Ελληνική Δημοκρατία
Τεχνολογικό Εκπαιδευτικό
Ίδρυμα Ηπείρου

# Προγραμματισμός Ι

## **Ενότητα 6 :** Υποπρογράμματα ΙΙΙ

Αλέξανδρος Τζάλλας

Τμήμα Μηχανικών Πληροφορικής Τ.Ε

# Προγραμματισμός Ι

## Ενότητα 6 : Υποπρογράμματα ΙΙΙ

Αλέξανδρος Τζάλλας
Λέκτορας
Άρτα, 2015

# Άδειες Χρήσης

- Το παρόν εκπαιδευτικό υλικό υπόκειται σε άδειες χρήσης Creative Commons.
- Για εκπαιδευτικό υλικό, όπως εικόνες, που υπόκειται σε άλλου τύπου άδειας χρήσης, η άδεια χρήσης αναφέρεται ρητώς.

# Χρηματοδότηση

- 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.
- Το έργο «**Ανοικτά Ακαδημαϊκά Μαθήματα στο ΤΕΙ Ηπείρου**» έχει χρηματοδοτήσει μόνο τη αναδιαμόρφωση του εκπαιδευτικού υλικού.
- Το παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στα πλαίσια του εκπαιδευτικού έργου του διδάσκοντα.

Ευρωπαϊκή Ένωση
Ευρωπαϊκό Κοινωνικό Ταμείο
ΕΠΙΧΕΙΡΗΣΙΑΚΟ ΠΡΟΓΡΑΜΜΑ
ΕΚΠΑΙΔΕΥΣΗ ΚΑΙ ΔΙΑ ΒΙΟΥ ΜΑΘΗΣΗ
επένδυση στην κοινωνία της γνώσης
ΥΠΟΥΡΓΕΙΟ ΠΑΙΔΕΙΑΣ & ΘΡΗΣΚΕΥΜΑΤΩΝ, ΠΟΛΙΤΙΣΜΟΥ & ΑΘΛΗΤΙΣΜΟΥ
ΕΙΔΙΚΗ ΥΠΗΡΕΣΙΑ ΔΙΑΧΕΙΡΙΣΗΣ
ΕΣΠΑ 2007-2013
πρόγραμμα για την ανάπτυξη
ΕΥΡΩΠΑΪΚΟ ΚΟΙΝΩΝΙΚΟ ΤΑΜΕΙΟ
Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης

# Σκοποί ενότητας

- Να γίνει κατανοητή η έννοια των αναδρομικών υποπρογραμμάτων.
- Να περιγραφούν οι συντακτικοί κανόνες των αναδρομικών υποπρογραμμάτων.
- Να περιγραφούν οι συντακτικοί κανόνες των φωλιασμένων υποπρογραμμάτων.
- Να γίνει σαφές πως μπορούμε να χρησιμοποιήσουμε τα υποπρογράμματα σαν παραμέτρους.

# Περιεχόμενα ενότητας

- Αναδρομικά Υποπρογράμματα.
- Αναδρομή & Επανάληψη
- Φωλιασμένα Υποπρογράμματα
- Διαδικασίες & Συναρτήσεις σαν Παράμετροι

# Αναδρομικά Υποπρογράμματα$_{1/6}$

- Ένας ορισμός είναι αναδρομικός (recursive) όταν ορίζεται σαν συνάρτηση του εαυτού του
- Για παράδειγμα οι ακόλουθοι ορισμοί είναι αναδρομικοί
  - Ο Ν είναι φυσικός αριθμός αν ο Ν-1 είναι φυσικός αριθμός
  - Παραγοντικό: $0!=1$ για $n>0$, $n!=n.(n-1)!$
  - Δύναμη: $\alpha^1=\alpha$ για $n>0$, $\alpha^n=\alpha.\alpha^{n-1}$
  - Μερικό άθροισμα: $S(1)=1$ για $n>1$, $S(n)=S(n-1)+n$

# Αναδρομικά Υποπρογράμματα$_{2/6}$

- Στην Pascal μια διαδικασία ή συνάρτηση μπορεί να καλέσει όχι μόνο μια άλλη διαδικασία ή συνάρτηση αλλά και τον εαυτό της
- Ένα τέτοιο υποπρόγραμμα (διαδικασία ή συνάρτηση) που καλεί τον εαυτό του, λέγεται αναδρομικό

# Αναδρομικά Υποπρογράμματα 3/6

Να δοθεί ο αναδρομικός ορισμός της συνάρτησης που υπολογίζει τη δύναμη με ακέραιο εκθέτη. Ο αναδρομικός αλγόριθμος για τον υπολογισμό του $a^n$ είναι: Αν n=1 τότε $a^n=a$ αλλιώς $a^n=a*a^{n-1}$

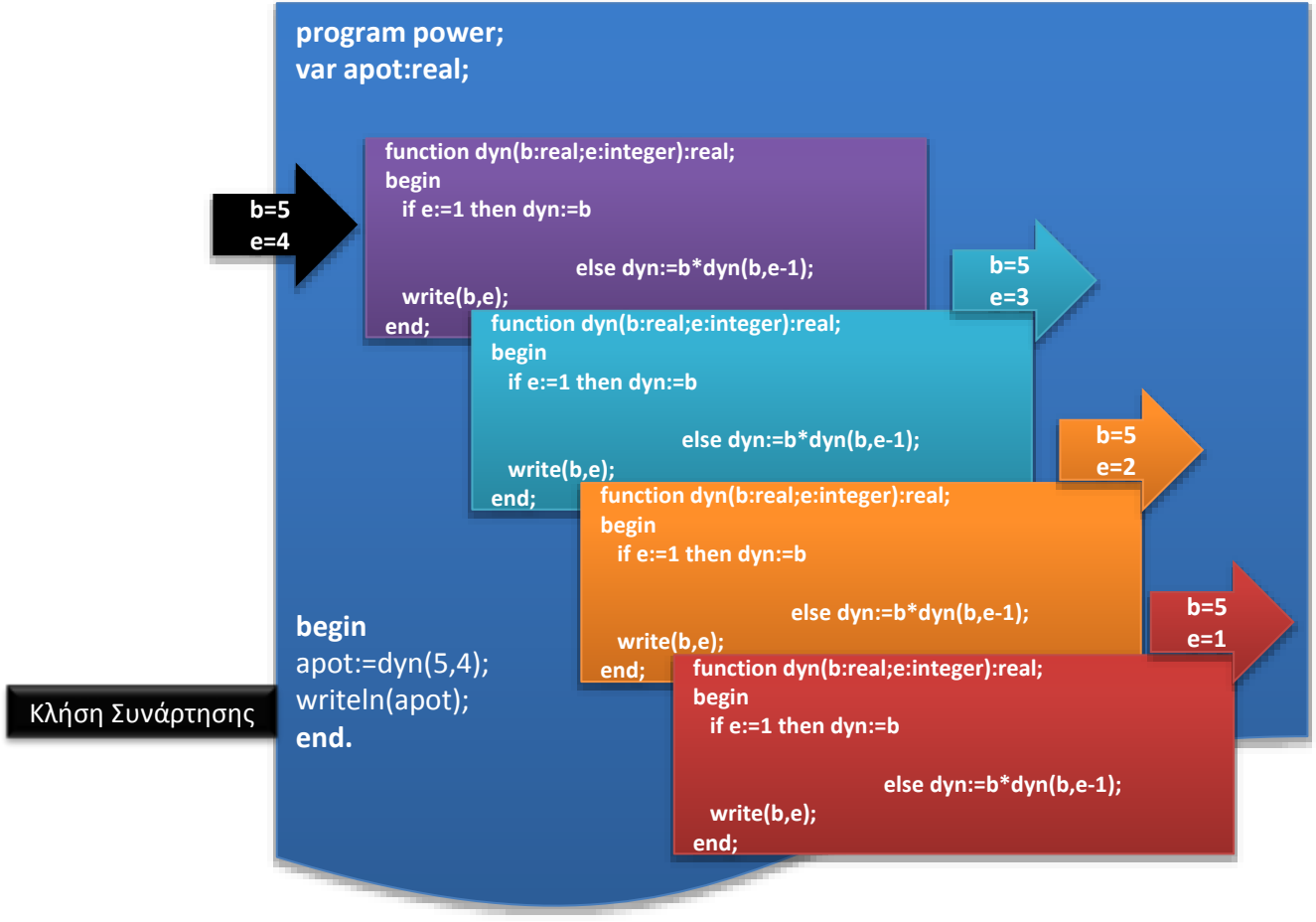
```
function dyn(basi: real; ekthetis: integer):real;
begin
  if ekthetis=1 then dyn:=basi      else
            dyn:=basi*dyn(basi,ethetis-1);
             (1)                    (2)
end;
```

- Σημειώνουμε τη διαφορά της αναφοράς της συνάρτησης στο αριστερό (σημείο 1) και δεξιό (σημείο 2) της πρότασης
- Στο αριστερό μέρος αναφέρεται μόνο το όνομα της συνάρτησης γιατί απλά δηλώνεται που θα γίνει ανάθεση της τιμής, η οποία θα υπολογισθεί στο δεξιό μέρος της πρότασης
- Στο δεξιό μέρος της πρότασης γίνεται κλήση της συνάρτησης, συνεπώς το όνομα να συνοδεύεται από μια λίστα πραγματικών παραμέτρων

# Αναδρομικά Υποπρογράμματα$_{4/6}$

- Όταν ένα υποπρόγραμμα (διαδικασία ή συνάρτηση) ‘Α’ καλεί ένα άλλο υποπρόγραμμα ‘Β’ ή τον εαυτό του, οι τιμές των τοπικών του δηλώσεων και παραμέτρων του αποθηκεύονται σε μια περιοχή της μνήμης που ονομάζεται στοίβα (stack)
- Όταν η εκτέλεση του υποπρογράμματος που κλήθηκε τελειώσει, ανασύρονται από τη στοίβα οι τιμές των δηλώσεων και παραμέτρων του αρχικού υποπρογράμματος ‘Α’ και η εκτέλεση συνεχίζει από το σημείο που σταμάτησε

# Αναδρομικά Υποπρογράμματα 5/6



# Αναδρομικά Υποπρογράμματα 6/6

Να ορισθεί αναδρομική συνάρτηση για τον υπολογισμό του ΜΚΔ δύο αριθμών. Ο αναδρομικός ορισμός του ΜΚΔ είναι: ΜΚΔ(α,β)=ΜΚΔ(α-β,β) αν α>β ή ΜΚΔ(α,β-α) ανβ>α

```
function mkd(a,b:integer):integer;
begin
if a=b then mkd:=a;
        else if a>b then mkd:=mkd(a-b,b)
        else mkd:=mkd(a,b-a)
end;
```

# Αναδρομή & Επανάληψη$_{1/3}$

- ✓Ένας αναδρομικός αλγόριθμος μπορεί να υλοποιηθεί εξίσου αποδοτικά και με ένα μη αναδρομικό υποπρόγραμμα χρησιμοποιώντας εντολές επανάληψης
- ✓Ο δεύτερος αυτός τρόπος ονομάζεται επαναληπτικός (iterative)

# Αναδρομή & Επανάληψη$_{2/3}$

- Να ορισθεί συνάρτηση για τον ορισμό του αθροίσματος Ν ακεραίων αριθμών

```
function sum(N:integer):integer;
begin
if N=0 then sum:=0;
      else sum:=N+sum(N-1);
end;
```

***Αναδρομική Μέθοδος***

```
function sum(N:integer):integer;
var i,s:integer
begin
s:=0;
for i:=1 to N do s:=s+i;
sum:=s;
end;
```

***Επαναληπτική Μέθοδος***

# Αναδρομή & Επανάληψη$_{3/3}$

- Όταν μια εφαρμογή μπορεί να υλοποιηθεί και με τις δύο μεθόδους, η επιλογή μας στηρίζεται στις δύο ακόλουθες παρατηρήσεις:
  - Τα αναδρομικά υποπρογράμματα έχουν σημαντικά μικρότερο μέγεθος και είναι περισσότερο ευανάγνωστα από τα αντίστοιχα επαναληπτικά
  - Τα αναδρομικά υποπρογράμματα έχουν μεγάλες απαιτήσεις μνήμης

# Φωλιασμένα Υποπρογράμματα$_{1/2}$

- Κάθε υποπρόγραμμα (διαδικασία ή συνάρτηση) μπορεί να περιέχει οποιαδήποτε αριθμό άλλων υποπρογραμμάτων
- Τα υποπρογράμματα που περιέχονται μέσα στα άλλα λέγονται φωλιασμένα

# Φωλιασμένα Υποπρογράμματα 2/2

Να ορισθεί συνάρτηση για τον υπολογισμό της σειράς:

$$\sum_{i=1}^{N}\sum_{j=1}^{i} j^{j}$$

δηλαδή του αθροίσματος:

$$1^1 + (1^1 + 2^2) + (1^1 + 2^2 + 3^3) + \ldots + (1^1 + 2^2 + \ldots + N^N)$$

*<u>Υπόδειξη:</u> Να ορισθούν συνολικά τρείς συναρτήσεις. Για τη δύναμη, το μερικό άθροισμα και για το συνολικό άθροισμα

```
program athroisma;
var N,apot:integer;
{.....
.....}
function sum(k:integer):integer;
var x,ath:integer;
{.....
.....}
function dyn(basi,ekth:integer):integer;
var t,d:integer;
begin
d:=1;
for t:=1 to ekth do d:=d*basi;
dyn:=d;
end;
{.....}
function meriko(m:integer):integer;
var x,s:integer;
begin
s:=0;
for x:=1 to m do s:=s+dyn(x,x);
meriko:=s;
end; {.....}
```

```
begin {Σώμα της συνάρτησης sum}
ath:=1;
for x:=1 to k do ath:=ath+meriko(x);
sum:=ath;
end;
{.....
.....}
begin {Κύριο Πρόγραμμα}
write('Dwse to N:');
readln(N);
apot:=sum(N);
writeln('Athroisma seiras=',apot);
end.
```

# Διαδικασίες & Συναρτήσεις σαν Παράμετροι$_{1/4}$

- Μια παράμετρος τύπου προγράμματος, στην τυπική λίστα παραμέτρων ενός υποπρογράμματος, έχει ίδια σύνταξη με την επικεφαλίδα του υποπρογράμματος:
  - **procedure** A(x:integer; procedure P);

    Η διαδικασία Α έχει δύο παραμέτρους, την x τύπου integer και την διαδικαστική παραμέτρου P που δεν έχει επιπλέον παραμέτρους

# Διαδικασίες & Συναρτήσεις σαν Παράμετροι 2/4

- **procedure** Q(function F(c:char):integer; y:real);

  Η διαδικασία Q έχει δύο παραμέτρους, μια τύπου real και μια συναρτησιακή παραμέτρου F, η οποία επιπλέον έχει μια παράμετρο τύπου char

# Διαδικασίες & Συναρτήσεις σαν Παράμετροι$_{3/4}$

- Κατά την κλήση ενός υποπρογράμματος με διαδικαστική ή συναρτησιακή παράμετρο, πρέπει η αντίστοιχη πραγματική παράμετρος να είναι το όνομα μια διαδικασίας ή συνάρτησης με τη σωστή αντιστοιχία παραμέτρων και τύπου αποτελέσματος (αν πρόκειται για συνάρτηση)

# Διαδικασίες & Συναρτήσεις σαν Παράμετροι 4/4

Να ορισθεί μια διαδικασία που να υπολογίζει τις τιμές μιας οποιαδήποτε συνάρτησης πραγματικών τιμών, με δεδομένα τα όρια και το βήμα μεταβολής της παραμέτρου της συνάρτησης

```
procedure eval(function
f(x:real):real,lower,upper,step:real);
var x,value:real;
begin
x:=lower;
while x<=upper do begin
                        value:=f(x);
                        writeln(x:value);
                        x:=x+step;
                        end
end;
```

# Βιβλιογραφία

Βλαχάβας Ι. (1994). Η γλώσσα προγραμματισμού Pascal. Εκδόσεις Γαρταγάνης Διονύσιος.

Κάβουρας Ι.Κ. (1999). Δομημένος Προγραμματισμός με Pascal. Εκδόσεις Κλειδάριθμος.

Αλεβίζου Θ., & Καμπουρέλης Α. (1995). Μαθήματα Προγραμματισμού: Εισαγωγή με τη Γλώσσα Pascal. Εκδόσεις Παπασωτηρίου.

Cooper D. (1993). Oh! Pascal!, An Introduction to Computing, του. Εκδόσεις Norton.

Larry R.N. (1998). Advanced Programming in Pascal with Data Structures. Εκδόσεις Macmillan USA.

Τσελίκης Γ.Σ., Τσελίκας Ν.Δ. (2012). C: από τη Θεωρία στην Εφαρμογή (Β' Έκδοση). Εκδόσεις Παπασωτηρίου.

Aho A.V., Hopcroft J.E., & Ullman J.D. (1974). The design and analysis of computer algorithms. Εκδόσεις Addison Wesley.

Abelson H., Sussman G.J., Sussman J. (1985). Structure and Interpretation of Computer Programs, MIT Press, McGraw Hill Book Company.

# Σημείωμα Αναφοράς

# Σημείωμα Αδειοδότησης



[1] http://creativecommons.org/licenses/by-nc-nd/4.0/deed.el

# Τέλος Ενότητας

**Επεξεργασία: Ευάγγελος Καρβούνης**

Άρτα, 2015

Ευρωπαϊκή Ένωση
Ευρωπαϊκό Κοινωνικό Ταμείο
ΕΠΙΧΕΙΡΗΣΙΑΚΟ ΠΡΟΓΡΑΜΜΑ ΕΚΠΑΙΔΕΥΣΗ ΚΑΙ ΔΙΑ ΒΙΟΥ ΜΑΘΗΣΗ
επένδυση στην κοινωνία της γνώσης
ΥΠΟΥΡΓΕΙΟ ΠΑΙΔΕΙΑΣ & ΘΡΗΣΚΕΥΜΑΤΩΝ, ΠΟΛΙΤΙΣΜΟΥ & ΑΘΛΗΤΙΣΜΟΥ
ΕΙΔΙΚΗ ΥΠΗΡΕΣΙΑ ΔΙΑΧΕΙΡΙΣΗΣ
ΕΣΠΑ 2007-2013
ΕΥΡΩΠΑΪΚΟ ΚΟΙΝΩΝΙΚΟ ΤΑΜΕΙΟ
Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης

# Τέλος Ενότητας

Υποπρογράμματα ΙΙΙ



Ευρωπαϊκή Ένωση
Ευρωπαϊκό Κοινωνικό Ταμείο
ΕΠΙΧΕΙΡΗΣΙΑΚΟ ΠΡΟΓΡΑΜΜΑ
ΕΚΠΑΙΔΕΥΣΗ ΚΑΙ ΔΙΑ ΒΙΟΥ ΜΑΘΗΣΗ
επένδυση στην κοινωνία της γνώσης
ΥΠΟΥΡΓΕΙΟ ΠΑΙΔΕΙΑΣ & ΘΡΗΣΚΕΥΜΑΤΩΝ, ΠΟΛΙΤΙΣΜΟΥ & ΑΘΛΗΤΙΣΜΟΥ
ΕΙΔΙΚΗ ΥΠΗΡΕΣΙΑ ΔΙΑΧΕΙΡΙΣΗΣ
Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης
ΕΣΠΑ 2007-2013
Πρόγραμμα για την ανάπτυξη
ΕΥΡΩΠΑΪΚΟ ΚΟΙΝΩΝΙΚΟ ΤΑΜΕΙΟ