Αρχές Γλωσσών Προγραμματισμού και Μεταφραστών

Εαρινό Εξάμηνο 2021

Ασκήσεις

Σύνολο 2

Εμβέλεια – Διαχείριση Μνήμης – Υποπρογράμματα

Γ. Γαροφαλάκης

ΑΣΚΗΣΗ 1

Δίνεται το παρακάτω πρόγραμμα σε μια γλώσσα τύπου Pascal, στην οποία η μεταβίβαση παραμέτρων γίνεται *με αναφορά* (*call by reference*).

I. Για την περίπτωση που ισχύει *στατικός κανόνας εμβέλειας*, ζητούνται:

a) Να ορίσετε τα περιβάλλοντα αναφοράς (τοπικά, μη-τοπικά και καθολικά), όλων των τμημάτων του προγράμματος.

b) Να παρουσιάσετε τα αποτελέσματα που θα εμφανιστούν στην οθόνη του υπολογιστή κατά την εκτέλεση του προγράμματος (με τη σωστή, χρονικά, σειρά). Παρουσιάστε και εξηγείστε τις αλλαγές τιμών όλων των μεταβλητών κατά τη διαδικασία υπολογισμού.

II. Να απαντήσετε στα παραπάνω ερωτήματα (a) και (b), υποθέτοντας ότι η γλώσσα ακολουθεί *δυναμικό κανόνα εμβέλειας*.

```
program  A;
   var K, L: integer;

   procedure SUB1 (var J: integer);
         begin
            J:= J+1;
            print("J=", J);
         end;

   procedure SUB2;
         begin
              SUB1(K);
              SUB1(L);
         end;

   procedure SUB3;
         var L: integer;
         begin
              L:= 8;
              SUB2;
         end;

   BEGIN
              K:= 3;
              L:= 4;
              SUB3;
              print("K=", K);
              print("L=", L);
   END.
```

ΑΠΑΝΤΗΣΕΙΣ ΑΣΚΗΣΗΣ 1
Ι – Στατικός Κανόνας Εμβέλειας

```
program  A;
   var K, L: integer;

   procedure SUB1 (var J: integer);
         begin
            J:= J+1;
            print("J=", J);
         end;

   procedure SUB2;
         begin
              SUB1(K);
              SUB1(L);
         end;

   procedure SUB3;
         var L: integer;
         begin
              L:= 8;
              SUB2;
         end;

   BEGIN
              K:= 3;
              L:= 4;
              SUB3;
              print("K=", K);
              print("L=", L);
   END.
```

a) Program A: - Τοπικό, Καθολικό ΠΑ : K, L, SUB1 (κλήση), SUB2 (κλήση), SUB3 (κλήση)

SUB1: - Τοπικό ΠΑ: J

- Μη-τοπικό και Καθολικό ΠΑ: K, L (από Α), SUB1 (κλήση), SUB2 (κλήση), SUB3 (κλήση)

SUB2: - Τοπικό ΠΑ: -

- Μη-τοπικό και Καθολικό ΠΑ: K, L (από Α), SUB1 (κλήση), SUB2 (κλήση), SUB3 (κλήση)

SUB3: - Τοπικό ΠΑ: L

- Μη-τοπικό και Καθολικό ΠΑ: K, SUB1 (κλήση), SUB2 (κλήση), SUB3 (κλήση)

b)

1. *Εκτέλεση του A*: K = 3, L_A = 4

2. *Κλήση της SUB3*: L_{SUB3} = 8

3. *Κλήση της SUB2*:

4. *Κλήση της SUB1(K)*: K ⟺ J = 3, J = J + 1 = 4, **K = J = 4**

   print: **J = 4**

5. *Κλήση της SUB1(L_A)*: L_A ⟺ J = 4, J = J + 1 = 5 **L_A = J = 5**

   print: **J = 5**

6. *Κλείσιμο των SUB2, SUB3*.

7. *Επιστροφή στο MAIN*:

   print: **K = 4**

   print: **L_A = 5**

```
program  A;
   var K, L: integer;

   procedure SUB1 (var J: integer);
         begin
            J:= J+1;
            print("J=", J);
         end;

   procedure SUB2;
         begin
              SUB1(K);
              SUB1(L);
         end;

   procedure SUB3;
         var L: integer;
         begin
              L:= 8;
              SUB2;
         end;

   BEGIN
              K:= 3;
              L:= 4;
              SUB3;
              print("K=", K);
              print("L=", L);
   END.
```

ΑΠΑΝΤΗΣΕΙΣ ΑΣΚΗΣΗΣ 1
ΙΙ – Δυναμικός Κανόνας Εμβέλειας

```
program  A;
   var K, L: integer;

   procedure SUB1 (var J: integer);
         begin
            J:= J+1;
            print("J=", J);
         end;

   procedure SUB2;
         begin
             SUB1(K);
             SUB1(L);
         end;

   procedure SUB3;
         var L: integer;
         begin
             L:= 8;
             SUB2;
         end;

   BEGIN
              K:= 3;
              L:= 4;
              SUB3;
              print("K=", K);
              print("L=", L);
   END.
```

a) Program A: - Τοπικό, Καθολικό ΠΑ : K, L, SUB1 (κλήση), SUB2 (κλήση), SUB3 (κλήση)

SUB1: - Τοπικό ΠΑ: J

- Μη-τοπικό ΠΑ: L (από SUB3), K, SUB1 (κλήση), SUB2 (κλήση), SUB3 (κλήση)
- Καθολικό ΠΑ: K, SUB1 (κλήση), SUB2 (κλήση), SUB3 (κλήση)

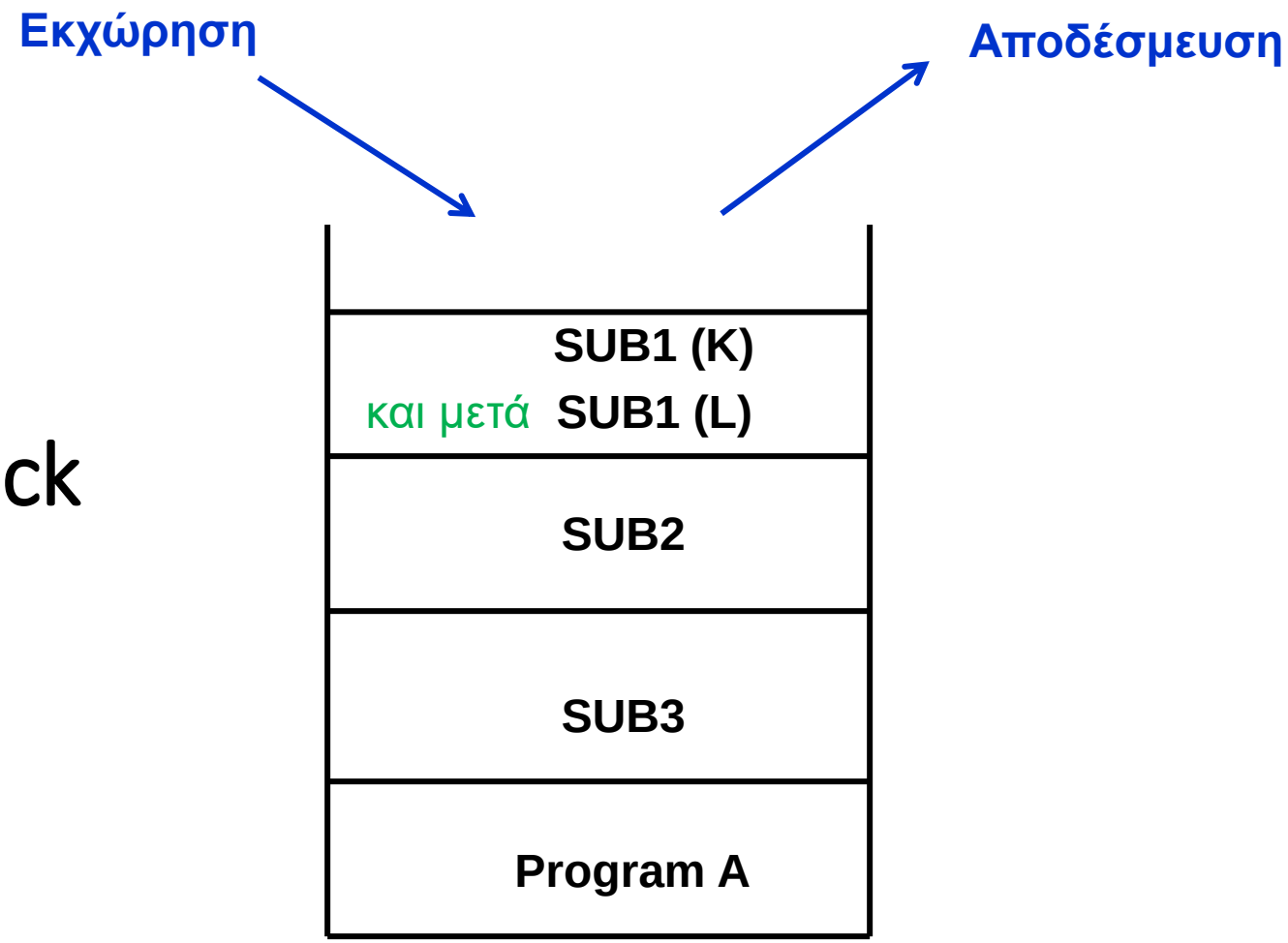
SUB2: - Τοπικό ΠΑ: -

- Μη-τοπικό ΠΑ: L (από SUB3), K, SUB1 (κλήση), SUB2 (κλήση), SUB3 (κλήση)
- Καθολικό ΠΑ: K, SUB1 (κλήση), SUB2 (κλήση), SUB3 (κλήση)

SUB3: - Τοπικό ΠΑ: L

- Μη-τοπικό και Καθολικό ΠΑ: K, SUB1 (κλήση), SUB2 (κλήση), SUB3 (κλήση)

Κεντρική run-time stack



b)

1. *Εκτέλεση του A*: K = 3, L_A = 4

2. *Κλήση της SUB3*: L_{SUB3} = 8

3. *Κλήση της SUB2*:

4. *Κλήση της SUB1(K)*: K ⟺ J = 3, J = J + 1 = 4, K = J = 4

   print: **J = 4**

5. *Κλήση της SUB1(L_{SUB3})*: L_{SUB3} ⟺ J = 8, J = J + 1 = 9

   L_{SUB3} = J = 9 print: **J = 9**

6. *Κλείσιμο των SUB2, SUB3*.

7. *Επιστροφή στο MAIN*: print: **K = 4**

   print: **L_A = 4**

```
program  A;
   var K, L: integer;

   procedure SUB1 (var J: integer);
         begin
            J:= J+1;
            print("J=", J);
         end;

   procedure SUB2;
         begin
              SUB1(K);
              SUB1(L);
         end;

   procedure SUB3;
         var L: integer;
         begin
              L:= 8;
              SUB2;
         end;

   BEGIN
              K:= 3;
              L:= 4;
              SUB3;
              print("K=", K);
              print("L=", L);
   END.
```

ΑΣΚΗΣΗ 2

Δίνεται το παρακάτω πρόγραμμα σε μια γλώσσα τύπου Pascal.

a) Ποια είναι τα Περιβάλλοντα Αναφοράς (Τοπικά, Μη-τοπικά, Καθολικά) όλων των τμημάτων του προγράμματος;

b) Στο πρόγραμμα υπάρχει μία εντολή η οποία δεν είναι σωστή από σημασιολογική άποψη και θα προκαλέσει run-time error. Ποια είναι αυτή και γιατί; Θεωρήστε για τη συνέχεια, ότι δεν υπάρχει η εντολή αυτή.

Τι τυπώνεται με την εκτέλεση του προγράμματος στις παρακάτω περιπτώσεις τρόπου μεταβίβασης παραμέτρων; Εξηγήστε σύντομα τις διαδικασίες υπολογισμού σε όλες τις περιπτώσεις.

c) Όλες οι παράμετροι μεταβιβάζονται με *κλήση με τιμή* (call by value).

d) Τα **a** και **b** μεταβιβάζονται με *κλήση με αναφορά* (call by reference) και το **c** με *κλήση με τιμή.*

e) Τα **a** και **b** μεταβιβάζονται με *κλήση με τιμή – αποτέλεσμα* (call by value-result) και το **c** με *κλήση με τιμή.*

f) Όλες οι παράμετροι μεταβιβάζονται με *κλήση με όνομα* (call by name).

```
program MAIN;
   var x, y: integer;
   procedure F(a, b, c: integer);
       begin
          b:= b + 5;
          b:= a + c + 4;
          write(a, b, c)
       end;
   BEGIN
       x:= 10;
       y:= 20;
       c:= 50;
       F(x, x, x + y);
       write(x, y)
   END.
```

ΑΠΑΝΤΗΣΕΙΣ ΑΣΚΗΣΗΣ 2

```
program MAIN;
   var x, y: integer;
   procedure F(a, b, c: integer);
        begin
           b:= b + 5;
           b:= a + c + 4;
           write(a, b, c)
        end;
   BEGIN
        x:= 10;
        y:= 20;
        c:= 50;
        F(x, x, x + y);
        write(x, y)
   END.
```

a) Program MAIN: - Τοπικό, Καθολικό ΠΑ : x, y, F (κλήση)

Procedure F: - Τοπικό ΠΑ: a, b, c

- Μη-τοπικό και Καθολικό ΠΑ: x, y, F (κλήση)

b) Η εντολή "`c:= 50;`" στη MAIN: Το **c** δεν είναι στο Περιβάλλον Αναφοράς της MAIN, δηλαδή δεν είναι ορατό σε αυτή.

c) Όλα by value:

1. *Εκτέλεση της MAIN*: x = 10, y = 20

2. *Κλήση της F*: (a = 10, b = 10, c = 30) b = b + 5 = 15, **b** = a + c + 4 **= 44**

write [a, b, c]: **10, 44, 30**

3. *Επιστροφή στη MAIN*: **write [x, y]:** **10, 20**

```
program MAIN;
   var x, y: integer;
   procedure F(a, b, c: integer);
       begin
           b:= b + 5;
           b:= a + c + 4;
           write(a, b, c)
       end;
   BEGIN
       x:= 10;
       y:= 20;
       c:= 50;
       F(x, x, x + y);
       write(x, y)
   END.
```

d) a, b by reference, c by value:

1. *Εκτέλεση της MAIN*: x = 10, y = 20

2. *Κλήση της F*: (x ⟷ a ⟷ b = 10, c = 30),

   b = b + 5 = **15** (= x = a), **b** = a+c+4 = 15+30+4 **= 49** (= x = a)

   write [a, b, c]: **49, 49, 30**

3. *Επιστροφή στη MAIN*: **write [x, y]:** **49, 20**

e) a, b by value-result, c by value:

1. *Εκτέλεση της MAIN*: x = 10, y = 20

2. *Κλήση της F*: (a = 10, b = 10, c = 30) b = b + 5 = 15, **b** = a + c + 4 **= 44**

   x = a = 10, x = b = 44

   write [a, b, c]: **10, 44, 30**

3. *Επιστροφή στη MAIN*: **write [x, y]:** **44, 20**

f) Όλα by name:

1. *Εκτέλεση της MAIN*: x = 10, y = 20

2. *Κλήση της F*:

b (↔ x) = b (↔ x) + 5 = 10 + 5 = **15** (↔ x = 15),

b (↔ x) = a (↔ x = 15) +c (↔ x + y = 15 + 20 = 35) +4 =

= 15 + 35 + 4 **= 54** (↔ x ↔ a = 54)

write [a, b, c (↔ x + y = 54 + 20 = 74)**]:** **54, 54, 74**

3. *Επιστροφή στη MAIN*: **write [x, y]:** **54, 20**

```
program MAIN;
   var x, y: integer;
   procedure F(a, b, c: integer);
       begin
          b:= b + 5;
          b:= a + c + 4;
          write(a, b, c)
       end;
   BEGIN
       x:= 10;
       y:= 20;
       c:= 50;
       F(x, x, x + y);
       write(x, y)
   END.
```