Εισαγωγή στον Προγραμματισμό

Ενότητα C++ (75%) :

5η εβδομάδα

Παναγιώτης Τζουνάκης

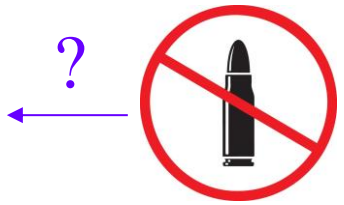
(βασισμένο στις σημειώσεις που ευγενικά προσέφερε ο καθηγητής κ. Γεώργιος Ραχώνης)

Χειμώνας 2026

Θυμίζουμε ότι με το επόμενο πρόγραμμα «υπολογίσαμε» τις λύσεις της εξίσωσης δευτέρου βαθμού:

```
#include <iostream>
#include <math.h>
using namespace std;
int main ( )
{
   float a, b, c, x1, x2;
   cout <<“Δώσε τους συντελεστές a, b, c”;
   cin >>a>>b>>c;
   x1=(-b+sqrt(pow(b,2)-4*a*c))/(2*a);
   x2=(-b-sqrt(pow(b,2)-4*a*c))/(2*a);
   cout <<“Οι λύσεις της εξίσωσης είναι: ”
        <<“\nx1=”<<x1<<“\nx2= ”<<x2;
   return 0;
}
```

demo



Neuman on *Mad* 30, published December 1956

Μπορείτε να βρείτε τις λογικές «αδυναμίες» του προγράμματος;

- Τι θα συμβεί αν το τρέξω εισάγοντας «0» για το a ;
- Τι θα συμβεί αν εισάγω 1 1 1 για τους συντελεστές a b c ;

Η δυνατότητα λήψης αποφάσεων στον προγραμματισμό

Η πρόταση **if**

Η απλούστερη μορφή:

Σύνταξη:

```
if (συνθήκη)
    πρόταση;
```

Παράδειγμα:

```
if (a>=1)
        cout << "a is >= than 1";
```





Η σύνθετη πρόταση if:

Σύνταξη:

```
if (συνθήκη)
{
    πρόταση 1;
    πρόταση 2;
        .
        .
        .
    πρόταση n;
}
```

Παράδειγμα W5.0

Να γραφεί πρόγραμμα που υπολογίζει τη τιμή της συνάρτησης:

$$Y = \begin{cases} 1/(x-2) & x \neq 2 \\ 10 & x = 2 \end{cases}$$

Το αντίστοιχο τμήμα πηγαίου κώδικα του προγράμματος θα ήταν το εξής :

```
if (x==2)
        y=10;
if (x!=2)
        y=1/(x-2);
cout<<“\nx=“<<x<<‘ ‘<<“y=“<<y;
```





Η πρόταση if -else:

Σύνταξη:

```
if (συνθήκη)
    πρόταση1;
else
    πρόταση 2;
```

Η γενική μορφή if -else:

```
if (συνθήκη)
{
    πρόταση 1;
    πρόταση 2;
        .
    πρόταση n;
}
else
{
    πρόταση 1;
    πρόταση 2;
        .
    πρόταση m;
}
```





Παράδειγμα

Το προηγούμενο πρόγραμμα με χρήση της if-else:

```
if (x==2)
    y=10;
else
    y=1/(x-2);
cout<<“x=“<<x<<‘ ‘<<“y=“<<y;
```





Η δυνατότητα πολλών εναλλακτικών συνθηκών:

```
if (συνθήκη 1)
{
   σύνολο προτάσεων 1;
}

else if (συνθήκη 2)
{
   σύνολο προτάσεων 2;
}
          .
          .
else
{
   σύνολο προτάσεων κ+1;
}
```





Παράδειγμα W5.1

Να γραφεί πρόγραμμα που θα υπολογίζει τις τιμές της συνάρτησης:

$$Y = \begin{cases} \dfrac{x^2+1}{x-1}, & x > 1 \\ 0, & x = 1 \\ \dfrac{x^{10}+3}{x-1}, & x < 1 \end{cases}$$





```
#include<iostream>
#include<math.h>
using namespace std;
int main( )
{
    float x,y;
    cout<<“Assign value to x:  ”;
    cin>>x;
    if (x>1)
          y=(pow(x,2)+1)/(x-1);
    else if (x<1)
          y=(pow(x,10)+3)/(x-1);
    else
          y=0;
    cout<<“\ny=”<<y;
    return 0;
}
```

Παράδειγμα W5.2:

Να γράψετε πρόγραμμα που θα υπολογίζει τις λύσεις της εξίσωσης δευτέρου βαθμού:

$$ax^2 + bx + c = 0$$

Στη συνέχεια δίνουμε μία λύση για το παράδειγμα αυτό. Μπορείτε να καταλάβετε γιατί το πρόγραμμα που παρουσιάζεται δεν είναι το αποδοτικότερο από πλευράς χρόνου; Προσπαθήστε να γράψετε ένα καλύτερο πρόγραμμα.





```
#include<iostream>
#include<math.h>
using namespace std;
int main( )
{
    float a,b,c,x,x1,x2;
    cout<<“Assign value to a:  ”;
    cin>>a;
    cout<<“Assign value to b:  ”;
    cin>>b;
    cout<<“Assign value to c:  ”;
    cin>>c;
```





```
if ((a==0)&&(b==0)&&(c==0))
        cout<<"\nThere are infinite solutions";
else if ((a==0)&&(b==0)&&(c!=0))
        cout<<"\nThere are no solutions";
else if ((a==0)&&(b!=0))
{
        x=-c/b;
        cout<<"\nThere is one solution  x= "<<x;
}
else if ((a!=0)&&((pow(b,2)-4*a*c)==0))
{
        x=-b/(2*a);
        cout<<"\nThere is one solution x= "<<x;
}
```





```
else if ((a!=0)&&((pow(b,2)-4*a*c)>0))
{
        x1=(-b+sqrt(pow(b,2)-4*a*c))/(2*a);
        x2=(-b-sqrt(pow(b,2)-4*a*c))/(2*a);
        cout<<"\nThere are two solutions";
        cout<<"\nx1="<<x1;
        cout<<"\nx2="<<x2;
}
else if ((a!=0)&&((pow(b,2)-4*a*c)<0))
{
        x1=-b/(2*a);
        x2=sqrt(fabs(pow(b,2)-4*a*c))/(2*a);
        cout<<"\nThe solutions are complex
                                    numbers:";
        cout<<'\n'<<x1<<"+("<<x2<<")i";
        cout<<'\n'<<x1<<"-("<<x2<<")i";
}
return 0;
}
```





Παράδειγμα W5.3:

Να γραφεί πρόγραμμα το οποίο θα δέχεται έναν ακέραιο αριθμό a, θα τον συγκρίνει με τον αριθμό 1000 και θα επιστέφει ένα σχετικό μήνυμα, πχ «Ο a είναι μικρότερος από το 1000».





Παράδειγμα W5.4:

Να γραφεί πρόγραμμα το οποίο θα δέχεται έναν ακέραιο αριθμό a, θα τον συγκρίνει με τους αριθμούς 1000 και 2000 και θα επιστέφει ένα σχετικό μήνυμα, π.χ. «Ο a είναι μεγαλύτερος από το 1000 και μικρότερος από το 2000».





Άσκηση W5.1

Να γραφεί πρόγραμμα το οποίο θα δέχεται 6 αριθμούς a, b, c, d, e, f και στη συνέχεια θα επιλύει το παρακάτω σύστημα :

$$ax + by = c$$

$$dx + ey = f$$





Εντολές επανάληψης

Παράδειγμα:

Να βρεθεί το παραγοντικό του φυσικού αριθμού $n > 0$:

$$n! = 1 \cdot 2 \cdot 3 \cdots n$$

Η πρόταση for

Η απλούστερη μορφή

Σύνταξη

```
for (μεταβλητή=αρχική τιμή; συνθήκη;
                μεταβολή  της τιμής της μεταβλητής)
        πρόταση;
```

Παράδειγμα:

```
for (a=1; a<=100; a=a+1)
      cout<<“\n This is the :”<<a<<“time”;
```





Παράδειγμα:

Έστω ότι θέλουμε να βρούμε το μέγιστο κοινό διαιρέτη (ΜΚΔ) δύο θετικών ακεραίων αριθμών α, β με τη βοήθεια του τύπου

$$\text{ΜΚΔ}(\alpha,\beta)=\begin{cases}\text{ΜΚΔ}(\alpha-\beta,\beta) & \text{αν } \alpha>\beta\\ \text{ΜΚΔ}(\alpha,\beta-\alpha) & \text{αν } \alpha<\beta\\ \alpha & \text{αν } \alpha=\beta\end{cases}$$

Η παραπάνω διαδικασία θα πρέπει να επαναληφθεί ένα πλήθος φορών που δε γνωρίζουμε, έως ότου οι α, β να γίνουν ίσοι.

Είναι βολική η πρόταση for για αυτό το πρόβλημα;





Παράδειγμα W5.5:

Να βρεθεί το παραγοντικό του φυσικού αριθμού *n*.

```
#include<iostream>
using namespace std;
int main( )
{
int n, a;
long int factorial;
cout<<“\nAssign value to n”;
cin>>n;
if (n==0)
        cout<<“\n The factorial of 0 is 1”;
else
{
        factorial=1;
        for (a=1; a<=n; a=a+1)
                factorial=factorial*a;
        cout<<“\nThe factorial of ”<<n<<“is”
                        <<factorial;
}
return 0;
}
```





```
#include<iostream>
using namespace std;
int main( )
{
int n, a;
long int factorial;
cout<<“\nΔώσε το n”;
cin>>n;
if (n==0)
        cout<<“\nThe factorial of 0 is ”;
else
{
        factorial=1;
        for (a=1; a<=n; a++)
                factorial=factorial*a;
        cout<<“\nThe factorial of “<<n
                        <<“is”<<factorial;
}
return 0;
}
```

Τι παρατηρείτε;





Η χρήση του a++ είναι ισοδύναμη με το a=a+1.

Έτσι κάθε φορά που εκτελείται η εντολή που συνοδεύει την πρόταση, η τιμή του a αυξάνει κατά 1.

Όμοια μπορούμε να χρησιμοποιήσουμε το a+=2 που σημαίνει ότι η τιμή του a αυξάνει κατά 2.

Παράδειγμα W5.6:

Να γράψετε πρόγραμμα που θα τυπώνει όλους τους περιττούς αριθμούς που βρίσκονται στο διάστημα [2,100].





```
#include<iostream>
using namespace std;
int main()
{
int a;
for (a=3; a<100; a+=2)
   cout<<“   ”<<a;
return 0;
}
```

Ομοίως μπορούμε αντί του a=a-1 να χρησιμοποιήσουμε το a--





Η πρόταση for

Η γενική μορφή

Σύνταξη

```
for (μεταβλητή=αρχική τιμή;
                συνθήκη; μεταβολή  της τιμής
                                        μεταβλητής)
{
        πρόταση  1;
        πρόταση  2;
            .
            .
        πρόταση  n;
}
```





Παράδειγμα W5.7:

Να γράψετε πρόγραμμα που διαβάζει ένα φυσικό αριθμό *n* και *n* ακέραιους αριθμούς, και θα υπολογίζει:

1) το άθροισμα των ακεραίων που βρίσκονται εκτός του διαστήματος [-30, 30],

2) το γινόμενο των ακεραίων που βρίσκονται στο διάστημα [-3, 3], και

3) το πλήθος των αρτίων.





```
#include<iostream>
using namespace std;
int main()
{
int n, number, sum, product, plithos, i ,j, k;
cout<<"\nAssign value to n :";
cin>>n;
sum=0;
product=1;
plithos=0;
j=0;
k=0;
```





```
for (i=1; i<=n; i++)
{
   cout<<"\nAssign value to the "<<i<<"number: ";
   cin>>number;
   if ((number<-30)||(number>30))
   {
        sum=sum+number;
        j=j+1;
   }
   if ((number>=-3)&&(number<=3))
   {
        product=product*number;
        k=k+1;
}
```





```
    if ((number%2)==0)
        plithos=plithos+1;          //ή plithos++
}
if (j==0)
    cout<<"\nThere are no integers
                        out of the interval [-30,30]";
else
    cout<<"\nThe sum is: "<<sum;
if (k==0)
    cout<<"\nThere are no integers
                        in the interval [-3,3]";
else
    cout<<"\nThe product is: "<<product;
cout<<"\nThe number of even numbers is: "<<plithos;
return 0;
}
```





Περισσότερες πρακτικές συμβουλές

1. Σε σύνθετες λογικές εκφράσεις προσέχουμε την ιεραρχία των τελεστών. Καλύτερα αχρείαστες παρενθέσεις, παρά λάθη, αβλεψίες και δυσανάγνωστος κώδικας…
2. Πριν βάλουμε φωλιασμένα (nested) IFs, ελέγχουμε μήπως ο ίδιος υπολογισμός μπορεί να υλοποιηθεί με σύνθετες λογικές εκφράσεις, ή IF-THEN-ELSE ή κάποια άλλη διαθέσιμη εντολή επιλογής της γλώσσας.
3. Η εντολή FOR χρησιμοποιείται μόνο για προκαθορισμένο αριθμό επαναλήψεων, ο οποίος πρέπει να είναι ήδη καθορισμένος ή υπολογίσιμος.
4. Αποφύγουμε την αλλαγή της αρχικής/τελικής τιμής, του βήματος ή της μεταβλητής που ελέγχει την επανάληψη μέσα σε ένα βρόχο FOR, ακόμα κι αν η γλώσσα προγραμματισμού το επιτρέπει, διότι το πρόγραμμα γίνεται δυσνόητο και πιθανά λανθασμένο.

Υπενθύμιση!





Παράδειγμα W5.8

Να βρεθούν οι ακέραιοι αριθμοί *a, b, c* (Πυθαγόρειοι αριθμοί) που ανήκουν στο διάστημα [1,50] και ικανοποιούν τη σχέση:

$$a^2 + b^2 = c^2$$

```
#include<iostream>
#include<math.h>
using namespace std;
int main()
{
int a,b,c;
```

```
for (a=1;a<=50;a++)
{
    for (b=1;b<=50;b++)
    {
        for (c=1;c<=50;c++)
        {
            if (pow(a,2)+pow(b,2)==pow(c,2))
                cout<<a<<“  “<<b<<“  “<<c<<endl;
        }
    }
}
return 0;
}
```





Παράδειγμα W5.9

Να γραφεί πρόγραμμα που υπολογίζει το παρακάτω γινόμενο

$$P = \prod_{n=1}^{30} \frac{(n+1)^2}{n(n+2)}$$

Παράδειγμα W5.10

Να βρεθούν οι ακέραιοι αριθμοί *a, b, c* που ανήκουν στο διάστημα [1,50] με a<b και ικανοποιούν τη σχέση :

$$a^2 + b^2 = c^2$$

Παράδειγμα W5.11

Να βρεθούν όλες οι τετράδες ακέραιων αριθμών *a, b, c, d* που ανήκουν στο διάστημα [1,1000], είναι διαφορετικοί μεταξύ τους και ικανοποιούν τη σχέση:

$$a^2+ b^2+c^2 =d^2$$





Άσκηση W5.2

Να γράψετε πρόγραμμα που θα διαβάζει *n* ακέραιους αριθμούς και στη συνέχεια θα τυπώνει:

α) το πλήθος των αριθμών που βρίσκονται στο διάστημα [-300,300],

β) το γινόμενο των αριθμών που είναι διάφοροι του μηδενός και βρίσκονται στο διάστημα [-45,55],

γ) το άθροισμα των αριθμών που είναι πολλαπλάσια του 5,

δ) τον μέσο όρο των άρτιων αριθμών που βρίσκονται έξω από το διάστημα [-100,200].



