Wydział EiTI PW

Dokumentacja

Projekt 3.2

Paweł Maśluch, nr albumu 262955

Treść projektu

Zmodyfikować napisany uprzednio program, w taki sposób, aby możliwe były operacje na plikach (pobieranie danych wejściowych, zapisywanie danych wyjściowych, zapisywanie informacji o błędach). Plik z danymi wejściowymi powinien zawierać układ równań do rozwiązania zapisany w postaci układu *Cramera:*

$$\begin{cases} a_{11} \cdot x_1 + \cdots + a_{1n} \cdot x_n = b_1 \\ a_{21} \cdot x_1 + \cdots + a_{2n} \cdot x_n = b_2 \\ \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ a_{n1} \cdot x_1 + \cdots + a_{nn} \cdot x_n = b_n \end{cases}$$

Plik z danymi wyjściowymi powinien zawierać informację o wyliczonych wyznacznikach oraz znalezionych rozwiązaniach podanego w pliku wejściowym układu równań.

Uwagi:

- Liczba niewiadomych może być dowolna (nieograniczona rozmiarem żadnej statycznej struktury danych – konieczność wykorzystania dynamicznej alokacji pamięci).
- Program powinien uwzględniać przypadki szczególne i odpowiednio je identyfikować.
- Program powinien ponadto umożliwić edycję wczytanych z pliku danych, oraz ich ponowną modyfikację po wykonaniu obliczeń lub zasygnalizowaniu błędu.
- Wyniki kolejnych rozwiązań (po modyfikacji danych) powinny być zapisywane w oddzielnych plikach, których nazwy powinny być wprowadzane przez użytkownika w trakcie działania programu.
- Współczynniki przy zmiennych powinny być reprezentowane zmiennymi typu float lub double.

- Nazwy plików mogą być podawane jako argumenty wywołania programu lub w inny sposób (nie mogą być zdefiniowane w programie na stałe).
- W przypadku, w którym plik wejściowy będzie posiadał wszystkie wymagane parametry, program powinien rozpocząć działanie, w przeciwnym razie powinien poprosić użytkownika o podanie brakujących lub błędnych wartości.

Dodatkowe uwagi:

- Program nie powinien zakładać poprawności wprowadzanych danych wejściowych oraz powinien poinformować o ewentualnych nieprawidłowościach
- Wyznaczniki obliczać rekurencyjną metodą Laplace'a lub metodą eliminacji Gaussa
- Dokumentacja nie jest konieczna, choć jej brak będzie skutkował odjęciem ustalonej liczby punktów od oceny końcowej.

Opis programu

Plik wejściowy

W programie dane pobierane są z pliku wejściowego. Warto więc powiedzieć, jak powinien wyglądać poprawny plik wejściowy, a jak wygląda niepoprawny plik wejściowy.

Nazwa pliku wejściowego nie może mieć więcej niż MAX znaków (dodatkowo, nie akceptujemy spacji).
Dla przykładu, jeśli MAX = 5, natomiast nazwa pliku wejściowego to *qwerty* albo *A m c* , wtedy takie nazwy nie są akceptowane.
Jeśli jednak MAX = 5, natomiast nazwa pliku wejściowego to *AbC* , wtedy taką nazwę pliku wejściowego akceptujemy.

W pierwszym wierszu pliku wejściowego powinna znajdować się dodatnia liczba całkowita n, oznaczająca liczbę równań. W kolejnych $n^2 + n$ wierszach powinny znajdować się współczynniki rzeczywiste układu równań, po jednym współczynniku w każdym wierszu.

Przykładowo, dla pliku wejściowego

2
3
4
5
6
7
8

układ równań wygląda tak: $\begin{bmatrix} 3 & 4 \\ 6 & 7 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 5 \\ 8 \end{bmatrix}$

Zawartość kolejnych wierszy pliku wejściowego jest przez program ignorowana – przykładowo, plik wejściowy

2
4
5
7.
1
9
0
Pk;
p0)

uważamy za poprawny (układ równań wygląda wtedy tak: $\begin{bmatrix} 4 & 5 \\ 1 & 9 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 7 \\ 0 \end{bmatrix}$).

Poniżej podam przykłady plików wejściowych nieakceptowanych przez program.

1) Plik wejściowy

2
{pusty_wiersz}
4
5
7
1..

jest niepoprawny (druga wiersz jest pusty, a powinna być tam liczba rzeczywista).

2) Plik wejściowy

{pusty_wiersz}
4
5
7
1..

jest niepoprawny (pierwszy wiersz jest pusty, a powinna być tam liczba równań).

3) Plik wejściowy

0
4
5
7
1..

oraz

-1
0
9
*

jest niepoprawny (pierwszy wiersz dla obu tych plików wejściowych nie zawiera dodatniej liczby całkowitej).

4) Plik wejściowy

```
4.
1
M
```

oraz

```
.2m
8
0
```

jest niepoprawny (pierwszy wiersz w obu plikach wejściowych nie zawiera poprawnie zapisanej dodatniej liczby całkowitej).

5) Plik wejściowy

```
2
0
1
8..
```

oraz

```
2
9
.9i;
```

jest niepoprawny (w obu plikach wejściowych nie podano poprawnie zapisanej liczby rzeczywistej, odpowiednio w czwartym oraz trzecim wierszu).

6) Plik wejściowy

```
2
1
7
4
6
9
```

Jest niepoprawny (podano pięć współczynników układu równań, natomiast oczekiwano sześciu).

Plik wyjściowy

Wyniki pracy programu zapisywane są do pliku wyjściowego. Warto więc powiedzieć, jak powinien wyglądać poprawny plik wyjściowy, a jak wygląda niepoprawny plik wyjściowy.

Obostrzenia co do pliku wyjściowego dotyczą tylko jego nazwy, i są takie same, jak dla pliku wejściowego.

Stałe globalne

W programie używam następujących stałych globalnych:

- **const int MAX = 10** (maksymalna dozwolona liczba znaków dla nazwy pliku wejściowego oraz wyjściowego - w programie przyjąłem wartość 10)
- **const int POZ = 10** (podstawa systemu pozycyjnego, w którym podana jest liczba równań w pliku wejściowym – rozsądnym pomysłem jest system dziesiętny, stąd przyjąłem wartość 10)

Zmienne globalne

W programie nie używam zmiennych globalnych.

Funkcje i procedury

W programie wykorzystuję poniższe funkcje oraz procedury:

a) *double det(long int n, long int w, int * stare_kolumny, double ** A)*

Jest to funkcja zwracająca wyznacznik (jako liczbę zmiennoprzecinkową) macierzy A. Funkcja ta korzysta z metody rozwinięcia Laplace'a. Macierz A jest rozmiaru n x n. Zakładamy, że rozwijamy wyznacznik względem wiersza o numerze w (dodatkowo, do obliczenia wyznacznika nie uwzględniamy wierszy o numerach mniejszych niż w). Dalej, zbiór numerów kolumn, uwzględnianych przy wyliczeniu wyznacznika, znajduje się w tablicy stare_kolumny (tablica stare_kolumny ma n komórek).

b) *void podmiana(double ** A, long int j, double * b, long int n)*

Jest to procedura dokonująca zamiany miejscami j-tej kolumny macierzy A, oraz tablicy b. Macierz A ma wymiar n x n, natomiast tablica b ma n komórek.

c) *int main()*

Jest to funkcja główna całego programu, realizująca projekt. Dalej podaję skrócony opis działania tej funkcji. Najpierw próbujemy wczytać nazwę pliku wejściowego – jeśli nazwa ta nie jest akceptowalna, to kończymy działanie programu, inaczej program kontynuuje swe działanie. Dalej, próbujemy otworzyć plik wejściowy – jeśli nie udało się go otworzyć, kończymy działanie programu, inaczej kontynuujemy działanie programu. Dalej, próbujemy wczytać zawartość pliku wejściowego (wczytywane dane na bieżąco zapamiętujemy).

Jeśli w pewnym momencie uznamy, że zawartość ta nie jest akceptowalna (przykłady niepoprawnych zawartości plików wejściowych podałem wcześniej), kończymy działanie programu, inaczej kontynuujemy działanie programu. Dalej, próbujemy wczytać nazwę pliku wyjściowego – jeśli nazwa ta nie jest akceptowalna, to kończymy działanie programu, inaczej program kontynuuje swe działanie. Dalej, próbujemy otworzyć plik wyjściowy – jeśli nie udało się go otworzyć, kończymy działanie programu, inaczej kontynuujemy działanie programu. Dalej, do określenia, z jakim układem równań mamy do czynienia, oraz ewentualnego wyznaczenia rozwiązania, korzystamy z twierdzenia, zwanego wzorami Cramera (informacje te zapisywane są do pliku wyjściowego). Następnie kończymy działanie funkcji *main*, zwracając wartość 0.