Εισαγωγή στον δομημένο προγραμματισμό

Ενότητα 3η: Διαμόρφωση Ελέγχου Ροής Προγράμματος

Αν. καθηγητής Στεργίου Κώστας
e-mail: kstergiou@uowm.gr

Τμήμα Μηχανικών Πληροφορικής και Τηλεπικοινωνιών

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 παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στα πλαίσια του εκπαιδευτικού έργου του διδάσκοντα.
- Το έργο «**Ανοικτά Ψηφιακά Μαθήματα στο Πανεπιστήμιο Δυτικής Μακεδονίας**» έχει χρηματοδοτήσει μόνο τη 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.

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

Στόχοι της διάλεξης

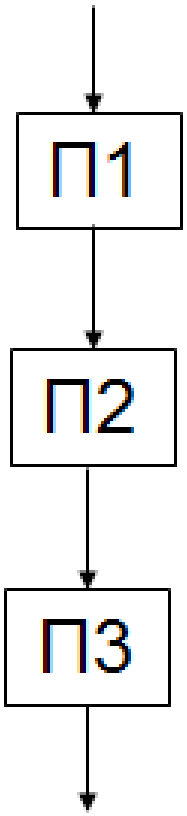
- Περιγραφή των δομών υπό συνθήκη διακλάδωσης και επανάληψης στη C.
- Εξοικείωση με απλά παραδείγματα χρήσης των δομών if...else, while, do…while, for.
- Κατανόηση της προτεραιότητας τελεστών σε εκφράσεις.
- Περιγραφή ειδικών εντολών διακλάδωσης .

Περιεχόμενα

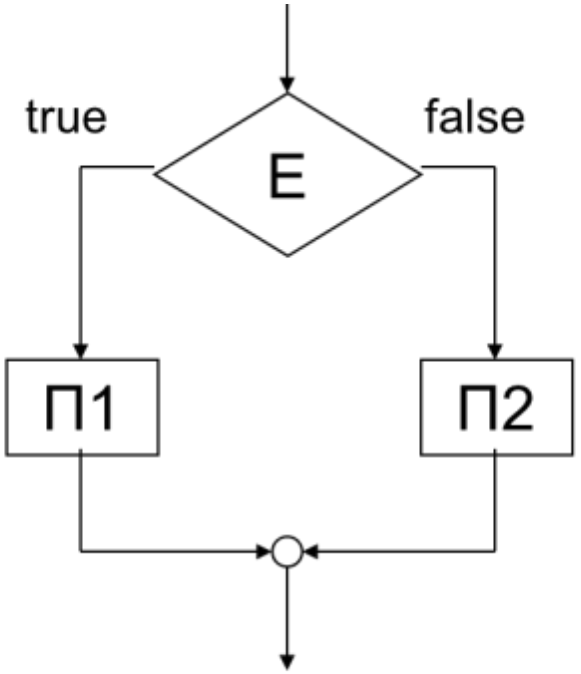
- Η δομή υπό συνθήκη διακλάδωσης if…else:
  - Εμφωλευμένες if.
- Οι δομές επανάληψης στη C:
  - while.
  - do…while.
  - for.
- Ειδικές εντολές διακλάδωσης:
  - break, continue, switch.

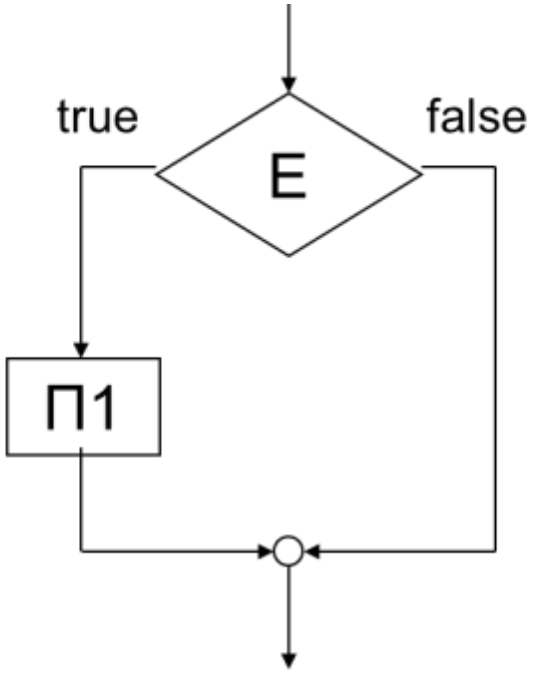
Δομημένος Προγραμματισμός (1/2)

- Ο πιο απλός και συνηθισμένος τρόπος εκτέλεσης μιας ακολουθίας εντολών είναι ο ακολουθιακός.
  - Η κάθε εντολή εκτελείται μετά από την άλλη όπως είναι γραμμένες στον κώδικα.
- Οι γλώσσες δομημένου προγραμματισμού επιτρέπουν πιο ευέλικτες δομές ελέγχου ροής του προγράμματος.
  - **Δομή υπό-συνθήκη διακλάδωσης.**
    - Συχνά θέλουμε να επιλέξουμε ποιο κομμάτι κώδικα θα εκτελεστεί .
  - **Δομή επανάληψης.**
    - Συχνά θέλουμε να επαναλάβουμε τα ίδια βήματα πολλές φορές.



Υπό-Συνθήκη Διακλάδωση





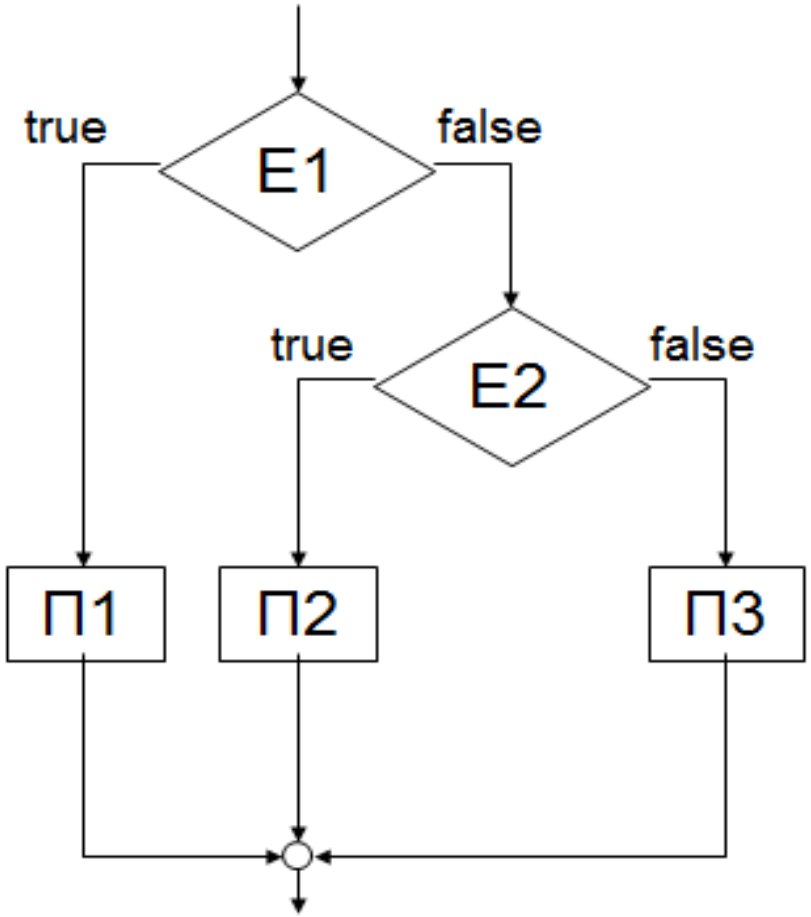
2 προτάσεις:

if E **then** Π1 **else** Π2

1 πρόταση:

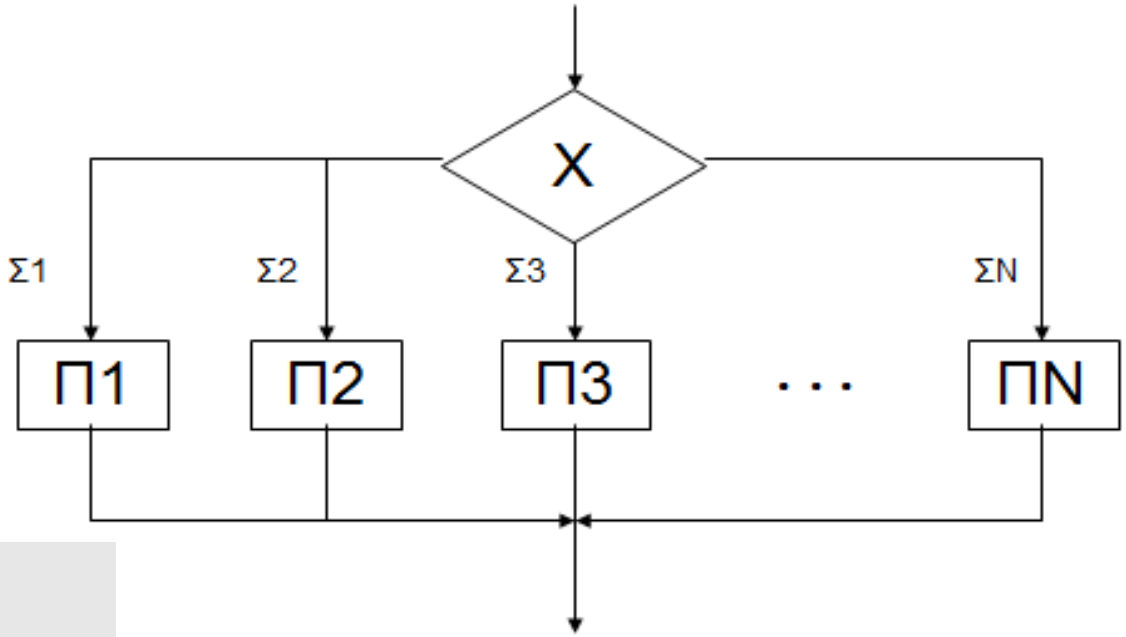
if E **then** Π1

Σύνθετη Δομή Επιλογής




```
if E1
    then Π1
else if E2
        then Π2
        else Π3
```

Δομή Επιλογής (Πολλαπλή)



if X==Σ1 **then** Π1
else if X==Σ2 **then** Π2
else if X==Σ3 **then** Π3
...
else if X==ΣN **then** ΠN

Προτάσεις Ελέγχου Ροής στη C (1/2)

- Διακλάδωση υπό συνθήκη:
  - if-else.
  - switch case.

Η Πρόταση if (1/2)

Η πρόταση μπορεί να περιέχει μια ή περισσότερες εντολές.

```
if (<συνθήκη>)
      <πρόταση1>
[else <πρόταση2>]
```

Απλή if

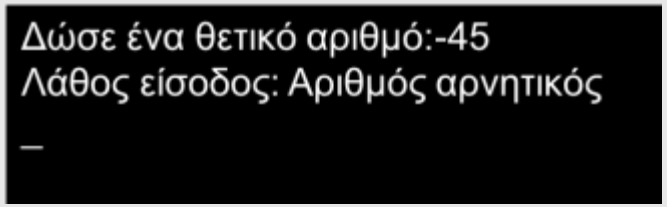
```
if (<συνθήκη1>)
      <πρόταση1>
      else if (<συνθήκη2>)
            <πρόταση2>
            else if
(<συνθήκη3>)
            <πρόταση3>
else <πρόταση4>
```

Εμφωλευμένη if

Παράδειγμα: if

Το πρόγραμμα ζητά ένα μη-αρνητικό αριθμό από το χρήστη και τυπώνει την τετραγωνική του ρίζα.

```
#include <stdio.h>
#include <math.h> /* βιβλ. μαθημ. συναρτήσεων */


main()
{
   double num;
   printf("Δώσε ένα θετικό αριθμό:");
   scanf("%lf",&num);
   if (num<0)
        printf("Λάθος είσοδος: Αριθμός αρνητικός\n");
   else
        printf("Η τετραγωνική ρίζα του %lf είναι
   %lf\n",num,sqrt(num));
}
```



Η Πρόταση if (2/2)

- Η πρόταση if δέχεται ως συνθήκη οποιαδήποτε έκφραση επιστρέφει έναν ακέραιο:
  - Αν ο ακέραιος είναι διάφορος του 0 (η συνθήκη είναι αληθής) τότε εκτελούνται οι εντολές που ακολουθούν.
  - Αν ο ακέραιος είναι ίσος με 0 (η συνθήκη είναι ψευδής) τότε δεν εκτελούνται.

```
1.   #include <stdio.h>

2.   int main ()
3.   {
4.    int bathmos;

1.    printf("Dwse ton bathmo sou sth C \n");
2.    scanf("%d",&bathmos);
3.    if (bathmos >= 5) {
4.      printf("Bravo! Exeis perasei to mathima\n");
        printf("O bathmos sou einai %d\n",bathmos);
5.    }
6.    else
        printf("Lupamai! Dokimase pali ton Septembrh\n");
7.    return 1;
8.   }
```

Η Πρόταση if (παράδειγμα)

```
#include <stdio.h>

int main()
{
    int num1, num2;

    printf( "Enter two integers\n" );
    scanf( "%d%d", &num1, &num2 );

    if ( num1 == num2 )
        printf( "%d is equal to %d\n", num1, num2 );

    if ( num1 != num2 )
        printf( "%d is not equal to %d\n", num1, num2 );

    if ( num1  < num2 )
        printf( "%d is less than %d\n", num1, num2 );

    if ( num1  > num2 )
        printf( "%d is greater than %d\n", num1, num2 );

    if ( num1  <= num2 )
        printf( "%d is less than or equal to %d\n", num1, num2 );

    if ( num1  >= num2 )
        printf( "%d is greater than or equal to %d\n", num1, num2 );
}
```

Εμφωλιασμένα (ένθετα) if

```
1.   #include <stdio.h>

2.   int main ()
3.   {
4.    int bathmos_eksetasis, bathmos_lab;

5.    printf("Dwse ton bathmo sou sth eksetasi ths C \n");
6.    scanf("%d",&bathmos_eksetasis);
7.    if (bathmos_eksetasis < 5) {
8.      printf("Lupamai! Dokimase pali ton Septembrh\n");
9.    }
10.   else {
11.     printf("Dwse ton bathmo sou sto ergastirio ths C \n");
12.     scanf("%d",&bathmos_lab);
        if (bathmos_lab >= 5) {
13.       printf("Bravo! Exeis perasei to mathima\n");
          printf("Oi bathmoi sou einai eksetasi:%d \t ergastirio:%d\n",
                  bathmos_eksetasis, bathmos_lab);
14.     }
15.     else
          printf("Lupamai! Dokimase pali ton Septembrh\n");
      }
16.   return 1;
17.  }
```

Η Πρόταση if (προσοχή)

- Χρειάζεται προσοχή όταν χρησιμοποιούμε τον τελεστή ισότητας στην συνθήκη μιας if.
  - Να μην γράψουμε αντί για αυτό τον τελεστή εκχώρησης!

```
if (bathmos == 10) {
  printf("Sugxaritiria! Pires arista!\n");
}
```

αλλά όχι

```
if (bathmos = 10) {
  printf("Sugxaritiria! Pires arista!\n");
}
```

  - Το δεύτερο if εκτελεί πάντα την printf ανεξάρτητα από την τιμή του x! Γιατί;

Επειδή η έκφραση x=10 επιστρέφει την τιμή 10 που είναι διάφορη του 0.

Άρα η συνθήκη είναι πάντα αληθής.

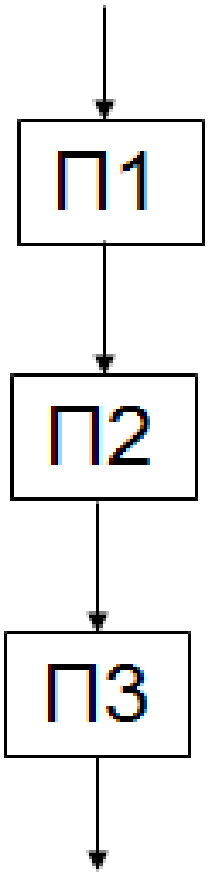
Το αποτέλεσμα θα είναι να νομίζει ο χρήστης ότι πάντα παίρνει άριστα!

Τελεστής επιλογής υπό συνθήκη

- Η C παρέχει έναν τελεστή που μπορεί να αντικαταστήσει, και να συντομεύσει, το **if** σε αρκετές περιπτώσεις.
- Υπό συνθήκη `(? :)`
  - `<έκφρ1> ? <εκφρ2> : <εκφρ3>;`
  - `(a > b) ? a : b;`
  - Αν a > b τότε a, αλλιώς b.

Δομημένος Προγραμματισμός (2/2)

- Ο πιο απλός και συνηθισμένος τρόπος εκτέλεσης μιας ακολουθίας εντολών είναι ο ακολουθιακός.
  - Η κάθε εντολή εκτελείται μετά από την άλλη όπως είναι γραμμένες στον κώδικα.
- Οι γλώσσες δομημένου προγραμματισμού επιτρέπουν πιο ευέλικτες δομές ελέγχου ροής του προγράμματος.
  - **Δομή υπό-συνθήκη διακλάδωσης.**
    - Συχνά θέλουμε να επιλέξουμε ποιο κομμάτι κώδικα θα εκτελεστεί .
  - **Δομή επανάληψης.**
    - Συχνά θέλουμε να επαναλάβουμε τα ίδια βήματα πολλές φορές.



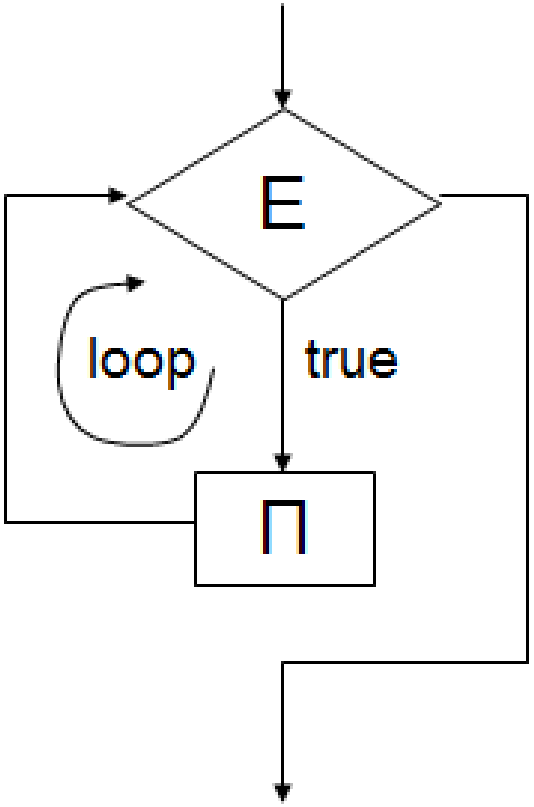
Δομές (Βρόχοι) Επανάληψης

- Υπό συνθήκη: άγνωστος αριθμός επαναλήψεων.
  - Συνθήκη εισόδου.
  - Συνθήκη εξόδου.
- Με μετρητή: γνωστός αριθμός επαναλήψεων.

Επανάληψη με Συνθήκη Εισόδου

while Ε Π

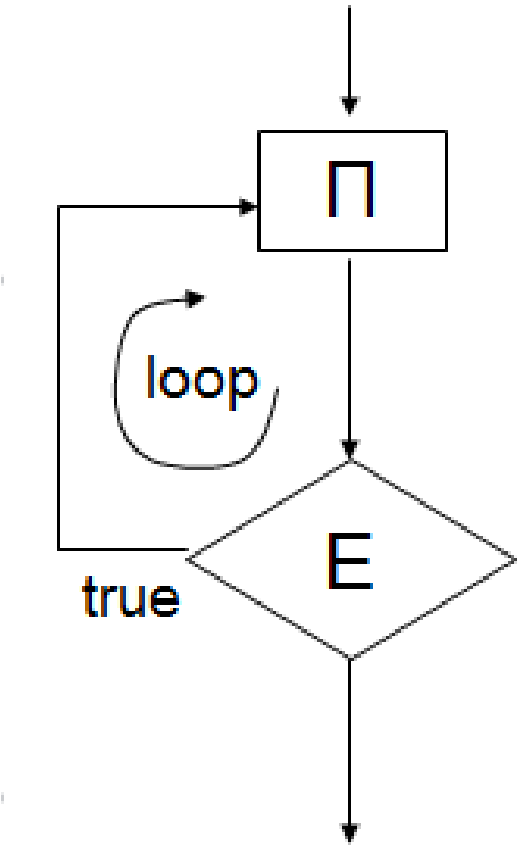
Το Π μπορεί να μην εκτελεστεί ποτέ.



Επανάληψη με Συνθήκη Εξόδου

do Π **while** Ε

Το Π θα εκτελεστεί τουλάχιστον μία φορά.

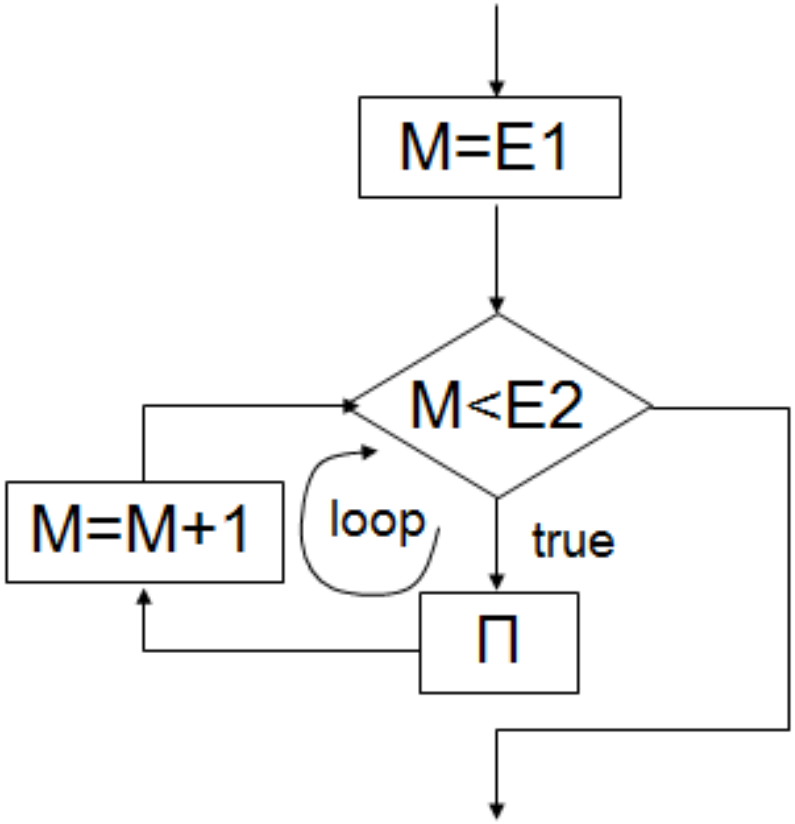


Επανάληψη με Μετρητή

```
M=E1
for M<E2
   Π, M=M+1
```

Το Π θα εκτελεστεί E2-E1 φορές.

M: Μετρητής.

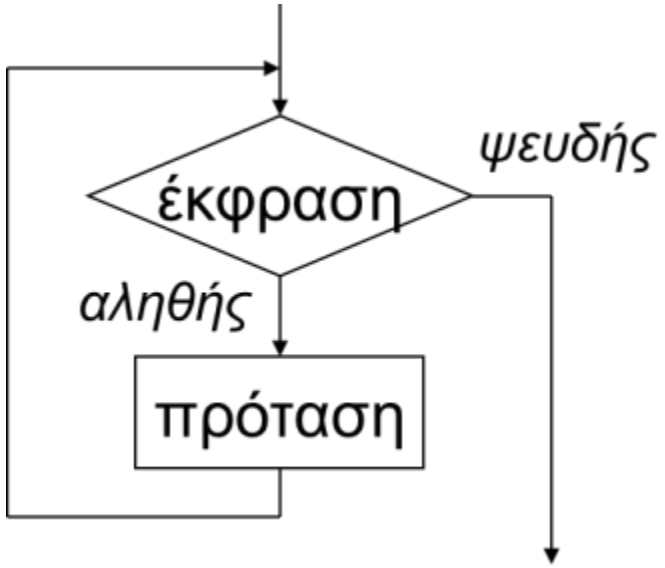


Προτάσεις Ελέγχου Ροής στη C (2/2)

- Επανάληψη:
  - while.
  - do while.
  - for.
- Διακλάδωση χωρίς συνθήκη:
  - break.
  - continue.
  - goto.

Πρόταση Επανάληψης while

```
while (<έκφραση>)
      <πρόταση>
```



Η πρόταση επανάληψης **while** επιτρέπει την εκτέλεση κάποιων γραμμών κώδικα επαναληπτικά όσο ισχύει μια συγκεκριμένη συνθήκη.

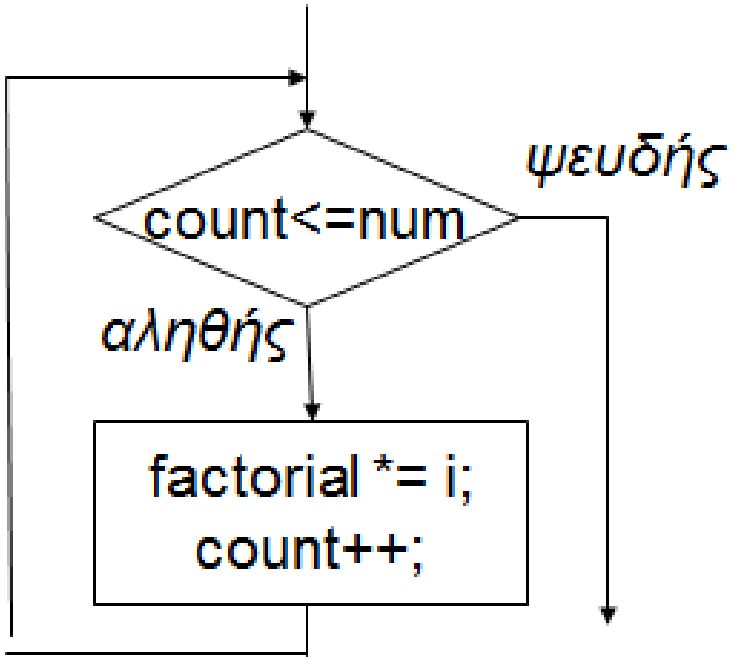
```
1.  #include <stdio.h>

2.  void main ()
3.  {
4.   int i=0;
5.    while (i<100) {
6.       printf("%d\n",i);
7.       i = i+1;
8.    }
9.  }
```

Τι κάνει αυτό το απλό πρόγραμμα

Τι θα γίνει αν παραλείψουμε τη γραμμή 7;

Παράδειγμα: while

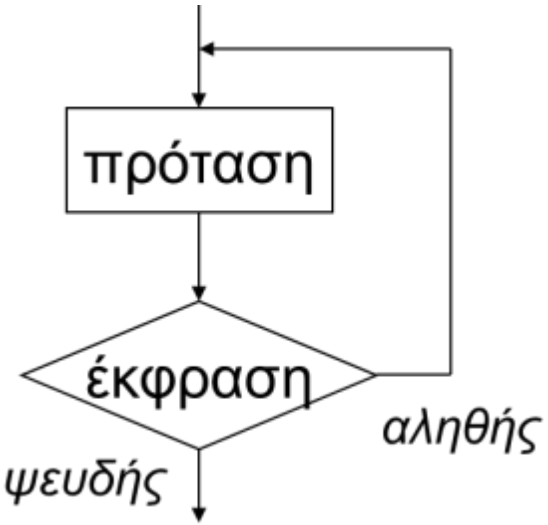
```
int num, count;
double factorial;
printf("Dwse
  arithmo:");
scanf("%d",&num);
factorial = 1.0;
count = 1;
while (count <= num)
   {
   factorial *= count;
   count++;
   }
```



Υπολογισμός παραγοντικού:
1!=1
n!=1*2*3... (n-2)*(n-1)*n

Η Πρόταση Επανάληψης do while

```
do {
        <πρόταση>
}       while (<έκφραση>)
```



Η πρόταση επανάληψης **do while** είναι παρόμοια με την **while** αλλά το κομμάτι κώδικα που περιέχει πάντα εκτελείται τουλάχιστον μια φορά.

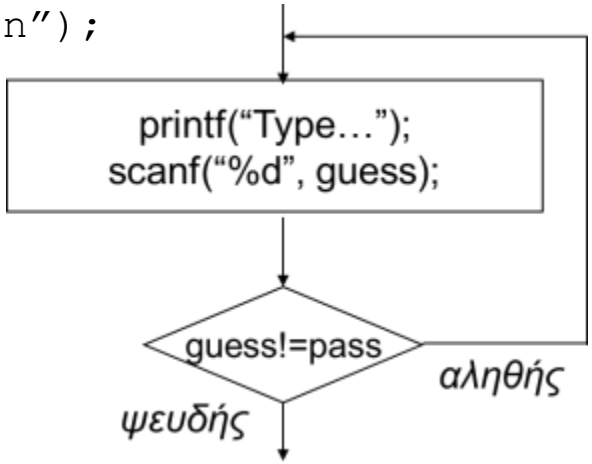
```
1.  #include <stdio.h>

2.  void main ()
3.  {
4.   int i=0;
5.    do {
6.       printf("%d\n",i);
7.       i = i+1;
8.    } while (i<100);
9.  }
```

Τι θα γίνει αν παραλείψουμε τη γραμμή 7;

Προσοχή στο ερωτηματικό. Είναι απαραίτητο.

Παράδειγμα: do while

```
1.  #include <stdio.h>

2.  void main ()
3.  {
4.   int guess;
5.   const int pass = 1231;
6.    do {
7.       printf("Type the password to proceed\n");
8.       scanf("%d",&guess);
9.    } while (guess != pass);
10.   printf("You have guessed correctly!\n");
11. }
```



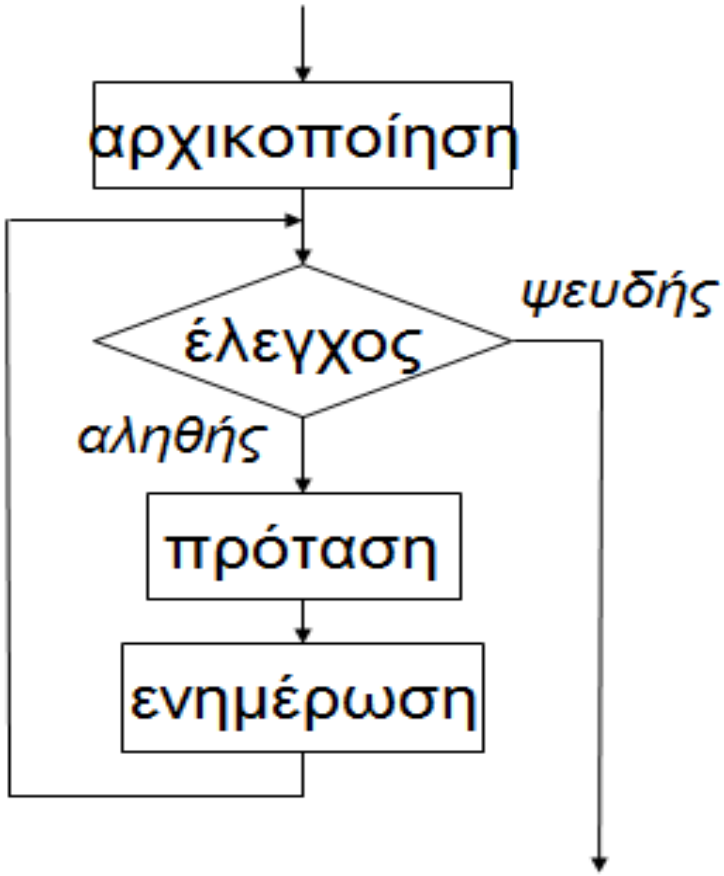
Μπορείτε να κάνετε ακριβώς το ίδιο χρησιμοποιώντας while;

Η Πρόταση for

for (<έκφραση1>;
<έκφραση2>; <έκφραση3>)
<πρόταση>

- έκφραση1: αρχικοποίηση.
- έκφραση2: έλεγχος.
- έκφραση3: ενημέρωση.

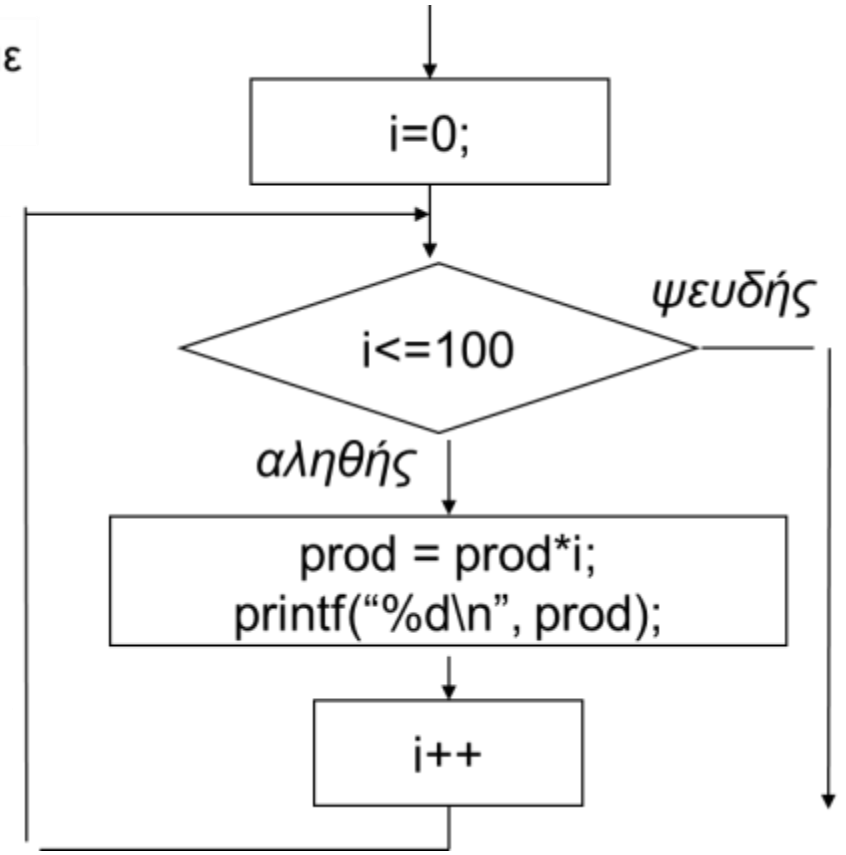
```
1.  #include <stdio.h>

2.  void main ()
3.  {
4.   int i;
5.    for (i=0; i<100; i++) {
6.       printf("%d\n",i);
7.    }
8.  }
```



Παράδειγμα: for

Τι θα γίνει αν γράψουμε κατά λάθος i--;

```
1.  #include <stdio.h>

2.  void main ()
3.  {
4.   int i, prod=1;
5.    for (i=1; i<=100; i++) {
6.       prod = prod*i;
7.       printf("%d\n",prod);
8.    }
9.  }
```



Τι κάνει αυτό το πρόγραμμα;

Μπορείτε να κατασκευάσετε ισοδύναμα προγράμματα με χρήση **while** και **do while**;

for loops (περισσότερα) (1/2)

Η συνθήκη μπορεί να είναι ανεξάρτητη από τις μεταβλητές του for.

```
1. #include <stdio.h>

2. void main ()
3. {
4.  int i, guess;
5.  const int pass = 1231;
6.   printf("Type the password to proceed\n");
7.   scanf("%d",&guess);
8.   if (guess == pass)
9.     printf("You have guessed correctly!\n");
10.  else {
11.    for (i=2; guess!=pass; i++) {
12.      printf("Type the password to proceed\n");
13.      scanf("%d",&guess);
14.      printf("attempt number %d\n",i);
15.    }
16.    printf("You have guessed correctly!\n");
17.  }
18.}
```

Αυτές οι γραμμές χρειάζονται;

for loops (περισσότερα) (2/2)

- Η συνθήκη του for μπορεί να περιέχει περισσότερες από μια μεταβλητές.

```
1.  #include <stdio.h>

2.  void main ()
3.  {
4.   int i,j;
5.    for (i=0,j=100; i!=j; i++,j--)
    {
6.       printf("%d\n",j-i);
7.    }
8.  }
```

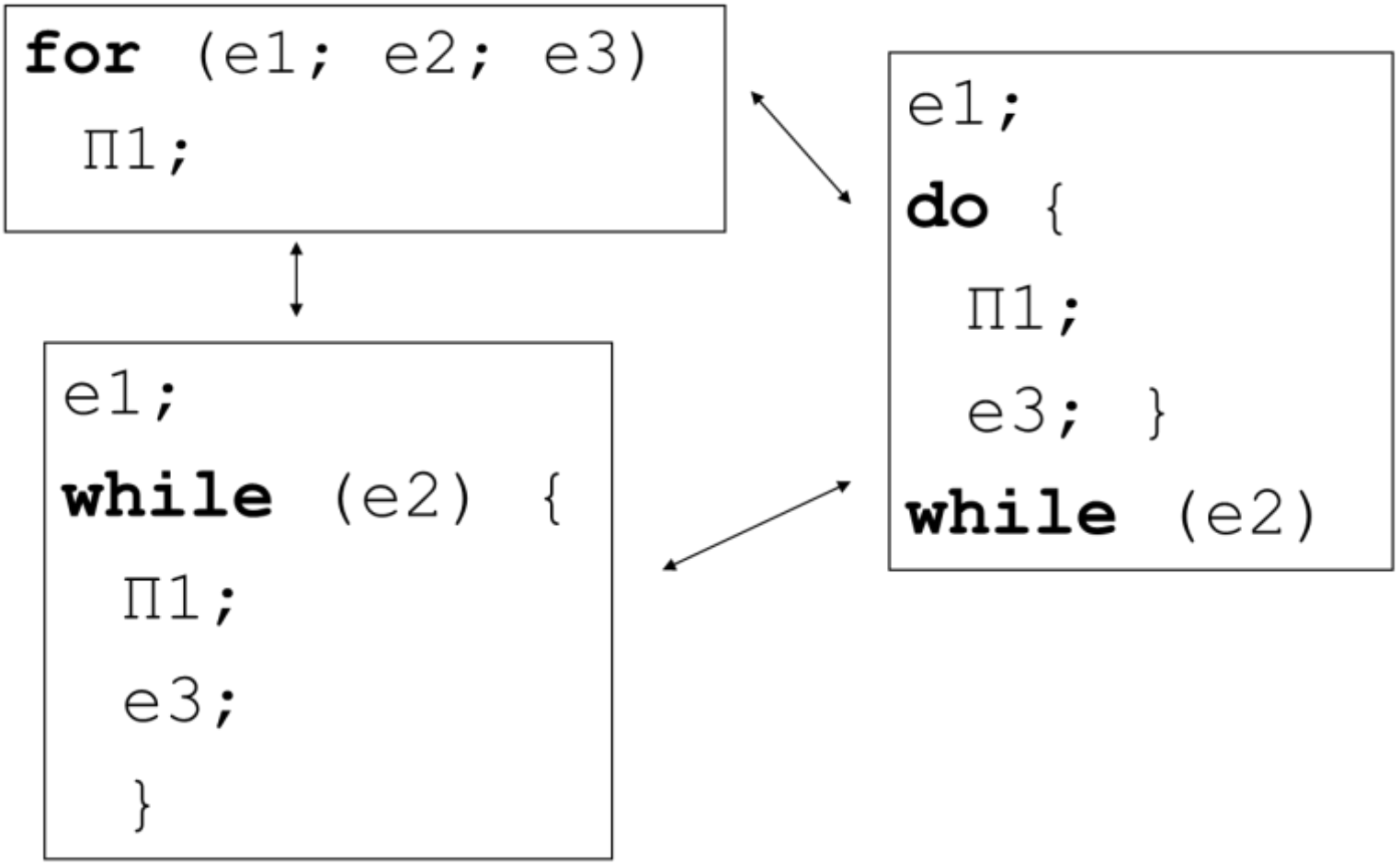
- Η αύξηση του μετρητή δεν είναι απαραίτητο να είναι κατά 1.

```
1.  #include <stdio.h>

2.  void main ()
3.  {
4.   int i;
5.    for (i=0; i<100; i+=5) {
6.       printf("%d\n",i);
7.    }
8.  }
```

Επιλογή Βρόχου (loop)

- Προτιμούμε τη δομή συνθήκης εισόδου (while) από τη δομή συνθήκης εξόδου (do while).
- Αν υπάρχει ή μπορεί να οριστεί απαριθμητής επαναλήψεων, προτιμούμε τη for από την while.
- Γενικά, οτιδήποτε εκφράζεται με τη while μπορεί να εκφραστεί και με τη for και ανάποδα.

Ισοδυναμία Δομών Επανάληψης



Ένθετοι Βρόχοι

```
#include <stdio.h>
int main() {
 int i,j,result;
 for (i=1; i<=10; i++)
  {
  for(j=1; j<=10; j++)
      {
      result=i*j;
      printf(" %d\t",result);
      }
  printf("\n");
  }
 return 1;
}
```

Διακλάδωση Χωρίς Συνθήκη

- Διαχείριση ειδικών περιπτώσεων σε προτάσεις επανάληψης:
  - break.
  - continue.

- Ρητή διακλάδωση:
  - goto.

Η Πρόταση break

- Μερικές φορές χρειάζεται να σταματήσουμε πρόωρα την εκτέλεση ενός βρόγχου.
- Η εντολή **break** προκαλεί την έξοδο μόνο από τον πιο εσωτερικό βρόχο.
- **Καταστρέφει** τη δομή του προγράμματος.
- **Πάντα** υπάρχει τρόπος για να αποφύγουμε τη χρήση της:
  - (εκτός από τη χρήση της στην εντολή switch).
  - Παρόλα αυτά χρησιμοποιείται αρκετά συχνά.

Χρήση της break

```
while (έκφραση) {
   if (ειδική περίπτωση)
   {
   προτάσεις επεξεργασίας ειδικών περιπτώσεων
   break;
   }
   προτάσεις επεξεργασίας κανονικής περίπτωσης
   }
```

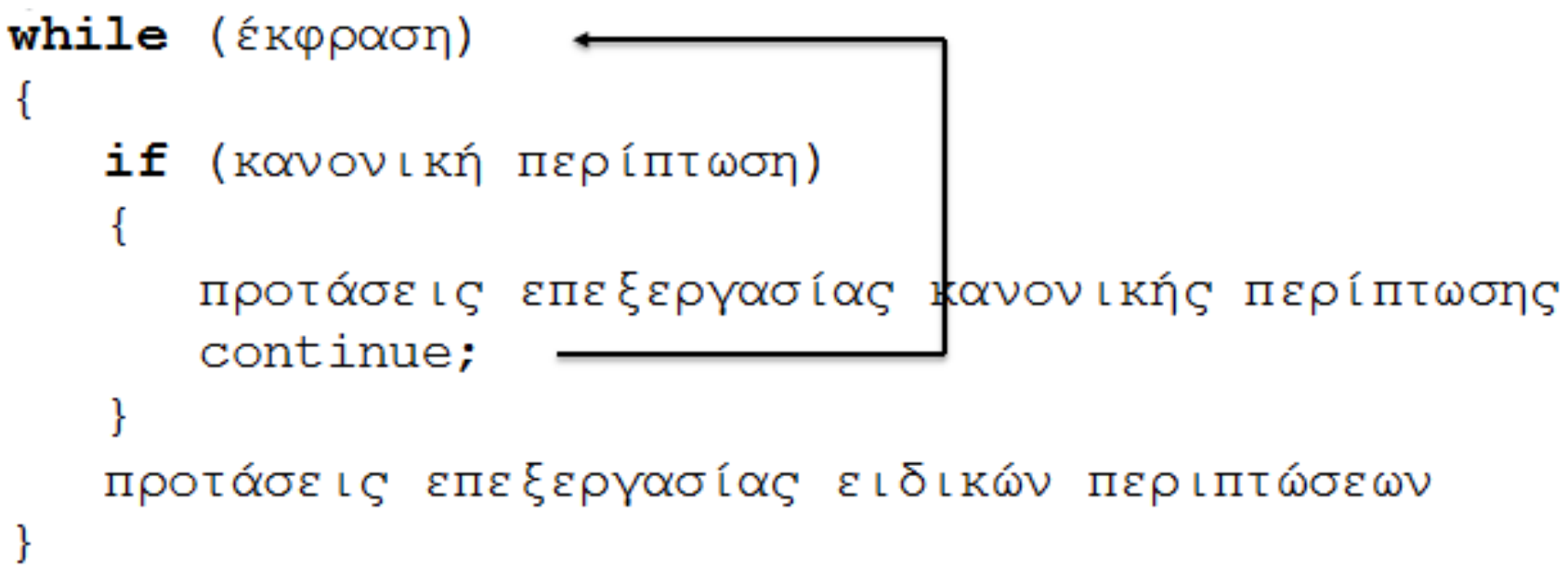
```
for (i=0; i < max; i++) {
   scanf("%d",&num);
   if (num < 0)
        break;
   printf("process is continued\n");
}
printf("end of process\n");
```

Παράδειγμα: break

```
1.  #include <stdio.h>

2.  void main ()
3.  {
4.   int i, guess;
5.   const int pass = 1231;
6.    printf("You have 10 tries to guess the password!\n");
7.    for (i=1; i<=10; i++) {
8.        printf("try number %d\n",i);
9.        printf("Type the password to proceed\n");
10.       scanf("%d",&guess);
11.       if (guess == pass) break;
12.   }
13.   if (i<=10)
14.       printf("You have guessed correctly!\n");
15.   else
16.       printf("You have failed\n");
17. }
```

Η Πρόταση continue

- Προκαλεί την έναρξη της επόμενης επανάληψης for, while, do.
- Επηρεάζει μόνο τον πιο εσωτερικό βρόχο.



Παράδειγμα: continue

```
for (i=0; i < max; i++) {
     scanf("%d",&num);
   if (num > 0)
        continue;
   printf("process is continued\n");
}
printf("end of process\n");
```

Τι κάνει αυτό το πρόγραμμα;

```
1.  #include <stdio.h>

2.  void main ()
3.  {
4.   int i;
5.   for (i=1; i<=100; i++) {
6.        if ((i%3)!=0) continue;
7.   printf("%d\n",i);
8.   }
9.  }
```

Η Πρόταση switch

Σε κάποιες περιπτώσεις (όταν υπάρχουν πολλές εναλλακτικές επιλογές) το if…else δεν είναι η καλύτερη επιλογή.

```
switch (<έκφραση>) {
   case <σταθ-εκφρ1> :
        πρόταση1
        break;
   case <σταθ-εκφρ2> :
        πρόταση2
        break;
   ...
   case <σταθ-εκφρN> :
        πρότασηN
        break;
   default : πρόταση
}
```

- Κάθε case πρέπει να έχει μία int ή char σταθερά ή μία σταθερά έκφραση.
- Δύο case δεν μπορούν να έχουν την ίδια τιμή.
- Η πρόταση της default εκτελείται όταν καμία από τις case δεν ικανοποιείται.
- Η default δεν είναι απαραίτητο να είναι τελευταία.

Παράδειγμα: switch

```
...
printf(“Enter your choice\n”);
scanf(“%d”,&choice);
switch(choice) {
   case 1: result = num1+num2; break;
   case 2: result = num1-num2; break;
   case 3: result = num1*num2; break;
   case 4: if (num2) result = num1/num2;
             else printf("Δεν επιτρέπεται
   διαίρεση με το
   μηδέν\n"); break;
   default: printf("Λάθος επιλογή \n");
}
printf("Το αποτέλεσμα είναι %f",result);
...
```

Παράδειγμα: Αριθμομηχανή

- Να γραφεί πρόγραμμα C που να διαβάζει 2 αριθμούς και το σύμβολο μιας αριθμητικής πράξης και να υπολογίζει (εμφανίζει) το αποτέλεσμα.

  - `Είσοδος: 4 + 5    Έξοδος: 9`
  - `Είσοδος: 4 * 5    Έξοδος: 20`
  - `Είσοδος: 4 / 5    Έξοδος: 0.8`

Αριθμομηχανή

```
#include <stdio.h>

    main()
    {
       float num1, num2;

       char op;

       printf("ΔΩΣΤΕ ΑΡΙΘΜΟ ΠΡΑΞΗ ΑΡΙΘΜΟ : ");

        scanf("%f %c %f", &num1, &op, &num2 );

       if ( op == '+' )
          printf ( " = %f", num1 + num2 );

       else if ( op == '-' )

       printf ( " = %f", num1 - num2 );

       else if ( op == '*' )

       printf ( " = %f", num1 * num2 );

       else if ( op == '/' )

       printf ( " = %f", num1 / num2 );

       printf ( "\n\n" );
    }
```

```
ΔΩΣΤΕ ΑΡΙΘΜΟ ΠΡΑΞΗ ΑΡΙΘΜΟ
: 5 + 3

= 8

_
```

Αριθμομηχανή με Switch

```
void main() {
float a,b;
char op;
printf("Dose 2 times:\n");
scanf("%f %f",&a,&b);
printf("Dose telesti praxis:\n");
scanf("%c",&op);
switch(op)  {
   case '+': printf(" = %f\n",a+b); break;
   case '-': printf(" = %f\n",a-b); break;
   case '*': printf(" = %f\n",a*b); break;
   case '/': if (b!=0)
              printf(" = %f\n",a/b);
             else
              printf("Den orizetai piliko\n"); break;
   default: printf("Lathos telestis\n");   }
}
```

Μενού επιλογών

```
#include <stdio.h>
main()
{
   int cmd = 0;
   printf(" 1. ΠΡΩΤΗ\n 2. ΔΕΥΤΕΡΗ\n 3. ΤΡΙΤΗ 4. ΤΕΤΑΡΤΗ 9.
ΕΞΟΔΟΣ\n");
   printf("\nΕΠΙΛΟΓΗ : ");
   scanf("%d", &cmd);
   switch ( cmd )
   {
      case 1 : printf("ΕΔΩ ΕΚΤΕΛΕΙΤΑΙ Η 1η ΕΠΙΛΟΓΗ\n");
break;
      case 2 : printf("ΕΔΩ ΕΚΤΕΛΕΙΤΑΙ Η 2η ΕΠΙΛΟΓΗ\n");
break;
      case 3 : printf("ΕΔΩ ΕΚΤΕΛΕΙΤΑΙ Η 3η ΕΠΙΛΟΓΗ\n");
break;
      case 4 : printf("ΕΔΩ ΕΚΤΕΛΕΙΤΑΙ Η 4η ΕΠΙΛΟΓΗ\n");
break;
      case 9 : printf("ΕΔΩ ΕΚΤΕΛΕΙΤΑΙ Η ΕΞΟΔΟΣ\n"); break;
      default: printf("ΛΑΘΟΣ ΕΠΙΛΟΓΗ\n");
   }
}
```

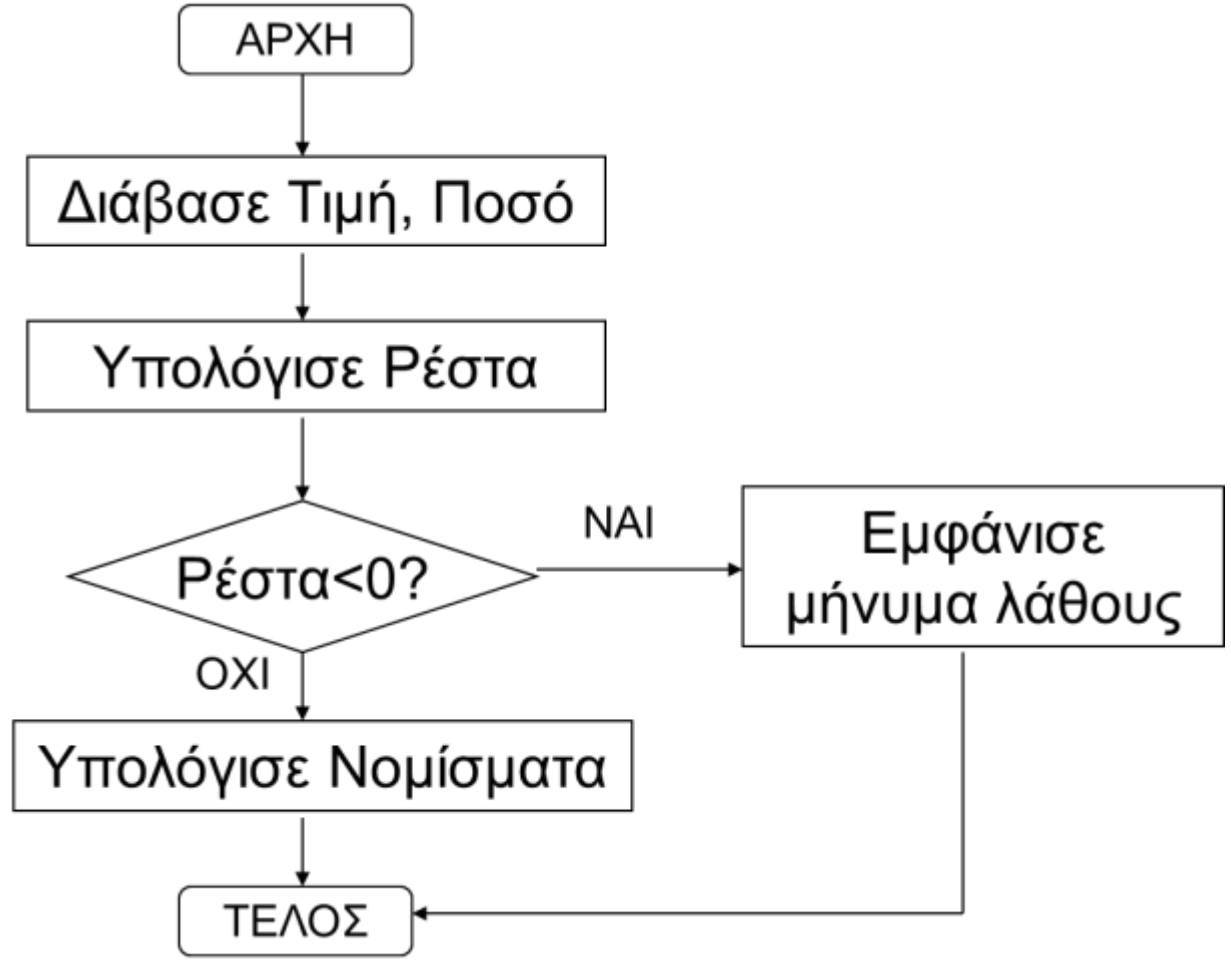
```
 1. ΠΡΩΤΗ
 2. ΔΕΥΤΕΡΗ
 3. ΤΡΙΤΗ
 4. ΤΕΤΑΡΤΗ
 9. ΕΞΟΔΟΣ


ΕΠΙΛΟΓΗ : 6
ΛΑΘΟΣ ΕΠΙΛΟΓΗ

_
```

Άσκηση

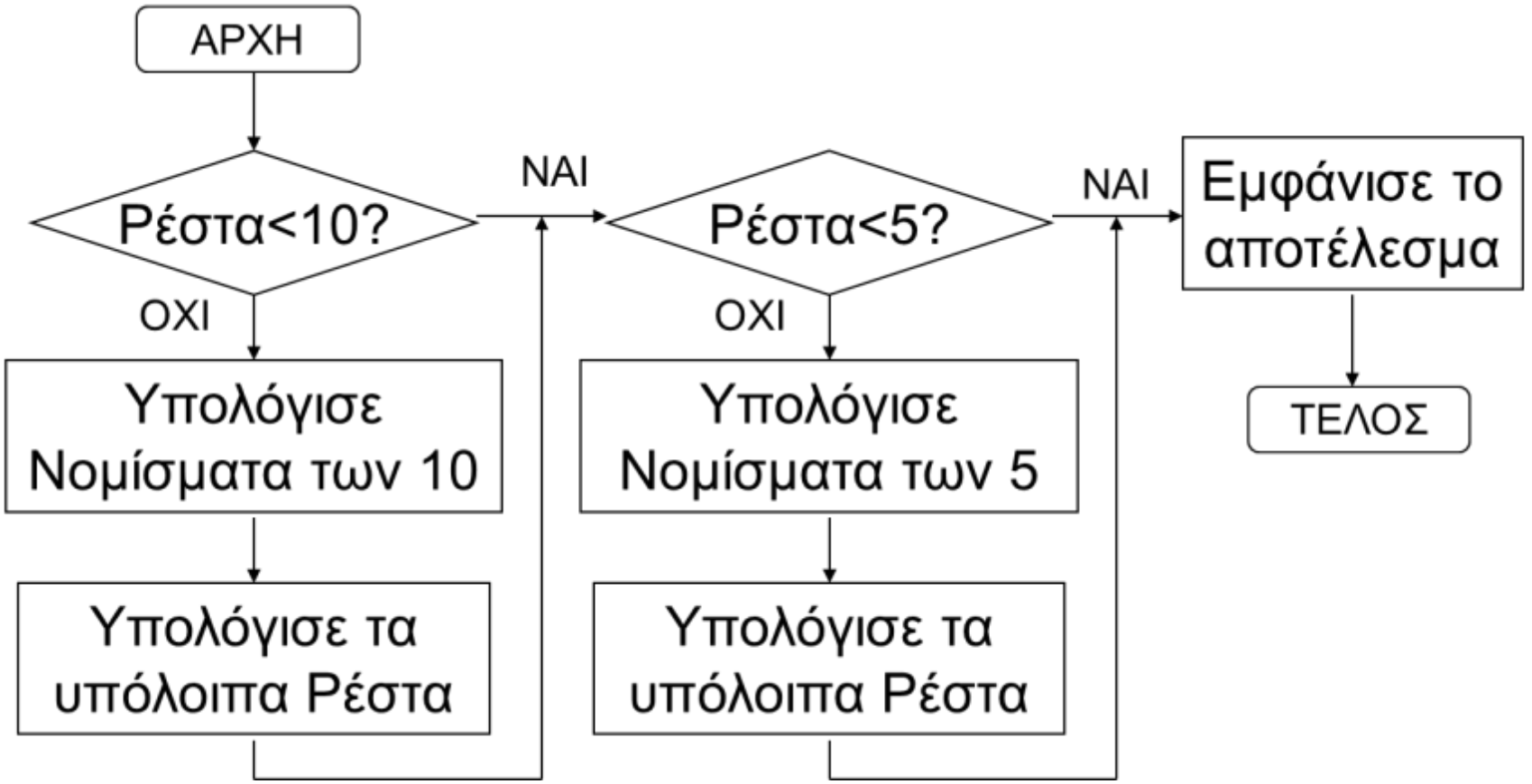
- Να γραφεί πρόγραμμα C που να διαβάζει από το πληκτρολόγιο 2 αριθμούς που αντιστοιχούν στην τιμή ενός προϊόντος και στο ποσό που δίνει κάποιος για να το αγοράσει και υπολογίζει (εμφανίζει) τα ρέστα σε νομίσματα των 10, 5 και 1 ευρώ.

  - Είσοδος: τιμή=65, ποσό=100.
  - Έξοδος: 3 των 10 ευρώ και 1 των 5 ευρώ.

Διάγραμμα Ροής



Διάγραμμα Ροής: Υπολόγισε Νομίσματα



Πολύπλοκες συνθήκες με χρήση λογικών τελεστών

- Μια βασική χρήση των λογικών τελεστών &&, ||, ! είναι στην κατασκευή πολύπλοκων συνθηκών σε βρόγχους και προτάσεις επιλογών.

τελεστής	σημασία	παράδειγμα
&&	λογικό AND	**x** > 10 && **x** < 20
||	λογικό OR	**x** <= 10 || **x** >= 20
!	λογικό NOT	!(**x**>10 && **x** < 20)

Πολύπλοκες συνθήκες (παράδειγμα) (1/2)

```
1.  #include <stdio.h>
2.  void main ()
3.  {
4.   int i, guess;
5.   const int pass = 1231;
6.    printf("Type the password to proceed\n");
7.    scanf("%d",&guess);
8.    if (guess == pass)
9.      printf("You have guessed correctly!\n");
10.   else {
11.      printf("You have 9 tries left!\n");
12.      i = 2;
13.      while ((i<=10) && (quess != pass)) {
14.         printf("try number %d\n",i);
15.         printf("Type the password to proceed %d\n");
16.         scanf("%d",&guess);
17.         i++;
18.      }
19.      if (i<=10)
20.         printf("You have guessed correctly!\n");
21.      else
22.         printf("You have failed\n");
23.   }
24. }
```

Πολύπλοκες συνθήκες (παράδειγμα) (2/2)

```
1.  #include <stdio.h>

2.  void main ()
3.  {
4.   char c;
5.   int count=0;
6.    scanf("%c",&c);
7.    while ((c>='a' && c <= 'z') || (c>='A' && c <= 'Z')) {
8.       if (c=='s' || c=='S' || c=='t' || c=='T')
9.          count++;
10.      scanf("%c",&c);
11.   }
12.   if (!(count==0))
13.      printf("%d\n",count);
14. }
```

- Τι κάνει αυτό το πρόγραμμα;
- Πως αλλιώς μπορούμε να γράψουμε το !(count==0);

Προτεραιότητα τελεστών

Ο τρόπος υπολογισμού μιας έκφρασης εξαρτάται από την προτεραιότητα των τελεστών:

1. **παρενθέσεις:** $()$
   Υπολογίζονται πρώτα, από τα αριστερά προς τα δεξιά. Εάν υπάρχουν ένθετες υπολογίζονται πρώτα οι εσωτερικές.
2. **μοναδιαίοι τελεστές:** $+$, $-$, $++$, $--$ και $!$
   Υπολογίζονται από δεξιά προς τα αριστερά.
3. **πολλαπλασιασμός, διαίρεση και υπόλοιπο:** $*$, $/$, ή $\%$
   Υπολογίζονται δεύτερα από αριστερά προς τα δεξιά.
4. **πρόσθεση, αφαίρεση:** $+$ ή $-$
   Εάν υπάρχουν πολλοί, υπολογίζονται από τα αριστερά προς τα δεξιά.
5. **Σχεσιακοί:** $<$, $>$, $<=$, $>=$
   Υπολογίζονται από τα αριστερά προς τα δεξιά.
6. **Ισότητας:** $==$, $!=$
   Υπολογίζονται από τα αριστερά προς τα δεξιά.
7. **λογικό AND:** && Από αριστερά προς τα δεξιά.
8. **λογικό OR:** || Από αριστερά προς τα δεξιά.
9. **εκχώρησης:** $=$, $+=$, $-=$, $*=$, $/=$, $\%=$
   Από δεξιά προς τα αριστερά.

Η Πρόταση goto

goto <ετικέτα>;
<ετικέτα> :
<πρόταση>

- Ο έλεγχος μεταφέρεται στην εντολή που σημειώνεται με την ετικέτα.
- Η χρήση της **καταστρέφει** τη δόμηση του κώδικα.
- Είναι **πάντα** δυνατό να αποφύγουμε τη χρήση της.

```
1. #include <stdio.h>

2. void main ()
3. {
4.  int num;
5.   scanf("%d",&num);
6.   if (num>0) goto positive;
7.   else if (num<0) goto
   negative;
8.   printf("number is 0\n");
9.
10.  positive:
11.    printf("number is
   positive");
12.  negative:
13.    printf("number is
   negative");
14. }
```

Η goto ΑΠΑΓΟΡΕΥΕΤΑΙ!

Τέλος Ενότητας

Σημείωμα Αναφοράς

- Copyright Πανεπιστήμιο Δυτικής Μακεδονίας, Τμήμα Μηχανικών Πληροφορικής και Τηλεπικοινωνιών, Στεργίου Κωνσταντίνος. «Εισαγωγή στον Δομημένο Προγραμματισμό». Έκδοση: 1.0. Κοζάνη 2015. Διαθέσιμο από τη δικτυακή διεύθυνση:

  https: //eclass.uowm.gr/courses/ICTE258/

Σημείωμα Αδειοδότησης

Διατήρηση Σημειωμάτων

Οποιαδήποτε αναπαραγωγή ή διασκευή του υλικού θα πρέπει να συμπεριλαμβάνει:

- το Σημείωμα Αναφοράς
- το Σημείωμα Αδειοδότησης
- τη δήλωση Διατήρησης Σημειωμάτων
- το Σημείωμα Χρήσης Έργων Τρίτων (εφόσον υπάρχει)

μαζί με τους συνοδευόμενους υπερσυνδέσμους.