Διάλεξη 7: Αναδρομή

Στην ενότητα αυτή θα μελετηθούν τα εξής επιμέρους θέματα:

Η έννοια της αναδρομής

Μη-αναδρομικός / Αναδρομικός Ορισμός Συναρτήσεων

Παραδείγματα Ανάδρομης: Παραγοντικό, Δύναμη, Αριθμοί Fibonacci

Αφαίρεση της Αναδρομής

Διδάσκων: Δημήτρης Ζεϊναλιπούρ

Μη-αναδρομικές συναρτήσεις

- Προτού δούμε τι είναι αναδρομή θεωρήστε το πρόβλημα εύρεσης του παραγοντικού κάποιου αριθμού (factorial).

  0! *= 1 ,* ***1!*** *= 1* ***2!*** *= 1x2 = 2* ***3!*** *= 1x2x3 = 6*

  4! *= 1x2x3x4=24* ***5!*** *= 1x2x3x4x5 = 120*

  Ζητούμενο: *Να υλοποιήσουμε την factorial(int n) η οποία μας επιστρέφει το παραγοντικό κάποιου θετικού ακεραίου n.*

  Λύση

```
int factorial(int n) {
        int i, result=1;
        for (i=1; i<=n; i++)  {
                result *= i;
        }
        return result;
}
```

Αναδρομή

- Βασική έννοια στα Μαθηματικά και στην Πληροφορική.

- Στην πληροφορική η αναδρομή χρησιμοποιείται σαν *τεχνική προγραμματισμού* και σαν *μέθοδος σχεδιασμού αλγορίθμων*.

- Στον προγραμματισμό η αναδρομή εμφανίζεται με την **κλήση ενός υποπρογράμματος από τον εαυτό του**. Ένα αναδρομικό υποπρόγραμμα αποτελείται από:
  - ένα **βήμα διακοπής**, όπου ορίζεται η εκτέλεση του υποπρογράμματος για κάποιες "μικρές "τιμές των παραμέτρων του, και
  - ένα **αναδρομικό βήμα**, κατά το οποίο η εκτέλεση του υποπρογράμματος ορίζεται ως συνδυασμός κλήσεων του υποπρογράμματος σε άλλες "μικρότερες" τιμές των παραμέτρων.

Παράδειγμα 2: Παραγοντικό Αριθμού

- Ας ορίσουμε τώρα τον αναδρομικό ορισμό της factorial.

0! *= 1 ,* ***1!*** *= 1* ***2!*** *= 1x2 = 2* ***3!*** *= 1x2x3 = 6*

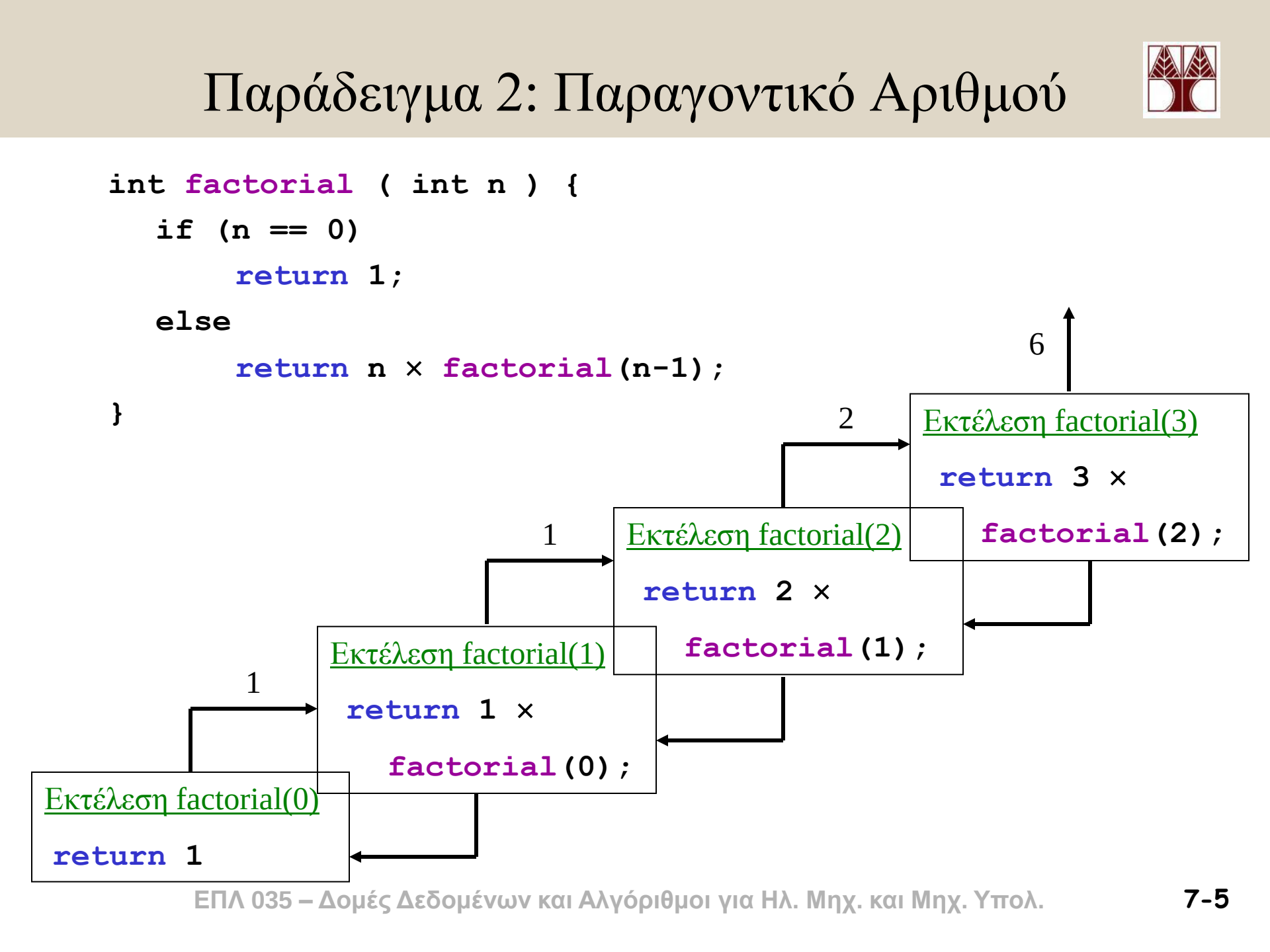
4! *= 1x2x3x4=24* ***5!*** *= 1x2x3x4x5 = 120 ……*

Αναδρομικός Ορισμός Factorial

$$0! = 1 \quad \text{(Βήμα Διακοπής)}$$

$$n! = n \times (n-1)! \ , \quad n > 0 \quad \text{(Αναδρομικό Βήμα)}$$

Παράδειγμα 2: Παραγοντικό Αριθμού



```
int factorial ( int n ) {
   if (n == 0)
        return 1;
   else
        return n × factorial(n-1);
}
```

Υλοποίηση αναδρομής

- Σε κάθε κλήση οποιασδήποτε συνάρτησης ένα σύνολο από λέξεις **(stack frame) φυλάσσεται σε μια στοίβα (την στοίβα του προγράμματος)**, από όπου μπορεί να ανασυρθεί.
- Όταν μια συνάρτηση διακόψει την εκτέλεσή της με την κλήση μιας άλλης συνάρτησης ***οι παράμετροι της συνάρτησης***, *η* ***διεύθυνση επιστροφής*** και *οι* ***τοπικές μεταβλητές*** της **καλούσας συνάρτησης** φυλάσσονται μέσα στη στοίβα του προγράμματος.
- Έτσι όταν η κληθείσα συνάρτηση τερματίσει το περιβάλλον την καλούσας συνάρτησης **ανασύρεται από τη στοίβα** για να συνεχιστεί κανονικά η εκτέλεσή της.
- Αφού κάθε κλήση μιας διαδικασίας εκτελείται στο δικό της περιβάλλον, είναι επιτρεπτή και **η κλήση συναρτήσεων από τον εαυτό** τους (αναδρομή).

Παράδειγμα 2: Δύναμη Αριθμού

2) Δύναμη (Power)

$a^0 = 1$

$a^n = a^{n-1}.a \quad (n>=1)$

```
int mpower(int a, int n) {
    if (n==0)       return 1;
    return a*mpower(a, n-1);
}
```

Παράδειγμα

mpower(2,3) => 2*mpower(2,2)
= 2*2*mpower(2,1)
= 2*2*2*mpower(2,0)
= 2*2*2*1= 8

Παράδειγμα 3: Fibonacci Numbers

3) Αριθμοί Fibonacci (Leonardo of Pisa – 1202μX)

Χρησιμοποιήθηκαν για να εκφράσουν την αύξηση κουνελιών!

- Στον μήνα 0 έχουμε 0 ζεύγη , και στον μήνα 1 έχουμε 1 ζεύγος (η Γένεσης!)
- Το ζεύγος γονιμοποιείται μετά τον πρώτο μήνα (δηλ. στον δεύτερο).
- Κάθε μήνα, Κάθε ζεύγος παράγει ένα νέο ζεύγος
- Έστω ότι είμαστε στον μήνα **n** και έχουμε ένα πληθυσμό **F(n) ζευγών**. Αυτή την στιγμή μόνο κουνέλια που ήταν ζωντανά την στιγμή **n-2** παράγουν ένα νέο ζεύγος.
- Επομένως **F(n-2)** ζευγάρια προστίθεται στον παρόν πληθυσμό των F(n-1).
- Ο ολικός πληθυσμός την στιγμή F(n) είναι επομένως **F(n) = F(n-1) + F(n-2)**

0,1,1,2,3,5,8,13,21,34,55,89,….

$$F_n := F(n) := \begin{cases} 0 & \text{if } n = 0; \\ 1 & \text{if } n = 1; \\ F(n-1) + F(n-2) & \text{if } n > 1. \end{cases}$$

```
int fibonacci(int a) {
    if (a==0)       return 0;
    else if (a==1)          return 1;
    return fibonacci(a-1) + fibonacci(a-2);
}
```

Αφαίρεση της Αναδρομής

- Η χρήση της αναδρομής επιτρέπει την επίλυση πολύπλοκων προβλημάτων με **άμεσο και σαφή τρόπο**. Συχνά όμως **υστερεί από άποψη αποδοτικότητας.**

- Η **αφαίρεση της αναδρομής από μια συνάρτηση**, δηλαδή, η μετατροπή της σε επαναληπτική συνάρτηση χωρίς αναδρομή, είναι δυνατή (κάτω από κάποιες συνθήκες).

- Συχνά προϋποθέτει τη χρήση κάποιων βοηθητικών δομών (π.χ. στοίβα ή ουρά – θα μελετηθούν στην συνέχεια του μαθήματος).

- Επίσης στις περισσότερες περιπτώσεις μπορούμε να επιλύσουμε μια αναδρομική εξίσωση και να βρούμε την λύση της σε κλειστή μορφή (με την χρήση Αναδρομικών Σχέσεων – Recurrence Relations).

  Π.χ. η λύση της εξίσωσης Fibonacci είναι :

$$f_n = \frac{1}{\sqrt{5}}\left(\frac{1+\sqrt{5}}{2}\right)^n - \frac{1}{\sqrt{5}}\left(\frac{1-\sqrt{5}}{2}\right)^n$$