

# Προγραμματισμός Υπολογιστών & Υπολογιστική Φυσική

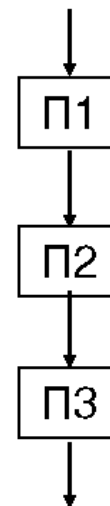
Θεόδωρος Σαμαράς

[theosama@auth.gr](mailto:theosama@auth.gr)

<http://users.auth.gr/voyatzis/CProg>  
<http://users.auth.gr/theosama>

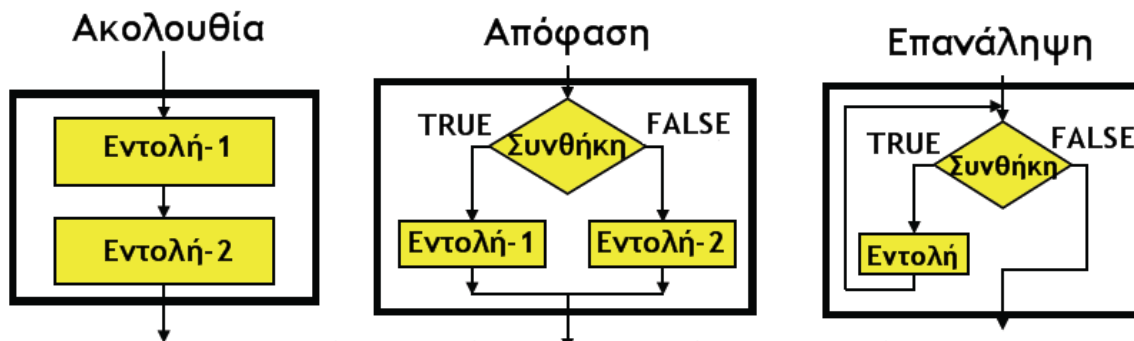
## Ακολουθία προτάσεων

- Ο πιο απλός και συνηθισμένος τρόπος εκτέλεσης μιας ακολουθίας εντολών είναι ο ακολουθιακός: Η κάθε πρόταση εκτελείται μετά από την άλλη
- Οι γλώσσες δομημένου προγραμματισμού επιτρέπουν πιο ευέλικτες δομές ελέγχου ροής του προγράμματος
  - Δομή υπό-συνθήκη διακλάδωσης
  - Δομή επανάληψης



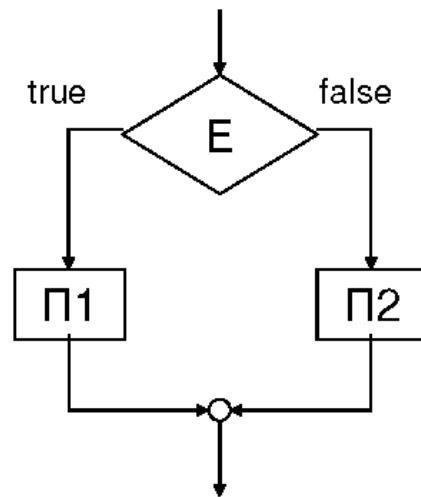
## Δομές Ελέγχου

- Οι Boehm και Jacorini απέδειξαν ότι οποιοσδήποτε αλγόριθμος **μπορεί να εκφραστεί** με συνδυασμούς μόνον τριών διαφορετικών δομών ελέγχου :

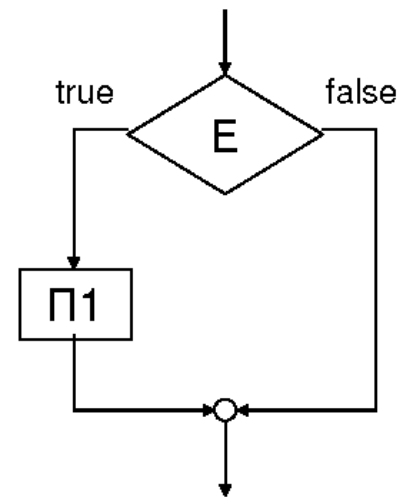


- Ο Dijkstra ισχυρίστηκε ότι οποιοσδήποτε αλγόριθμος **θα πρέπει να εκφράζεται** με χρήση μόνον των τριών δομών ελέγχου (Η GOTO εντολή θα πρέπει να μην χρησιμοποιείται)
- Μια ψευδογλώσσα και οι γλώσσες προγραμματισμού παρέχουν μια ποικιλία εντολών για να εκφράσουν τις παραπάνω δομές.

## Υπό-συνθήκη διακλάδωση

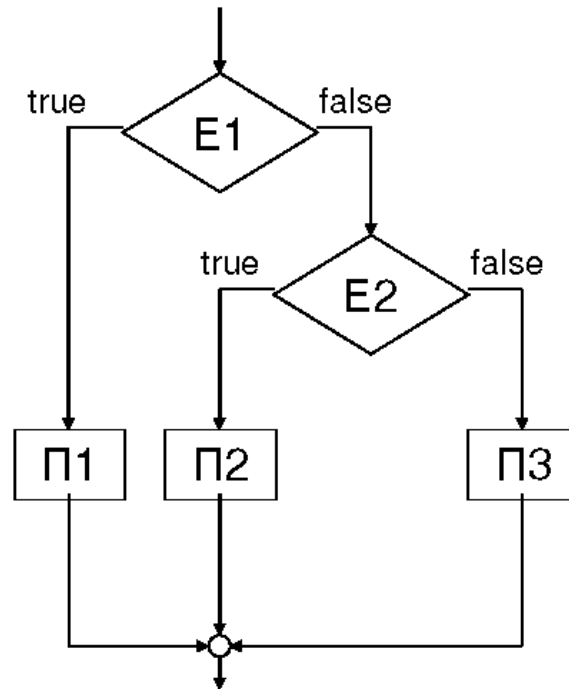


2 προτάσεις:  
if E then Π1 else Π2



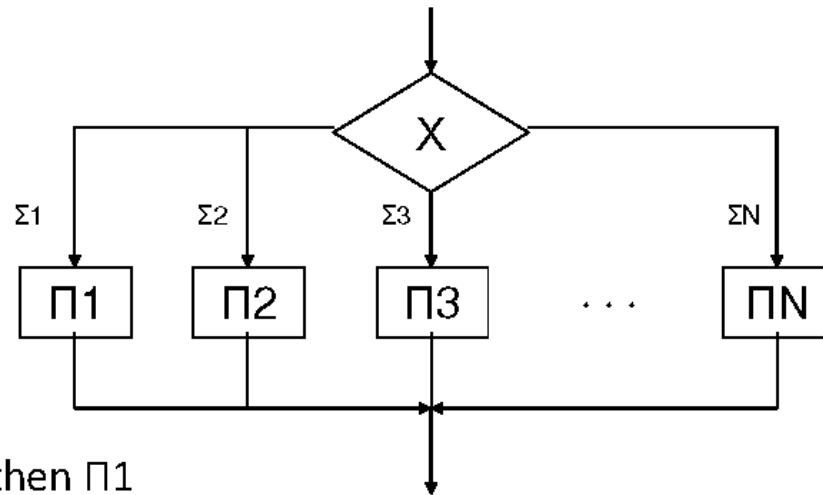
1 πρόταση:  
if E then Π1

## Σύνθετη δομή επιλογής



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

## Δομή επιλογής (πολλαπλή)



If  $X=\Sigma 1$  then  $\Pi 1$   
else if  $X=\Sigma 2$  then  $\Pi 2$   
else if  $X=\Sigma 3$  then  $\Pi 3$   
...  
else if  $X=\Sigma N$  then  $\Pi N$

## Προτάσεις ελέγχου ροής στη C

- Διακλάδωση υπό συνθήκη
  - if-else
  - switch case

## Η πρόταση **if**

```
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));
}
```

Δώσε ένα θετικό αριθμό:-45  
Λάθος είσοδος: Αριθμός αρνητικός

## Η πρόταση **switch**

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

## Κανόνες της switch

- 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

```
...
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 διαβάσει 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
```

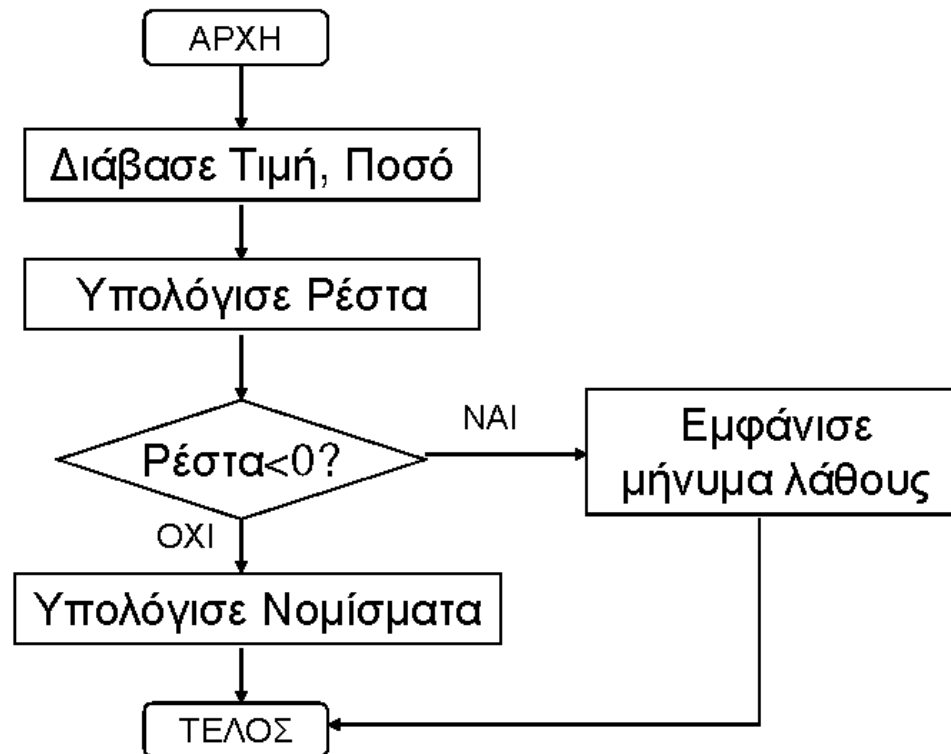
```
ΕΠΙΛΟΓΗ : 6
ΛΑΘΟΣ ΕΠΙΛΟΓΗ
```



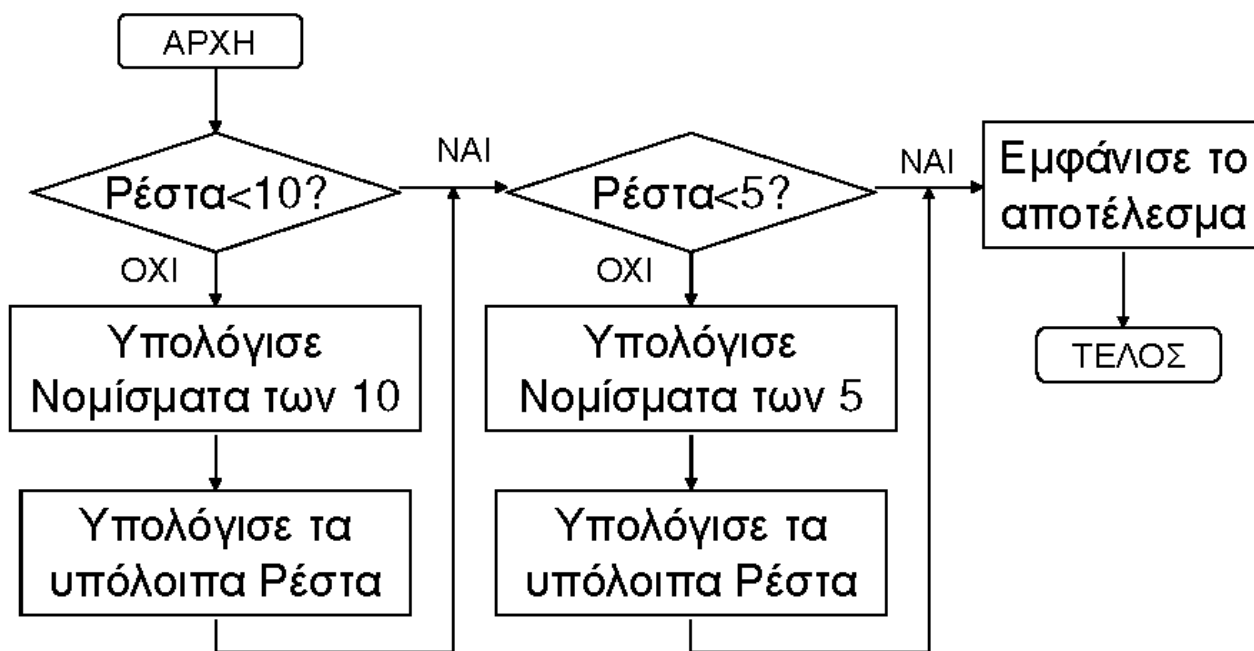
## Άσκηση

- 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



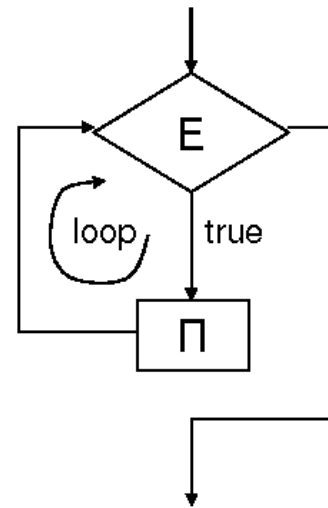
## Δομές (βρόχοι) επανάληψης

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

## Επανάληψη με συνθήκη εισόδου

**while** Ε Π

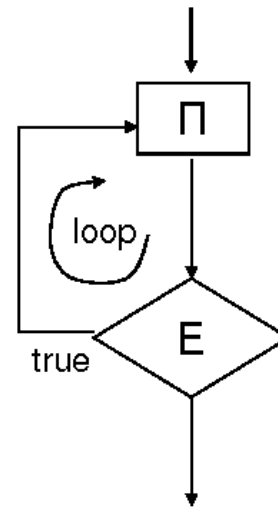
Το Π μπορεί να μην  
εκτελεστεί ποτέ



## Επανάληψη με συνθήκη εξόδου

**do**  $\Pi$  **while**  $E$

Το  $\Pi$  θα εκτελεστεί  
τουλάχιστον μία φορά



## Επανάληψη με μετρητή

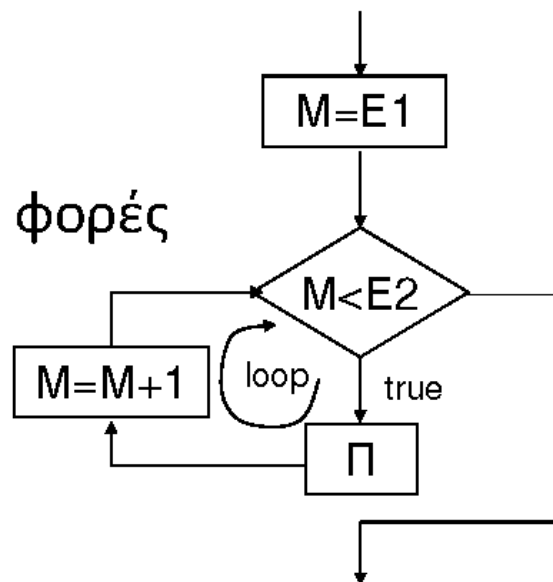
$M = E1$

**for**  $M < E2$

$\Pi$ ,  $M++$

Το  $\Pi$  θα εκτελεστεί  $E2 - E1$  φορές

$M$ : Μετρητής



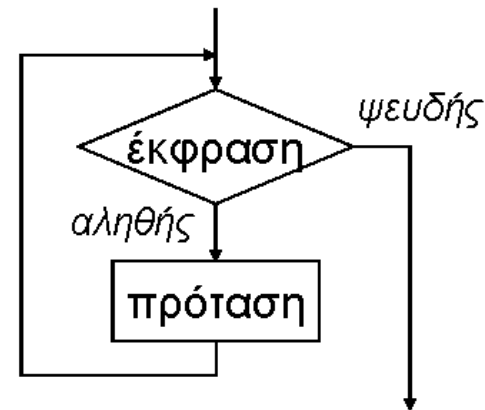
## Προτάσεις ελέγχου ροής στη C

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



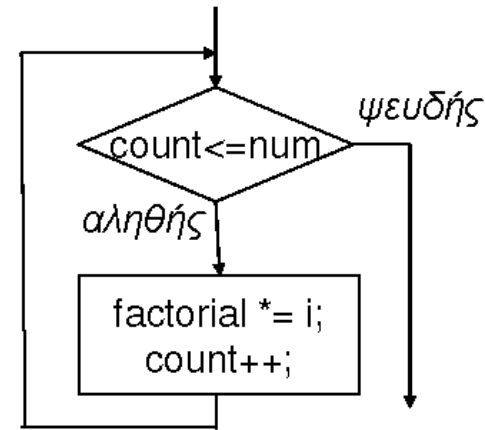
## Πρόταση επανάληψης **while**

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



## Παράδειγμα: while

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



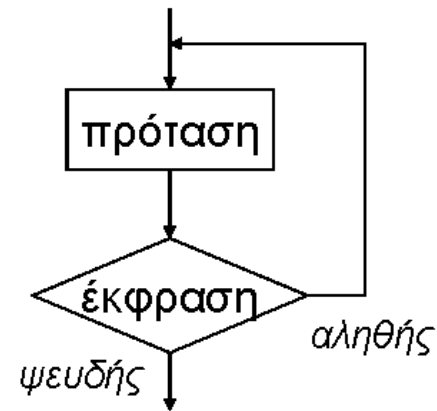
Υπολογισμός παραγοντικού

$0! = 1$

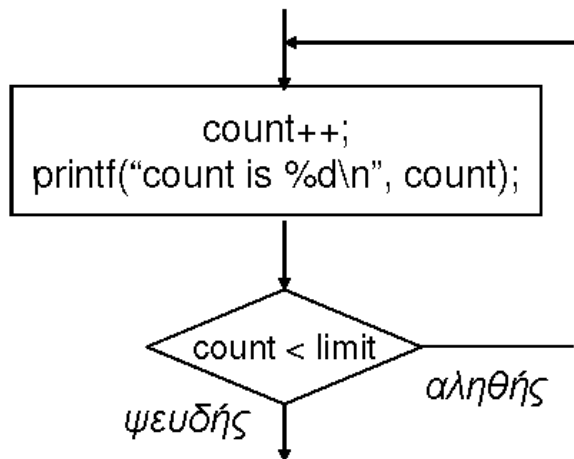
$n! = 1 * 2 * 3 \dots (n-2) * (n-1) * n$

## Η πρόταση επανάληψης **do while**

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



## Παράδειγμα: do while

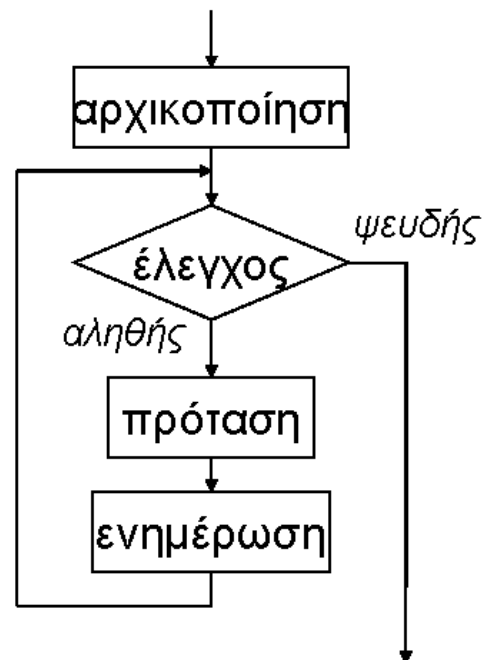


```
do {  
    count++;  
    printf("count is %d\\n", count);  
} while (count < limit)
```

## Η πρόταση **for**

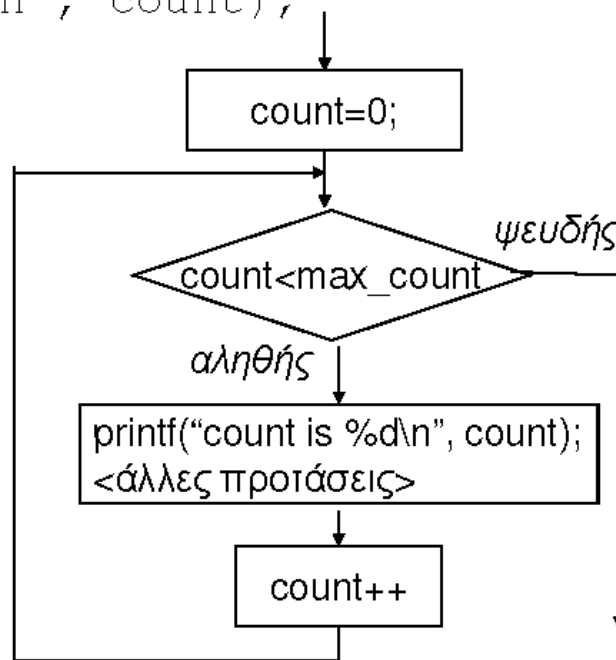
**for** (<έκφραση1>; <έκφραση2>; <έκφραση3>)  
    <πρόταση>

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



## Παράδειγμα: for

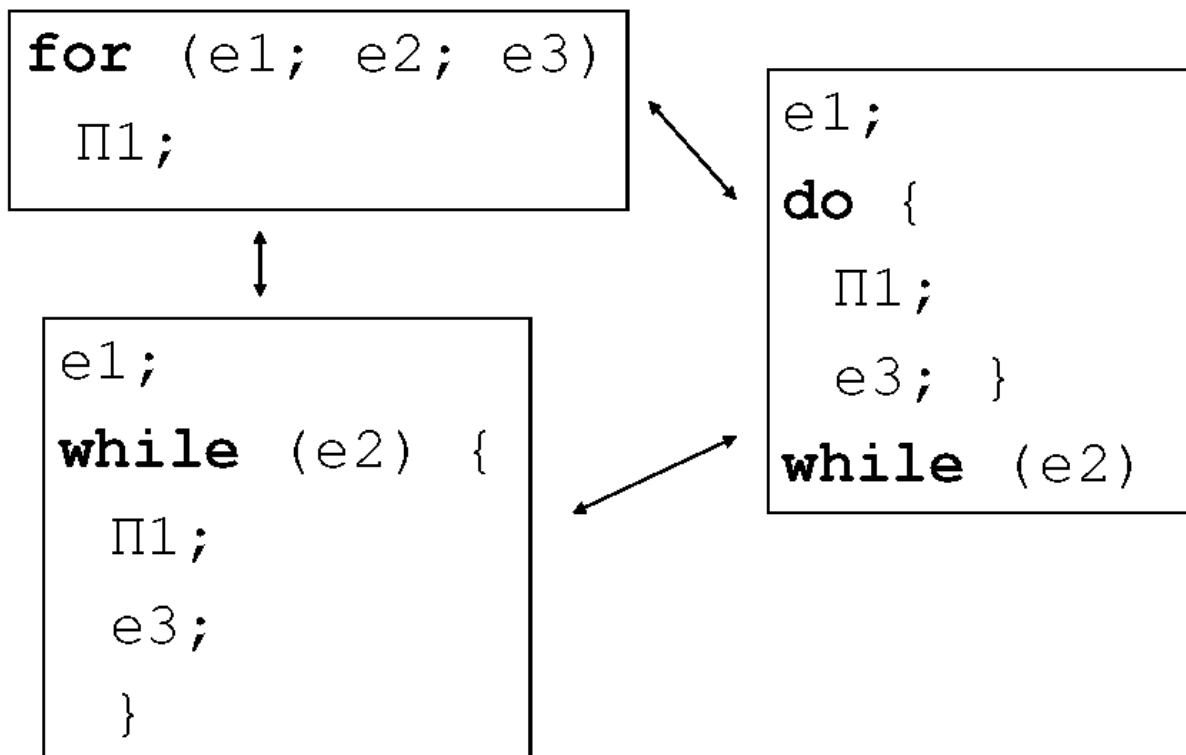
```
for (count=0; count<max_count; count++) {  
    printf("count is %d\n", count);  
    <άλλες προτάσεις>  
}
```



## 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

## Ισοδυναμία δομών επανάληψης





## Ένθετοι βρόχοι

```
#include <stdio.h>
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");
    }
}
```

## Διακλάδωση χωρίς συνθήκη

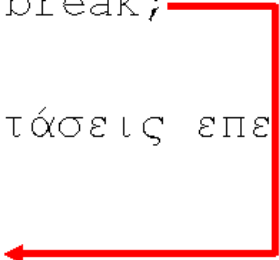
- Διαχείριση ειδικών περιπτώσεων σε προτάσεις επανάληψης
  - `break`
  - `continue`
- Ρητή διακλάδωση
  - `goto`

## Η πρόταση **break**

- Προκαλεί την έξοδο μόνο από τον πιο εσωτερικό βρόχο

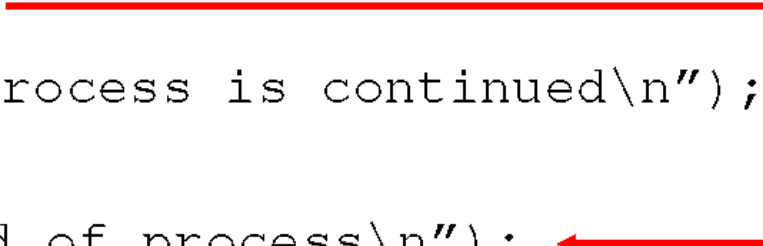
## Χρήση της `break`

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



## Παράδειγμα: break

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



## Η πρόταση **continue**

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

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

## Παράδειγμα: continue

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

## 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