macierz A o wymiarach $m \times p$ jest zapisana w tablicy `A`, a macierz B o wymiarach $p \times n$ jest zapisana w tablicy `B` (liczba kolumn `SIZE` $\geqslant m, p, n$).Funkcja oblicza iloczyn macierzy $A \cdot B$ i zapisuje go w tablicy `AB`.

Test 2

- **Wejście**
  Numer testu, liczba wierszy i liczba kolumn macierzy A
  elementy macierzy A
  liczba wierszy i liczba kolumn macierzy B
  elementy macierzy B
- **Wyjście**
  elementy macierzy AB
- **Przykład:**
  Wejście:
  2
  2 3
  1 2 3 10 20 30
  3 2
  11 23 1 1.5 -2 0
  Wyjście:
  7.0000 26.0000
  70.0000 260.0000

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

Tablice tablic `A[SIZE][SIZE], B[SIZE][SIZE], C[SIZE][SIZE]` są zdefiniowane i wypełniane wczytanymi danymi w segmencie głównym `main`. Rozmiarów tych tablic nie należy zmieniać.

Szablon programu należy uzupełnić o definicję funkcji `gauss_simplified(double A[][SIZE], size_t n)`. Macierz A o wymiarach $n \times n$ jest zapisana w tablicy `A` (liczba kolumn `SIZE` $\geqslant n$). Funkcja przekształca macierz kwadratową A do postaci trójkątnej górnej metodą Gaussa, zapisuje ją w tablicy `A` i zwraca wartość wyznacznika. W przypadku, gdy element na przekątnej głównej jest równy zeru, to triangularyzacja nie jest kontynuowana, a wyznacznik = NAN.

Test 3

- **Wejście**
  Numer testu, liczba wierszy (i kolumn) macierzy A
  elementy macierzy A
- **Wyjście**
  wyznacznik macierzy
  elementy macierzy A
- **Przykład 1:**
  Wejście:
  3
  3
  3 5 7
  1 -3 8
  2 4 -2

Wyjście:
82.0000
3.0000 5.0000 7.0000
0.0000 -4.6667 5.6667
0.0000 0.0000 -5.8571

- **Przykład 2:**
  Wejście:
  3
  3
  1.25 0.125 -2.5
  5.0 0.5 -3.2
  2.5 1.8 0.

  Wyjście:
  nan
  1.2500 0.1250 -2.5000
  0.0000 0.0000 6.8000
  0.0000 1.5500 5.0000

5.2 Rozwiązywanie układu równań liniowych, odwracanie macierzy

5.2.1 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[], size_t n, double eps)`, która przekształca macierz kwadratową A zapisaną w tablicy `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`, to triangularyzacja nie jest dokończana, a wyznacznik przyjmuje wartość 0.
Jeżeli argumenty funkcji `b` i `x` oraz wyznacznik nie są zerowe, to funkcja rozwiązuje układ równań i rozwiązanie zapisuje w tablicy `x`.
Funkcja może zmienić wartości elementów tablicy `A`.
Poprawność funkcji można sprawdzić korzystając z funkcji `mac_vec_product`

Test 4

- **Wejście**
  Numer testu
  liczba wierszy i macierzy `A`
  elementy macierzy `A`
  elementy wektora `b`

- **Wyjście**
  wyznacznik macierzy
  elementy wektora `x`

- **Przykład:**
  Wejście:
  4
  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

5.2.2 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 (i zapamiętuje w tablicy `B`) macierz odwrotną do nieosobliwej macierzy A zapisanej w tablicy `A`. Należy zastosować metodę Gaussa - Jordana z rozszerzaniem macierzy A o macierz jednostkową. Wiersze macierzy rozszerzonej są zamieniane analogicznie do zadania 5.2.1. 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, i wyprowadzany jest tylko wyznacznik = 0 (układ równań nie jest rozwiązywany).
Funkcja może zmienić wartości elementów tablicy `A`.
Poprawność funkcji można sprawdzić korzystając z funkcji `mac_product()`.

Test 5

- **Wejście**
  Numer testu
  liczba wierszy i macierzy `A`
  elementy macierzy `A`
- **Wyjście**
  wyznacznik macierzy
  elementy macierzy odwrotnej `B`
- **Przykład:**
  Wejście:
  5
  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