PODSTAWY METOD NUMERYCZNYCH

Prof. dr hab. Krzysztof Dems

Treści programowe:

- Podstawy programowania w języku FORTRAN (FORmulae TRANslation).
- Podstawy obliczeń numerycznych, obliczenia dokładne i zaokrąglenia, rodzaje błędów.
- Algebra macierzy: podstawowe operacje macierzowe.
- Metody rozwiązywania układów liniowych równań algebraicznych.
- Metody rozwiązywania nieliniowego równania z jedną niewiadomą
- Wielomiany jednej i więcej zmiennych.
- Interpolacja i aproksymacja funkcji jednej zmiennej.
- Różniczkowanie i całkowanie numeryczne.
- Metody rozwiązywania równań różniczkowych.

Literatura:

1. Z.Fortuna, B. Macukow, J. Wąsowski , „Metody Numeryczne”, PWN, 1999
2. E. Kącki, A. Małolepszy, A. Romanowicz, „Metody numeryczne dla inżynierów”, PŁ., Łódź 2000
3. K. Dems, Materiały pomocnicze do wykładu

Podstawy programowania w języku FORTRAN (FORmulae TRANslation)

Silverfrost Fortran
Strona domowa:
http://www.silverfrost.com/

Program instalacyjny (Personal Edition=:
http://www.silverfrost.com/32/ftn95/ftn95_personal_edition.aspx

Środowisko programu: PLATO

Edycja
Kompilowanie
Linkowanie

I. STRUKTURA PROGRAMU

- Sztywny format kodu źródłowego.
- Kolumny 1 – 5: etykiety (ciąg 1 do 5 cyfr)
- Kolumna 6: znacznik kontynuacji poprzedniego wiersza (cyfry 1 – 9, $)
- Kolumny 7 – 72: deklaracje i instrukcje kodu
- Litera C w kolumnie 1: wiersz traktowany jako komentarz (pomijany przy tworzeniu kodu wynikowego).

ETAPY TWORZENIA KODU WYNIKOWEGO:

Kod źródłowy → kompilacja → linkowanie → kod wykonywalny

1234567 72

```
C  struktura programu
       PROGRAM nazwa

         Blok deklaracji

         Blok instrukcji


       END

         Blok funkcji i procedur
```

Program główny

Definicje podprogramów

- Nazwy zmiennych: ciąg liter i cyfr zaczynający się od litery.
- Linię kodu tworzą słowa kluczowe deklaracji i instrukcji oraz nazwy zmiennych.
- Wyrażenia: arytmetyczne, alfanumeryczne (literały znakowe), logiczne.

WYRAŻENIA ARYTMETYCZNE

Pisze się tak jak w matematyce, wykorzystując następujące **operatory arytmetyczne**:

- Dodawanie: + a1+bb2
- Odejmowanie: - a1-bb2
- Mnożenie: * a1*dom
- Dzielenie: / licz/mian
- Potęgowanie ** a**b
- Przypisanie = a=c+d

Wyrażenie arytmetyczne przyjmuje wartość całkowitą (INTEGER) lub rzeczywistą (REAL)

Przykład: `a=(b+c)**2/(4*(d-c)` $a = \frac{(b+c)^2}{4(d-c)}$

WYRAŻENIE LOGICZNE

Przyjmuje wartość TRUE (prawda) lub FALSE (fałsz)

Operatory logiczne:

- Koniunkcja (i): .AND. a.AND.b
- Alternatywa (lub): .OR. a.OR.b

Operatory relacji logicznych

- Równość (a=b): .EQ. a.EQ.b
- Większe (a>b): .GT. a.GT.b
- Większe lub równe (a≥b): .GE. a.GE.b
- Mniejsze (a<b): .LT. a.LT.b
- Mniejsze lub równe (a≤b): .LE. a.LE.b

Przykład:

```
1)          (a.EQ.b).AND.(c.GT.d)
2)          (a.EQ.b).OR.(c.GT.d)
```

1) Prawdziwe (TRUE) gdy a=b oraz c>d, inaczej FALSE
2) Prawdziwe (TRUE) gdy a=b lub c>d. Fałszywe (FALSE) gdy a≠b lub/i c≤d.

WYRAŻENIA ALFANUMERYCZNE (LITERAŁY ZNAKOWE)

Ciąg znaków ujętych w pojedyncze apostrofy ‘ ‘.

Przykład: `‘ala’`

TYPY ZMIENNYCH

- Całkowite INTEGER domyślnie nazwy zaczynają cię od liter: i, j, k, l, m, n
  Inne nazwy muszą być wyraźnie zadeklarowane

- Rzeczywiste REAL domyślnie zaczynają się od pozostałych liter alfabetu.
  Nazwy zaczynające się od liter i, j, k, l, m, n muszą być wyraźnie zadeklarowane.

- Znakowe CHARACTER muszą być wyraźnie deklarowane.

- Logiczne LOGICAL muszą być wyraźnie deklarowane.

BLOK DEKLARACJI

- W tej części programu następuje deklaracja zmiennych i ich typów.
- Zmiennych o typie domyślnym nie trzeba deklarować.

```
INTEGER nazwa1,nazwa2,nazwa3,. . . .

REAL nazwa1,nazwa2,nazwa3,. . . .

LOGICAL nazwa1,nazwa2,nazwa3,. . . .

CHARACTER[[*n]] nazwa1,nazwa2,nazwa3,. . . .
```

n określa liczbę znaków kodu ASCII zapisanych w zmiennej znakowej.

Deklaracja tablicy(indeksowanego zbioru zmiennych):

```
DIMENSION nazwa1(id1:ig1,id2:ig2,   ,idn:ign),
nazwa2(id1:ig1,id2:ig2,    ,idn:ign),
 albo
DIMENSION nazwa1(ig1,ig2,   ,ign),nazwa2(ig1,ig2,   ,ign), . .
```

`COMMON/nazwa/ lista zmiennych oddzielonych przecinkami`

Ta deklaracja pozwala na globalne wykorzystanie zmiennych o nazwach wymienionych na liście,, w różnych jednostkach programu. Wymienione zmienne mają charakter zmiennych *globalnych*. Zmienne nie wymienione na liście COMMON są dostępne jedynie w jednostce programu, w której wystąpiły i mają charakter zmiennych *lokalnych*..

Przykład:

```
C przykład części deklaracyjnej program
      PROGRAM suma
      INTEGER ai
      REAL idodaj1,idodaj2
      CHARACTER*5 p1,p2
      LOGICAL test
      DIMENSION ij(5),ab(2,5)
      COMMON/ala/a1,a2,a3
C teraz nastąpią instrukcje program
      Instr1
      Instr2
10    Instr3
.
      Instr10
200   Instr11
      END
C deklaracja podprogramu
     FUNCTION fun(x,y)
     COMMON/ala/a1,a2,a3
     Instr1
     Instr2
     END
```

BLOK INSTRUKCJI

INSTRUKCJE STERUJĄCE

`CALL` *nazwa*(parametry) – wywołanie procedury

`CONTINUE` - nic nie robi

```
DO
Instr 1
Instr 2
.
Instr n
END DO
```

Instrukcja powtarzania

albo

```
DO licz=start,stop,[[krok]]
Instr 1
Instr 2
.
Instr n
END DO
```

Instrukcja powtarzania

Instrukcje warunkowe:

Instr 1
`IF`(*wyrażenie*) *Inst 2*
Instr 3
Instr 4

Albo:

Instr 1
`IF`(*wyrażenie*) `THEN`
Inst 2
Instr 3
`ENDIF`
Instr 4

Albo:

Instr 1
`IF`(*wyrażenie*) `THEN`
inst 2
Instr 3
`ELSE`
Instr 4
Instr 5
`ENDIF`
Instr 6

Instrukcja skoku:

`GOTO` *etykieta*

Inne instrukcje:

PAUSE - zawiesza wykonywanie programu do momentu naciśnięcia klawisza ‘Enter’
STOP - kończy wykonywanie programu
EXIT - pozwala na opuszczenie petli powtarzania i przekazuje sterowanie do instrukcji następującej po END DO
RETURN - występuje w podprogramach, zwraca sterownie do programu, który wywołał podprogram

```
      PROGRAM dodawanie
C teraz nastąpią instrukcje program
     suma=0.0
      A=1.0
      DO
      suma=suma+a
      IF (suma.GT.5.0)EXIT
      END DO
2     CONTINUE
      suma=0.0
      DO i=1,8
      suma=suma+a
      END DO
      suma=0.0
      DO i=1,8
      suma=suma+a
      IF(suma.GE.5)THEN
      Suma=0.0
      GOTO 4
      ENDIF
      END DO
4     CONTINUE
      END
```

INSTRUKCJE WEJŚCIA/WYJŚCIA (I/O)

FORTRAN korzysta z logicznych urządzeń I/O definiowanych przez *numer logiczny*.

- Domyślne numery urządzeń:

  1 lub 5 - klawiatura
  2 lub 6 - ekran monitora

albo

  * - klawiatura na wejściu, ekran monitora na wyjściu.

- Inne numery urządzeń można powiązać z plikiem instrukcją:

  OPEN(*nr,* FILE=nazwa_pliku)

Np. OPEN(3,FILE='dane1.txt') wiąże urządzenie logiczne o numerze 3 z plikiem o nazwie „dane1.txt".

- Rozłączenie urządzenia z plikiem następuje przez instrukcję

  `CLOSE`(*nr*)

Np. CLOSE(3) zamyka powiązanie urządzenia logicznego o numerze 3 z plikiem o nazwie „dane1.txt”.

- Inne instrukcje związane z operacjami na plikach:

  `REWIND`(*nr*) - przewija do początku pliku

  `BACKSPACE`(*nr*) - cofa o jeden rekord

  `ENDFILE`(*nr*) - zapisuje znacznik końca pliku

- Instrukcja czytania

READ(*nr,format*) lista zmiennych rozdzielona przecinkami

Najczęstsza postać instrukcji:

READ(*nr,**) lista zmiennych rozdzielona przecinkami

Np.:

```
READ(*,*)a,b,i
```
- wczytuje z klawiatury dwie liczby rzeczywiste i jedną całkowitą.

```
OPEN(3, FILE='dane')
READ(3,*)a,b,i
```
- wczytuje z pliku o nazwie „dane” dwie liczby rzeczywiste i jedną całkowitą.

- Instrukcja pisania

`WRITE`(*nr,format*) lista zmiennych rozdzielona przecinkami

Najczęstsza postać instrukcji:

`WRITE`(*nr,**) lista zmiennych rozdzielona przecinkami
(*wypisuje na urządzeniu nr wartości zmiennych w domyślnym formacie*)

Np.:

```
WRITE(*,*)a,b,i
```
- pisze na ekranie dwie liczby rzeczywiste i jedną całkowitą.

```
OPEN(4, FILE='wynik')
WRITE(4,*)a,b,i
```
- zapisuje do pliku o nazwie „wynik” dwie liczby rzeczywiste i jedną całkowitą.

Formaty wyprowadzania danych:

WRITE(*nr,'(specyfikacja formatu)'*) lista zmiennych rozdzielona przecinkami

Lista zmiennych **musi być zgodna** z specyfikacją formatu.

specyfikacja formatu

a - ciąg znaków (wielkości alfanumeryczne kodu ASCII)
i*n* - liczba całkowita zawierająca *n* pozycji.
f*w*.*d* - liczba rzeczywista mająca *d* cyfr po przecinku i zawierająca łącznie *w* pozycji.
e*w*.*d* - liczba rzeczywista z przedziału 0 - 1mająca *d* cyfr po przecinku pomnożona przez potęgę liczby 10 i zawierająca łącznie *w* pozycji.
\ - kontynuacja pisania w tej samej linii.
/ - przjście do nastepnej linii.

Np.:
numer=15
rzecz=1.23
WRITE(*,'(a,i5,a,.f6.2,a,e10.3)')'Ala ma kota,', numer,',', rzecz,',', rzecz

Wydruk: Ala ma kota, xx 15,xx1.23, x0.123E+01 (x – spacja)

DEKLARACJA FUNKCJI

FUNCTION *nazwa*(lista parametrów formalnych)
Blok deklaracji
Instr 1
Instr 2
.
.
Instr n
nazwa= wyrażenie
RETURN
END

```
      PROGRAM test1
      DIMENSION x(5), y(5)
C     czytanie
      READ(*,*)x
      READ(*,*)y
C     sumowanie
      s1=0.0
      DO i=1,5
      s1=s1+suma(x(i),y(i))
      END DO
C     wydruk
      WRITE(*,*)'wynik sumowania= ',s1
      PAUSE
      STOP
      END

      FUNCTION suma(a,b)
      REAL a,b                                  (może nie być tej deklaracji)
      c2=2*a
      c3=2*b
      suma=c2+c3
      RETURN
      END
```

DEKLARACJA PROCEDURY

SUBROUTINE *nazwa*(lista parametrów formalnych)
Blok deklaracji
Instr 1
Instr 2
.
.
Instr n
RETURN
END

Wywołanie procedury w programie nadrzędnym:

CALL *nazwa*(lista parametrów aktualnych)

```
      PROGRAM test2
      DIMENSION x(5), y(5)
c czytanie
      READ(*,*)x
      READ(*,*)y
C  sumowanie
      s1=0.0
      DO i=1,5
      CALL suma(s1,x(i),y(i))
      END DO
C wydruk
      WRITE(*,*)'wynik sumowania= ',s1
      PAUSE
      STOP
      END

      SUBROUTINE suma(s,a,b)
      c2=2*a
      c3=2*b
      rob=c2+c3
      s=s+rob
      RETURN
      END
```