D5. Podprogramy rozwiązywania układu równań

W niniejszym dodatku zamieściliśmy procedury rozwiazywania układu równań liniowych. Krótki opis algorytmów omówiliśmy w rozdz. 10.1. Procedury mają bogaty komentarz nagłówkowy objaśniający ich użycie.

```
c
c=====================================================================
c
c     Subroutine LDUSL(LiDiag, Nmax, L,D, U, Pilot, Symetr, NrU, ierr)
c
c    FUNKCJA
c
c          Procedura  LDUSL  dokonuje  rozkladu  Symetrycznej  lub
c      nieSymetrycznej, nieujemnie (niedodatnio) okreslonej  macierzy  o
c      budowie profilowej (o ile na wejsciu jest ona  podana  w  postaci
c      nierozlozonej)  na iloczyn A =  L*D*U,  gdzie  L  jest  macierza
c      trojkatna  dolna  o  jednosciach  na  glownej  diagonali,  U  -
c      trojkatna gorna,  a  D  macierza  diagonalna.  W  przypadku,  gdy
c      macierz poczatkowa jest Symetryczna, L i U sie pokrywaja.
c
c    PRZYKLADY
c
c         Procedura  LDUSL  jest  wykorzystywana  przede  wszystkim  do
c      rozwiazywania  ukladu(ow)  rownan  liniowych  (Symetrycznych  lub
c      nieSymetrycznych) o profilowej (sky - line) macierzy lewej strony.
c         Rozwiazanie  ukladu  rownan  wymaga  wywolania  nastepujacej
c      sekwencji  procedur  (dla  macierzy  nieSymetrycznej - Symetr =
c      .FALSE.):
c
c      call LDUSL(LiDiag,Nmax,L,D,U,Pilot,Symetr,NrU,ierr,energ)
c      call UKSL(LiDiag,Nmax,L,D,U,B,Pilot,NrU,ierr)
c
c         Odpowiednie wywolania dla macierzy Symetrycznych jest nastepu-
c      jace (Symetr = .TRUE.):
c
c
c      call LDUSL(LiDiag,Nmax,U,D,U,Pilot,Symetr,NrU,ierr,energ)
c      call UKSL(LiDiag,Nmax,U,D,U,B,Pilot,NrU,ierr)
c
c         Nalezy podkreslic, ze w przypadku, gdy  chcemy  rozwiazac  ten
c      uklad dla wiecej niz  jeden  wektorow  prawej  strony,  to  LDUSL
c      wykonywane  jest  tylko  raz  -  przed  znalezieniem  pierwszego
c      rozwiazania.
```

```
c
c    WEJSCIE
c
c      LiDiag - /integer*4/ liczba elementow diagonalnych macierzy A,
c      Nmax - /integer*4/ liczba elementow czesci trojkatnej gornej  ma-
c          cierzy A,
c      U(Nmax) - /REAL*8/ wektor zawierajacy na wejsciu  kolejne kolumny
c          elementow znajdujacych sie ponad  glowna  przekatna  macierzy
c          wejsciowej w profilu  obejmujacym  wszystkie  jej  niezerowe
c          elementy. Przykladowo -  jesli  oznaczymy  taka  4x4  macierz
c          przez A:
c                          I   2.D0  0.D0  1.D0  0.D0   I
c                          I   0.D0  4.D0  0.D0  3.D0   I
c                          I  -1.D0  0.D0  8.D0  5.D0   I
c                          I   0.D0 -2.D0  5.D0  9.D0   I
c          to Nmax=4, U(1)=1.D0, U(2) = 0.D0, U(3) = 3.D0, U(4) = 5.D0,
c       D(LiDiag) - /REAL*8/  wektor  o  wymiarze  LiDiag zawierajacy na
c          wejsciu elementy diagonalne macierzy A,
c       L(Nmax) - /REAL*8/ wektor zawierajacy na wejsciu kolejne wiersze
c          profilu obejmujacego elementy niezerowe  lezace  na  lewo  od
c          glownej przekatnej macierzy A,
c             W przypadku, gdy A jest Symetryczna przy  wywolaniu  LDUSL
c          nalezy w miejsce L podac parametr aktualny odpowiadajacy U.
c             W przypadku, gdy macierz A jest nieSymetryczna (dla poda-
c          nego wyzej przykladu) wartosci L sa nastepujace: L(1) = -1.D0,
c          L(2) = 0.D0, L(3) = -2.D0, L(4) = 5.D0, a wiec zapamietane sa
c          w sposob analogiczny do  sposobu  przechowywania  macierzy  U
c          (zamiast kolumn wiersze),
c      Pilot(LiDiag) - /integer*4/  wektor  zawierajacy  indeksy
c          najnizszych  elementow  ponaddiagonalnych  w  kolejnych
c          kolumnach. Przyjeto zalozenie, ze Pilot(1) jest rowny zero, a
c          w  przypadku,  gdy  w  j-tej  kolumnie  nie  ma  elementow
c          niezerowych, Pilot(J) =  Pilot(J-1). Latwo zauwazyc,  ze  w
c          przypadku  macierzy  nieSymetrycznych  Pilot  przekazuje
c          analogiczne informacje o rozmieszczeniu elementow w  macierzy
c          trojkatnej dolnej L (indeksy  elementow  najnizszych  sposrod
c          ponaddiagonalnych  odpowiadaja  indeksom  tych  elementow  z
c          kolejnych wierszy,  ktore  lewostronnie  sasiaduja  z  glowna
c          przekatna). Pilot w  trakcie  dzialania  procedury  nie  jest
c          zmieniany. Dla macierzy podanej w przykladzie,  wartosci wek-
c          tora  sa  nastepujace: Pilot(1) = 0  (dla  kazdej  macierzy),
c          Pilot(2) = 0, Pilot(3) = 3, Pilot(4) = 4 i Nmax = 4,
c      Symetr  -  /logical*1/  stala rowna  .TRUE.  jesli  macierz  jest
c          Symetryczna i .FALSE. w przeciwnym przypadku,
c      NrU - /integer*4/ numer urzadzenia wyjsciowego na ktory wypisywa-
c          ne sa komunikaty o bledach. NrU rowne zero oznacza,  ze komu-
c          nikaty nie beda wyswietlane.
```

```
c
c    WYJSCIE
c
c      U - /REAL*8/ wektor o wymiarze conajmniej Nmax zawierajacy czynnik
c          rozkladu  macierzy  A  na  iloczyn  L*D*U,  gdzie  U  jest
c          trojkatna gorna, L - trojkatna  dolna,  a  D  jest  macierza
c          diagonalna. Pierwotna zawartosc wektora U zostaje zniszczona,
c      L - /REAL*8/ wektor o wymiarze conajmniej Nmax zawierajacy czynnik
c          rozkladu  macierzy  A  na  iloczyn  L*D*U,  gdzie  U  jest
c          trojkatna gorna, L - trojkatna  dolna,  a  D  jest  macierza
c          diagonalna. W przypadku macierzy Symetrycznych L  jest  rowne
c          U.
c             Pierwotna zawartosc wektora L zostaje zniszczona,
c      D - /REAL*8/  wektor  o  wymiarze  conajmniej LiDiag  zawierajacy
c          odwrotnosc czynnika rozkladu  macierzy A  na  iloczyn  L*D*U,
c          gdzie U jest trojkatna gorna, L - trojkatna dolna,  a D  jest
c          macierza diagonalna. Pierwotna zawartosc  wektora  D  zostaje
c          zniszczona,
c      ierr - /integer*4/ zmienna zawierajaca kod bledu wykrytego podczas
c          dzialania procedury. Moze ona przyjmowac nastepujace wartosci:
c          -J - utrata conajmniej siedmiu cyfr znaczacych przy obliczaniu
c          J-tego elementu diagonalnego,
c          LiDiag + J - macierz wejsciowa jest osobliwa  (otrzymano  zero
c          przy obliczaniu J-tego elementu diagonalnego),
c          J - macierz wejsciowa nie jest ani nieujemnie, ani niedodatnio
c          okreslona (znak J-tego elementu diagonalnego jest rozny od J-1
c          -szego.
c             Obliczenia pomimo wykrytych bledow nie sa przerywane.
c             W przypadku niewykrycia bledu ierr jest rowny zero.
c
c    METODA
c
c          Rozklad macierzy wejsciowej odbywa sie przy  pomocy  adaptacji
c      Zwartej Metody Crouta (odmiana eliminacji Gaussa) dla macierzy  o
c      podanej wczesniej budowie.  Polega  ona  na  rozkladzie  macierzy
c      wejsciowej na iloczyn A = L*D*U, gdzie U jest macierza  trojkatna
c      gorna o jednosciach na glownej diagonali, L -  trojkatna dolna, a
c      D - macierz - diagonalna.
c
c    BLEDY
c
c          LDUSL rozpoznaje trzy rodzaje  blednych  sytuacji  i  wypisuje
c      odpowiednie komunikaty na urzadzenie o numerze NrU:
c      1. utrata  conajmniej  siedmiu  cyfr  znaczacych  przy obliczaniu
c      elementu diagonalnego rozkladu L*D*U (zle uwarunkowanie macierzy),
c      2. macierz wejsciowa jest osobliwa,
c      3. macierz wejsciowa nie jest ani dodatnio ani ujemnie okreslona.
```

```
c         dodatkowo podawany jest indeks wiersza (kolumny)  przy  ktorej
c      wytapilo rozpoznanie blednej sytuacji. Oprocz tego zmiennej  ierr
c      nadawana jest odpowiednio dla kazdej z tych sytuacji wartosc: -J,
c      LiDiag + J oraz J, gdzie J - oznacza indeks kolumny  przy  ktorym
c      rozpoznano blad.
c
c    UZYTE PODPROGRAMY
c      dot (MacOper)    - iloczyn skalarny dwoch wektorow,
c      DREDUC (UkRow)   - procedura  wyznaczajaca N-ty  element
c          diagonalny  macierzy D  powstalej  przy  rozkladzie  macierzy
c          poczatkowej na iloczyn L*D*U oraz ostateczne wartosci  wstep-
c          nie zredukowanych wczesniej elementow N-tego wiersza macierzy
c          macierzy L i N-tej kolumny macierzy U.
c
c=======================================================================
c                       POLITECHNIKA WARSZAWSKA
c                    OSRODEK METOD KOMPUTEROWYCH
c                ZESPOL OPROGRAMOWANIA INZYNIERSKIEGO
c                         Z.K./J.O. 1988.11.16
c=======================================================================
c
      Subroutine LDUSL(LiDiag,Nmax,L,D,U,Pilot,Symetr,NrU,ierr)
c
      integer*4 J,JD,JH,JR,JRH,IDH,IE,IH,IS,LiDiag,Pilot,Nmax,ierr,I,ID
      integer*4 NrU
      real*8 DD,L,U,D
      real*8 dot
      logical*1 Symetr
      dimension L(Nmax),D(LiDiag),U(Nmax),Pilot(LiDiag)
c
c  Redukcja macierzy (kolumnami)
c
      ierr = 0
      JD = 1
      do 600 J = 1, LiDiag
        JR = JD + 1
        JD = Pilot(J)
        JH = JD - JR
        if (JH) 500,300,100
100       IS = J - JH
          IE = J - 1
          do 200 I = IS, IE
            JR = JR + 1
            ID = Pilot(I)
            IH = min0(ID-Pilot(I-1),I-IS+1)
            if (IH .LE. 0) goto 200
              JRH = JR - IH
```

```
              IDH = ID - IH + 1
              U(JR) = U(JR) - dot(U(JRH),L(IDH),IH)
              if (.not. Symetr) L(JR) = L(JR) - dot(L(JRH),U(IDH),IH)
200        continue
c
c  wyznaczanie elementu diagonalnego i koncowa redukcja j-tego wiersza
c  i kolumny
c
300        DD = D(J)
           JR = JD - JH
           JRH = J - JH - 1
           call DREDUC(JH+1, L(JR), D(JRH), U(JR), D(J), Symetr)
c
c  Obsluga bledow
c
           if (dabs(D(J)) .GE. 0.5D-7*dabs(DD)) goto 400
              ierr = -J
              if (NrU .ne. 0) write(NrU,2000)
              if (NrU .ne. 0) write(NrU,2010) J
400        if (DD*D(J) .GE. 0.0D0) goto 500
              ierr = LiDiag + J
              if (NrU .ne. 0) write(NrU,2020)
              if (NrU .ne. 0) write(NrU,2030) J
500        if (D(J) .NE. 0.0D0) goto 550
              ierr = J
              if (NrU .ne. 0) write(NrU,2040) J
              if (NrU .ne. 0) write(NrU,2050)
              goto 600
C
C  Obliczanie odwrotnosci macierzy D
C
550        D(J) = 1.0D0 / D(J)
600    continue
       return
2000   format(2x,' ***LDUSL ostrzezenie 1*** utrata conajmniej siedmiu,')
2010   format(2x,' cyfr znaczacych przy obliczaniu ',I5,'-tej kolumny.')
2020   format(2x,' ***LDUSL ostrzezenie 2*** macierz wejsciowa nie jest')
2030   format(2x,' ani dodatnio ani ujemnie okreslona. Zmiana znaku ',
     +       'przy obliczaniu',I5,'-tego elementu diagonalnego.')
2040   format(2x,' ***LDUSL ostrzezenie 3*** ',I5,'-ty element ',
     +       'diagonalny')
2050   format(2x,' jest rowny zero - macierz jest osobliwa.')
       end
c
c=======================================================================
c
c    Subroutine UKSL(LiDiag, Nmax, L, D, U, B, Pilot, NrU, ierr, energ)
```

```
c
c   FUNKCJA
c
c        Rozwiazywanie ukladu rownan liniowych L*D*U*x  =  b,  gdzie  L
c      jest macierza trojkatna dolna, D - diagonalna, a  U  -  trojkatna
c      gorna. Macierze L i U sa podane "sky-line'ie".
c
c   PRZYKLADY
c
c        W  celu  rozwiazania  ukladu  rownan  liniowych   z   macierza
c      profilowa ("sky-line") nalezy  najpierw  wykonac  procedure LDUSL
c      rozkladajaca macierz poczatkowa na iloczyn L*D*U.
c        W  celu  rozwiazania  jednego  (lub  wiecej)   ukladu   rownan
c      liniowych symetrycznych nalezy wykonac:
c      CALL LDUSL(LiDiag,Nmax,U,D,U,Pilot,.TRUE.,NrU,ierr,energ)
c      CALL UKSL(LiDiag,Nmax,U,D,U,B,Pilot,NrU,ierr,energ)
c        W  celu  rozwiazania  jednego  (lub  wiecej)   ukladu   rownan
c      liniowych niesymetrycznych nalezy wykonac:
c      CALL LDUSL(LiDiag,Nmax,L,D,U,Pilot,.FALSE.,NrU,ierr,energ)
c      CALL UKSL(LiDiag,Nmax,L,D,U,B,Pilot,NrU,ierr,energ)
c
c   WEJSCIE
c
c      U(Nmax) - /real*8/ wektor zawierajacy na wejsciu kolejne  kolumny
c          elementow znajdujacych sie ponad  glowna  przekatna  macierzy
c          trojkatnej gornej D - czynnika rozkladu L*D*U -  znajdujacego
c          sie w profilu obejmujacym wszystkie jego niezerowe  elementy.
c          Przykladowo dla 4x4 macierzy D rownej
c                          I   1.D0  0.D0  1.D0  0.D0   I
c                          I   0.D0  1.D0  0.D0  3.D0   I
c                          I   0.D0  0.D0  1.D0  5.D0   I
c                          I   0.D0  0.D0  0.D0  1.D0   I
c          U(1) = 1.D0, U(2) = 0.D0, U(3) = 3.D0, U(4) = 5.D0,
c       D(LiDiag) - /real*8/ wektor o  wymiarze  LiDiag  zawierajacy  na
c          wejsciu elementy diagonalne macierzy A,
c       L(Nmax) - /real*8/ wektor zawierajacy na wejsciu wiersze macierzy
c          trojkatnej dolnej.
c             W przypadku, gdy A jest symetryczna przy  wywolaniu  LDUSL
c          nalezy w miejsce L podac parametr aktualny odpowiadajacy U.
c             W przypadku, gdy macierz A jest niesymetryczna jest real*8
c          wektor elementow zawierajacych kolejne wiersze  (na  lewo  od
c          diagonali) profilu niezerowych elementow tej macierzy,
c      D - /real*8/  wektor  o  wymiarze   conajmniej LiDiag zawierajacy
c          odwrotnosc czynnika rozkladu  macierzy A  na  iloczyn  L*D*U,
c          gdzie U jest trojkatna gorna, L - trojkatna dolna,  a D  jest
c          macierza diagonalna. Pierwotna zawartosc  wektora  D  zostaje
c          zniszczona,
```

```
c      B(LiDiag) - /real*8/  wektor prawej strony ukladu rownan,
c      NrU - /integer*4/ numer urzadzenia wyjsciowego na ktory wypisywa-
c          ne sa komunikaty o bledach. NrU rowne zero oznacza,  ze komu-
c          nikaty nie beda wyswietlane.
c
c   WYJSCIE
c
c      B(LiDiag) - /real*8/ wektor rozwiazania ukladu  rownan. Pierwotna
c          jego zawartosc zostala zniszczona,
c      ierr - /integer*4/ zmienna rowna zero jesli wektor prawej  strony
c          nie jest zerowy i LiDiag w przeciwnym przypadku.
c      energ - /real*8/ energia sprezysta ukladu
c
c   METODA
c
c          Korzystajac  z  podanego  na  wejsciu  rozkladu  macierzy  na
c      iloczyn L*D*U rozwiazuje sie uklad rownan w trzech etapach:
c         1. Rozwiazywany jest uklad  L*w = b.
c         2. Rozwiazywany jest uklad  D*z = w.
c         3. Rozwiazywany jest uklad  U*x = z,
c      co jest, jak latwo  zauwazyc,  rownowazne  poczatkowemu  ukladowi
c      A*x = b. Rozwiazanie kazdego z tych ukladow jest oczywiste (1 i 3
c      macierze trojkatne, a 2 - macierz diagonalna)
c
c   BLEDY
c
c          UKSL sygnalizuje sytuacje,  gdy  nastepuje  proba  rozwiazania
c      ukladu rownan  jednorodnych  (zerowy  wektor  prawej  strony).  W
c      sytuacji, gdy macierz  poczatkowa  jest  nieosobliwa  nie  ma  to
c      wiekszego sensu - istnieje tylko jedne zerowe rozwiazanie.
c          Odpowiedni komunikat zostanie zapisany na urzadzeniu o numerze
c      logicznym NrU. Oprocz tego zmiennej informujacej o bledach - ierr
c      nadana zostanie warosc Lidiag.
c
c   UZYTE PODPROGRAMY
c
c      dot (MacOper)    - iloczyn skalarny dwoch wektorow,
c      COLRED(U,X,N,B)  - procedura  dokonujaca  redukcji  kolumny  przy
c          rozwiazywaniu ukladu rownan z macierza trojkatna gorna.
c
c=======================================================================
c
      Subroutine UKSL(LiDiag,Nmax,L,D,U,B,Pilot,NrU,ierr,energ)
c
      integer*4 Pilot,LiDiag,Nmax,IS,J,JH,JJH,JR1,NrU,ierr,JR,IE
      real*8 BD,B,D,L,U,dot,energ
      dimension L(Nmax),U(Nmax),D(LiDiag),B(LiDiag),Pilot(LiDiag)
```

```
c
c  Szukanie pierwszej niezerowej skladowej wektora prawej strony
c
      ierr = 0
      do 100 IS = 1, LiDiag
          if (B(IS) .ne. 0.0D0) goto 200
100   continue
      if (NrU .ne. 0) write(NrU, 2000)
      ierr = LiDiag
      return
200   if (IS .eq. LiDiag) goto 400
        IE = IS + 1
c
c  Rozwiazywanie ukladu rownan z macierza trojkatna dolna L*w = b
c
        do 300 J = IE, LiDiag
          JR = Pilot(J-1)
          JH = Pilot(J) - JR
          JJH = J - JH
          if (JH .LE. 0) goto 300
            JR1 = JR + 1
            if (JJH .gt. 0) B(J) = B(J) - dot(L(JR1),B(JJH),JH)
300   continue
c
c  Rozwiazywanie ukladu rownan z macierza diagonalna D*z = w
c
 400  energ = 0.d0
      do 500 J = IS, LiDiag
        BD = B(J)
        B(J) = B(J)*D(J)
 500  energ = energ + BD * B(J)
c
c  Rozwiazywanie ukladu rownan z macierza trojkatna gorna U*x = z,
c  co jest rownowazne rozwiazaniu ukladu (L*D*U)*x = B
c
      J = LiDiag
600   JR = Pilot(J-1)
      JH = Pilot(J) - JR
      JJH = J - JH
      if (JJH .gt. 0) call ColRed(U(JR+1),B(J),JH,B(JJH))
        J = J - 1
        if (J .gt. 1) goto 600
      return
2000  format(' ***UKSL ostrzezenie 1*** zerowy wektor prawej strony.')
      end
c
c=====================================================================
```

```
c
c     Subroutine DREDUC(N, L, D, U, DJ, Symetr)
c
c   FUNKCJA
c
c        Procedura DREDUC mnozy kolejne elementy wektora U przez
c      odpowiadajace im elementy wektora D, nastepnie oblicza jego
c      iloczyn skalarny z wektorem L i otrzymana wartosc odejmuje od DJ.
c      Ponadto, jesli SYMETR jest rowny .FALSE., zmodyfikowany zostaje
c      wektor L w ten sposob, ze kolejne jego elementy sa rowne
c      "starym" pomnozonym przez odpowiadajace im elementy wektora D.
c
c    ZASTOSOWANIE
c
c        Podstawowym zastosowaniem DREDUC jest redukcja kolumn
c      macierzy trojkatnej gornej (w przypadku macierzy niesymetrycznych
c      - takze wierszy macierzy trojkatnej dolnej) oraz wyznaczenie
c      odwrotnosci kolejnego elementu macierzy diagonalnej podczas
c      wyznaczania rozkladu macierzy na iloczyn L*D*U, gdzie L jest
c      macierza trojkatna dolna, D - diagonalna, a U - trojkatna gorna.
c        Procedura wyznaczjaca rozklad ma nazwe LDUSL.
c
c    WEJSCIE
c
c      N - /INTEGER*4/ dlugosc wektorow L i U (indeks obliczonego w
c          poprzednim kroku elementu diagonalnego),
c      L - /REAL*8/ wektor o dlugosci conajmniej N (wiersz macierzy
c          trojkatnej dolnej L). W przypadku, gdy macierz jest
c          symetryczna L pokrywa sie z U,
c      D - /REAL*8/ wektor o dlugosci conajmniej N (liczba
c          obliczonych dotad elementow diagonalnych D),
c      U - /REAL*8/ wektor o dlugosci conajmniej N (kolumna macierzy
c          trojkatnej gornej L).
c      SYMETR - /LOGICAL/ stala wejsciowa rowna .TRUE. L ma zostac
c          niezmienione (macierz symetryczna) i .FALSE. w przeciwnym
c          przypadku,
c      DJ - /REAL*8/ zmienna (odpowiada aktualnie elementowi diagonalnemu
c          macierzy D - rownemu odpowiadajacemu jej elementowi macierzy
c          poczatkowej),
c
c    WYJSCIE
c
c      U - /REAL*8/ zawiera elementy podane na wejsciu i pomnozone przez
c          odpowiadajace im elementy wektora D.
c      L - /REAL*8/ wektor - jesli SYMETR jest rowny .TRUE. L jest rowne
c          U, a w przeciwnym przypadku kolejne elementy L sa (podobnie
c          jak U) mnozone przez odpowiadajace im elementy wektora D,
```

```
c      DJ - /REAL*8/ zmienna zawierajaca roznice  miedzy stara wartoscia
c           a iloczynem skalarnym zmodyfikowanego wektora U i wektora L.
c
c=====================================================================
c
      Subroutine DREDUC(N, L, D, U, DJ, Symetr)
c
      integer*4 N, j
      logical*1 Symetr
      real*8 L, D, U, DJ, pom
      dimension L(N), D(N), U(N)
c
      do 100 j = 1, N
        pom = U(J) * D(J)
        DJ = DJ - L(J) * pom
        U(J) = pom
 100  continue
c
c     Dodatkowe obliczenia sa potrzebne dla macierzy niesymetrycznej
c
      IF (Symetr) return
      do 200 j = 1, N
          L(j) = L(j) * D(j)
 200  continue
      return
      end
c
c=====================================================================
c
c    Subroutine ColRed(u, x, n, b)
c
c    FUNKCJA
c
c      Odjecie od wektora B wektora U pomnozonego przez X
c
c    WEJSCIE / WYJSCIE
c
c      n                - /integer*4/ wymiar wektorow U i B,
c      x                - /real*8/ mnoznik,
c      b                - /real*8/ modyfikowany wektor,
c      u                - /real*8/ wektor.
c
c=====================================================================
c
      Subroutine ColRed(u, x, n, b)
c
      real*8 u, b, x
```

```
      integer*4 n, i
      dimension u(n), b(n)
c
      if(n .lt. 1) return
c
      do 100 i = 1, n
      b(i) = b(i) - u(i)*x
 100  continue
      return
      end
c
c=====================================================================
c
c     real*8 function dot(a, b, n)
c
c    FUNKCJA
c
c      Obliczanie iloczynu skalarnego wektorow a i b o dlugosci n
c
c    WEJSCIE
c
c      n                - /integer*4/ dlugosc wektorow a i b,
c      a, b             - /real*8/ wektory odlugosci n.
c
c=====================================================================
c
      real*8 function dot(a, b, n)
c
      real*8 a, b, dot
      integer*4  n, i
      dimension a(n), b(n)
c
      dot = 0.0d0
      do 100 i = 1,n
        dot = dot + a(i)*b(i)
 100  continue
      return
      end
```