ΑΝΟΙΚΤΑ ακαδημαϊκά μαθήματα ΠΠ

Εισαγωγή στον Προγραμματισμό Η/Υ για Χημικούς Μηχανικούς

Παρουσίαση Διαλέξεων: 11. Διεπιφάνειες

Καθηγητής Δημήτρης Ματαράς

Copyright © 2014 by Prof. D. S. Mataras (mataras@upatras.gr). This work is made available under the terms of the Creative Commons Attribution-Noncommercial-NoDerivs 3.0 license, http://creativecommons.org/licenses/by-nc-nd/3.0/

Φτιάχνω διεπιφάνειες με τη δομή **INTERFACE**

1. **Ρητή Διεπιφάνεια Διαδικασιών**
   - Χρήση Εξωτερικών Διαδικασιών
   - Συνεργασία με Διαδικασίες γραμμένες σε C (και άλλες γλώσσες)

2. **Διεπιφάνεια Γενικών Διαδικασιών (generic procedures)**
   - Υπερφόρτωση Εγγενών Συναρτήσεων

3. **Δημιουργία Διεπιφάνειας Τελεστών**
   - Δημιουργία Νέων Τελεστών (π.χ. **.prime.**, **.polar.**)
   - Υπερφόρτωση Τελεστών (π.χ. **+**, **-**, *****, **/**)
   - Υπερφόρτωση Τελεστή Ανάθεσης (**=**)

Γενικές Διαδικασίες

ανταλλαγή τιμών I.

```
module swap_module
    implicit none; private    ! Όλα είναι ιδιωτικά

    interface swap ! Ένα όνομα εκπροσωπεί 2 διαδικασίες
        module procedure swap_reals, swap_integers
    end interface

    public :: swap            ! Το swap είναι δημόσιο

contains

    subroutine swap_reals(a, b)
        real, intent(inout)  :: a, b
        real                 :: temp
        temp = a
           a = b
           b = temp
    end subroutine swap_reals
```

Γενικές Διαδικασίες

ανταλλαγή τιμών II.

```
    subroutine swap_integers(a, b)
       integer, intent(inout)  :: a, b
       integer                 :: temp
       temp = a
          a = b
          b = temp
    end subroutine swap_integers
end module swap_module

program test_swap ! Κυρίως Πρόγραμμα
    use swap_module
    implicit none
    real    :: x, y
    integer :: i, j
    x = 1.1;   y = 2.2
    i = 1;     j = 2
    call swap(x, y); print *, x, y
    call swap(i, j); print *, i, j
end program test_swap
```

Γενικές Διαδικασίες

ανταλλαγή τιμών πινάκων III.

```
contains

  elemental subroutine swap_reals(a, b)
        real, intent(inout)  :: a, b
        real                 :: temp
        temp = a
           a = b
           b = temp
    end subroutine swap_reals

  elemental subroutine swap_integers(a, b)
     integer, intent(inout)  :: a, b
     integer                 :: temp
     temp = a
        a = b
        b = temp
  end subroutine swap_integers

end module swap_module
```

Να φτιάξουμε ένα δικό μας τελεστή

πρώτοι αριθμοί I.

```
module primer
    private                              ! Όλα είναι ιδιωτικά

        interface operator(.prime.)
            module procedure is_it_a_prime
        end interface

    public operator(.prime.)     ! Ο τελεστής είναι δημόσιος

contains

    function is_it_a_prime(n) result(test)
        integer,intent(in):: n
        logical           :: test
        integer           :: nt, p, pmax

        nt = abs(n)
```

Να φτιάξουμε ένα δικό μας τελεστή

πρώτοι αριθμοί II.

```
        select case(nt)
       case(3:)
            pmax = sqrt(real(nt))
           do p = 2, pmax
               if(mod(nt, p) == 0) then
                   test = .false.
                   exit
              else
                   test = .true.
               end if
           end do
       case (2)
            test = .true.
       end select
    end function is_it_a_prime
end module primer
```

Να φτιάξουμε ένα δικό μας τελεστή

πρώτοι αριθμοί III.

```
program prime_tester
    use primer
    implicit none
    integer  :: n
    logical  :: query

    do
        print *,'Input integer to be tested'
        read *, n;  if (n <= 1) exit
        query = .prime.n   ! κλήση της διαδικασίας
                           ! με χρήση του τελεστή .prime.
        select case (query)
            case(.true.)
                print *, 'This is a prime number :)'
            case default
                print *, 'This number is NOT a prime :('
        end select
    end do
    print *, 'Bye Bye'
end program prime_tester
```

Προσαρμογή Εκθετικής Συνάρτησης

- Για να προσαρμόσουμε συνάρτηση της μορφής $y = A \cdot e^{B \cdot x}$,
- Λογαριθμίζουμε τα δύο μέλη και προκύπτει $\ln y = \ln A + B \cdot x$, στην οποία μπορούμε να προσαρμόσουμε ευθεία ελαχίστων τετραγώνων κατά τα γνωστά.
- Οπότε:

$$a = \frac{\sum_{i=1}^{n} \ln y_i \sum_{i=1}^{n} x_i^2 - \sum_{i=1}^{n} x_i \sum_{i=1}^{n} x_i \ln y_i}{n \sum_{i=1}^{n} x_i^2 - \left(\sum_{i=1}^{n} x_i\right)^2}$$

$$b = \frac{n \sum_{i=1}^{n} x_i \ln y_i - \sum_{i=1}^{n} x_i \sum_{i=1}^{n} \ln y_i}{n \sum_{i=1}^{n} x_i^2 - \left(\sum_{i=1}^{n} x_i\right)^2}$$

- Όπου: $B \equiv b$ και $A \equiv \exp(a)$

Προσαρμογή Εκθετικής Συνάρτησης

II.

```
module Tfit
   implicit none
   private

   interface operator (.FtoC.)
      module procedure Farenheit_to_Celsius
   end interface

   interface operator (.CtoF.)
      module procedure Celsius_to_Farenheit
   end interface

   public operator(.FtoC.), operator(.CtoF.), expfit

contains
```

Προσαρμογή Εκθετικής Συνάρτησης

III.

```
elemental real function Farenheit_to_Celsius(F) result(C)
   real, intent(in):: F
   C = (F - 32) * 5. / 9.
end function Farenheit_to_Celsius

elemental real function Celsius_to_Farenheit(C) result(F)
   real, intent(in):: C
   F = C * 9. / 5. + 32
end function Celsius_to_Farenheit

function expfit(x,y) result(z)
 real, intent(in), dimension(:) :: x, y
 real, dimension(size(x))       :: z
 real                           :: a, b
 integer                        :: n

 n = size(x)
```

Προσαρμογή Εκθετικής Συνάρτησης

IV.

$$a = \frac{\sum_{i=1}^{n} \ln y_i \sum_{i=1}^{n} x_i^2 - \sum_{i=1}^{n} x_i \sum_{i=1}^{n} x_i \ln y_i}{n \sum_{i=1}^{n} x_i^2 - \left(\sum_{i=1}^{n} x_i\right)^2}$$

```
a = (sum(log(y)) * sum(x * x) - sum(x) * sum (x *log(y)))&
    &/ (n * sum(x * x) - (sum(x)* sum(x)))
```

$$b = \frac{n \sum_{i=1}^{n} x_i \ln y_i - \sum_{i=1}^{n} x_i \sum_{i=1}^{n} \ln y_i}{n \sum_{i=1}^{n} x_i^2 - \left(\sum_{i=1}^{n} x_i\right)^2}$$

```
b = (n * sum(x* log(y)) - sum(x) * sum (log(y)))&
    &/ (n * sum(x * x) - (sum(x)* sum(x)))
```

```
a = exp(a)
```

$A \equiv \exp(a)$

```
z = a * exp(b * x) ! επιστρέφει πίνακα

end function expfit

end module Tfit
```

Προσαρμογή Εκθετικής Συνάρτησης

V.

```
program temperatures
  use Tfit; implicit none

  integer, parameter :: n = 12
  real::TF(n)=[200,192,185,179,174,169,151,138,126,117,106,84]
  real::t(n)=[0,1,2,3,4,5,10,15,20,25,30,60]
  real::TC(n), Texp(n); integer::i

  TC = .FtoC. TF; print '(*(f5.1,:," "))', TC
  TC = TC - 25

  Texp = expfit(t, TC); print '(*(f5.1,:," "))', Texp

  open (1, file = 'fitting.txt')
    do i = 1, n
      write (1,*) t(i), TC(i), Texp(i)
    enddo
  close(1)

end program temperatures
```

Υπερφόρτωση (overloading) εγγενούς συνάρτησης

I.

```
module integer_sqrt_module
    implicit none
    private

      interface sqrt
        module procedure sqrt_int
      end interface

    public sqrt

contains

    elemental function sqrt_int(i)
        integer, intent(in):: i
        integer             :: sqrt_int
        sqrt_int = int(sqrt(real(i) + 0.5))
    end function sqrt_int

end module integer_sqrt_module
```

Υπερφόρτωση (overloading) εγγενούς συνάρτησης

II.

```
program test_integer_sqrt
    use integer_sqrt_module
    implicit none

    integer, parameter    :: m = 16
    integer               :: i
    integer, dimension(m):: n = [(i, i = 1, size(n))]
    character(len=40)     :: using_form

    write (using_form, *) '(', m, 'i3)'
    print  using_form, n, sqrt(n)

end program test_integer_sqrt
```

```
  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16
  1  1  1  2  2  2  2  2  3  3  3  3  3  3  3  4
```

Υπερφόρτωση (overloading)

I.

```
module parameters
   type time                              ! χρόνος
      real :: seconds
   end type time
   type velocity                          ! ταχύτητα
      real :: metres_per_second
   end type velocity
   type length                            ! μήκος
      real :: metres
   end type length
   type area                              ! επιφάνεια
      real :: metres_squared
   end type length_squared
   interface operator(/) ! υπερφόρτωση τελεστή
      module procedure length_by_time
   end interface
   interface sqrt          ! υπερφόρτωση εγγενούς συνάρτησης
      module procedure sqrt_metres_squared
   end interface
contains
```

Υπερφόρτωση (overloading)

II.

```
    function length_by_time(s, t) ! υπερφόρτωση τελεστή '/'
       type(length), intent(in) :: s
       type(time),   intent(in) :: t
       type(velocity)           :: length_by_time

       length_by_time%metres_per_second = s%metres/t%seconds
    end function length_by_time

    function sqrt_metres_squared(L2) ! υπερφόρτωση sqrt()
       type(area), intent(in) :: L2
       type(length)           :: sqrt_metres_squared

       sqrt_metres_squared%metres = sqrt(L2%metres_squared)
    end function sqrt_metres_squared

end module parameters
```

Υπερφόρτωση (overloading)

III.

```
program test
   use parameters
   implicit none
   type(length)           :: s  = length(3.0), L
   type(length_squared)   :: s2 = length_squared(9.0)
   type(velocity)         :: v
   type(time)             :: t = time(2.0)

   v = s / t
   ! Note: v = s + t or v = s * t would be illegal
   ! Error: Operands of binary numeric operator '+'
   ! are TYPE(length)/TYPE(time)

   L = sqrt(s2)
   print '(a8,f8.5)','speed=',v, 'time=', t, 'length=', L

end program test
```

Σημείωμα Αναφοράς

Copyright Πανεπιστήμιο Πατρών, Όνομα μέλους ή μελών ΔΕΠ 2014: Δημήτριος Ματαράς. «Εισαγωγή στον Προγραμματισμό Η/Υ». Έκδοση: 1.0. Πάτρα 2014. Διαθέσιμο από τη δικτυακή διεύθυνση: https://eclass.upatras.gr/courses/CMNG2178.

Χρηματοδότηση

- Το παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στο πλαίσιο του εκπαιδευτικού έργου του διδάσκοντα.
- Το έργο «**Ανοικτά Ακαδημαϊκά Μαθήματα στο Πανεπιστήμιο Αθηνών**» έχει χρηματοδοτήσει μόνο την αναδιαμόρφωση του εκπαιδευτικού υλικού.
- Το έργο υλοποιείται στο πλαίσιο του Επιχειρησιακού Προγράμματος «Εκπαίδευση και Δια Βίου Μάθηση» και συγχρηματοδοτείται από την Ευρωπαϊκή Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο) και από εθνικούς πόρους.

Σημείωμα Αδειοδότησης

Το παρόν υλικό διατίθεται με τους όρους της άδειας χρήσης Creative Commons Αναφορά, Μη Εμπορική Χρήση Παρόμοια Διανομή 4.0 [1] ή μεταγενέστερη, Διεθνής Έκδοση. Εξαιρούνται τα αυτοτελή έργα τρίτων π.χ. φωτογραφίες, διαγράμματα κ.λ.π., τα οποία εμπεριέχονται σε αυτό και τα οποία αναφέρονται μαζί με τους όρους χρήσης τους στο «Σημείωμα Χρήσης Έργων Τρίτων».

[1] http://creativecommons.org/licenses/by-nc-sa/4.0/

Ως **Μη Εμπορική** ορίζεται η χρήση:

- που δεν περιλαμβάνει άμεσο ή έμμεσο οικονομικό όφελος από την χρήση του έργου, για το διανομέα του έργου και αδειοδόχο
- που δεν περιλαμβάνει οικονομική συναλλαγή ως προϋπόθεση για τη χρήση ή πρόσβαση στο έργο
- που δεν προσπορίζει στο διανομέα του έργου και αδειοδόχο έμμεσο οικονομικό όφελος (π.χ. διαφημίσεις) από την προβολή του έργου σε διαδικτυακό τόπο

Ο δικαιούχος μπορεί να παρέχει στον αδειοδόχο ξεχωριστή άδεια να χρησιμοποιεί το έργο για εμπορική χρήση, εφόσον αυτό του ζητηθεί.