ΣΥΝΑΡΤΗΣΕΙΣ

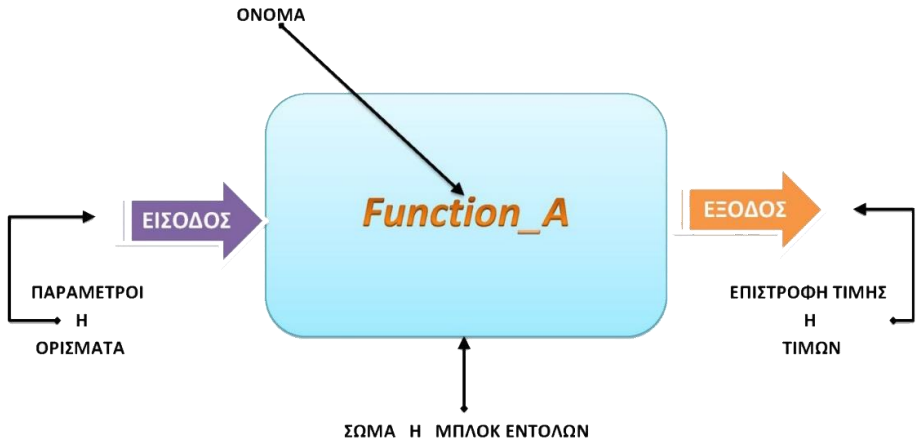
ΟΡΙΣΜΟΣ

Συνάρτηση είναι ένα μέρος του προγράμματος που εκτελεί μία συγκεκριμένη λειτουργία. Μέσα σε ένα πρόγραμμα της *C++* μπορεί να υπάρχουν αυτόνομες συναρτήσεις ή συναρτήσεις που περιλαμβάνονται μέσα σε *κλάσεις*. Η συνάρτηση περιέχει εντολές μέσα στο σώμα της (μπλοκ εντολών).
Μια συνάρτηση:

- καλείται μέσα στην βασική συνάρτηση της *C++ (main)* ή και από άλλες συναρτήσεις,
- αλλά και η ίδια καλεί άλλες συναρτήσεις.

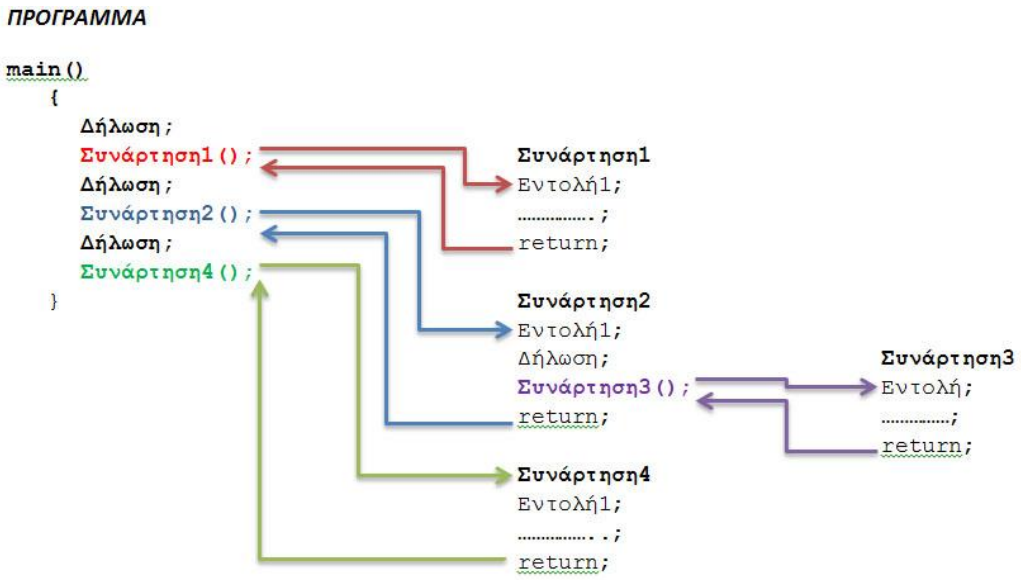
ΑΠΕΙΚΟΝΙΣΗ ΣΥΝΑΡΤΗΣΗΣ

Πιο κάτω φαίνεται σχηματικά μια συνάρτηση.



Όπως φαίνεται στο σχήμα η συνάρτηση έχει είσοδο που είναι οι παράμετροι (μεταβλητές εισόδου), έξοδο που είναι η τιμή που επιστρέφει η συνάρτηση στο σημείο που κλήθηκε και σώμα (μπλοκ εντολών) που περιλαμβάνει τον κώδικα των εντολών που διατρέχει για να ολοκληρώσει την δουλειά της.

ΣΧΗΜΑΤΙΚΗ ΠΑΡΑΣΤΑΣΗ ΚΛΗΣΕΩΝ ΣΥΝΑΡΤΗΣΕΩΝ



Στη πιο πάνω εικόνα το πρόγραμμα καλεί την πρώτη συνάρτηση *Συνάρτηση1()*. Ο έλεγχος φεύγει από το πρόγραμμα και εκτελεί τις εντολές της συνάρτησης. Η *Συνάρτηση1()* επιστρέφει τιμή στο πρόγραμμα. Το πρόγραμμα εν συνεχεία καλεί την *Συνάρτηση2()*. Ο έλεγχος πηγαίνει στην συνάρτηση και εκτελεί τις εντολές της. Μια από τις εντολές της συνάρτησης είναι και η κλήση της *Συνάρτηση3()*, ο έλεγχος πηγαίνει σε αυτήν και εκτελεί το σώμα των εντολών της. Αφού ολοκληρωθεί η *Συνάρτηση3()* επιστρέφει τιμή στην *Συνάρτηση2()* η οποία αφού ολοκληρωθεί επιστρέφει με την σειρά της τιμή στο πρόγραμμα. Κατόπιν με τον ίδιο τρόπο καλείται από το πρόγραμμα η *Συνάρτηση4()*.

Η ΣΥΝΑΡΤΗΣΗ ΜΕΣΑ ΣΤΟ ΠΡΟΓΡΑΜΜΑ

Οι συναρτήσεις μέσα σε ένα σημείο του προγράμματος *καλούνται* αφού πρώτα έχουν *δηλωθεί* στην αρχή του προγράμματος και έχουν *οριστεί* συνήθως μετά από την *main()*. Άρα για να χρησιμοποιήσουμε μια συνάρτηση πρέπει να κάνουμε τα ακόλουθα:

- ***Δήλωση***: Επισημαίνουμε στον Compiler ότι θα χρησιμοποιήσουμε στο πρόγραμμα τη συνάρτηση. Η δήλωση λέγεται και *Πρωτότυπο*.
- ***Ορισμός***: Περιγράφεται η λειτουργία της συνάρτησης με άλλα λόγια γράφουμε τον κώδικα της συνάρτησης.
- ***Κλήση***: Καλούμε την συνάρτηση ή αλλιώς εκτελούμε τις εντολές της (τον κώδικα).

ΔΗΛΩΣΗ ΣΥΝΑΡΤΗΣΗΣ ΣΤΟ ΠΡΟΓΡΑΜΜΑ

Η δήλωση μιας συνάρτησης στο πρόγραμμα συντάσσεται ως εξής:

```
int function(int param1, int param2);
```

όπου:

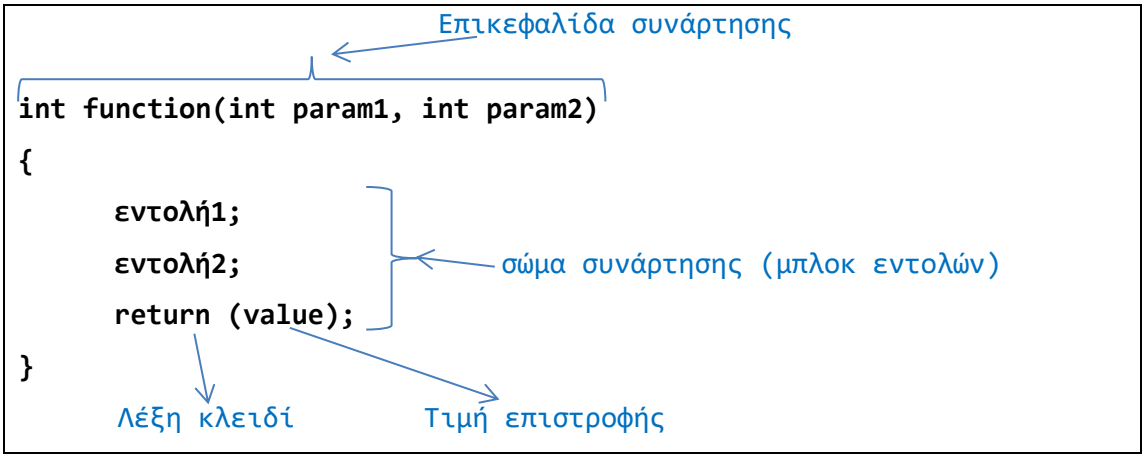
int -> Τύπος επιστρεφόμενης τιμής
function -> Όνομα συνάρτησης
int param1 -> Τύπος και Όνομα παραμέτρου 1
int param2 -> Τύπος και Όνομα παραμέτρου 2
; -> τερματικό εντολής

Το ***Πρότυπο*** (ή αλλιώς επικεφαλίδα της συνάρτησης) έχει όνομα, εισόδους (παράμετροι με τύπο) αλλά και έξοδο (τύπος επιστρεφόμενης τιμής). Το *Πρότυπο* της συνάρτησης είναι μία εντολή άρα θα τελειώνει με ***τερματικό εντολής (;).***

Σημ. Στη *δήλωση* (Πρότυπο) και τη *κλήση* μιας συνάρτησης χρησιμοποιούμε ***πάντα*** τερματικό εντολής, στον *ορισμό* της ποτέ.

ΟΡΙΣΜΟΣ ΣΥΝΑΡΤΗΣΗΣ ΣΤΟ ΠΡΟΓΡΑΜΜΑ

Πιο κάτω βλέπουμε τον ορισμό μιας συνάρτησης στην C++



Στον ορισμό της συνάρτησης *(δηλαδή στην περιγραφή της λειτουργίας της συνάρτησης)* χρησιμοποιούμε την επικεφαλίδα της συνάρτησης, δηλαδή: α) το όνομά της, β) τις παραμέτρους στο όρισμά της και γ) μέσα σε ένα ζεύγος αγκίστρων, τις εντολές της συνάρτησης καθώς και τις τοπικές μεταβλητές που χρησιμοποιεί. Επίσης την λέξη ***return*** αν η συνάρτηση πρέπει να επιστρέψει μία τιμή. Στον ορισμό της συνάρτησης *δεν* χρησιμοποιούμε το ερωτηματικό.

Σημ. Εάν δεν χρησιμοποιήσουμε πρωτότυπο τότε θα πρέπει να ορίζουμε την συνάρτηση πριν από την main() (πριν από την κλήση της).

ΚΛΗΣΗ ΣΥΝΑΡΤΗΣΗΣ ΣΤΟ ΠΡΟΓΡΑΜΜΑ

Κατά την κλήση μιας συνάρτησης μέσα σε ένα πρόγραμμα χρησιμοποιούνται:

α) το όνομά της, β) οι πραγματικές παράμετροι εάν υπάρχουν και γ) το τερματικό εντολής.

Δεν χρησιμοποιούμε τον τύπο επιστροφής.

```
void main()
{
   Εντολή1;
   Εντολή2;
   function (param1, param2); // κλήση συνάρτησης
   Εντολή3;
}
```

ΠΕΡΙΠΤΩΣΕΙΣ ΧΡΗΣΗΣ ΣΥΝΑΡΤΗΣΕΩΝ

1. **ΔΗΛΩΣΗ ΤΟΠΙΚΩΝ ΜΕΤΑΒΛΗΤΩΝ *(συνάρτηση με επιστροφή τιμής)***

Το παρακάτω πρόγραμμα υπολογίζει το άθροισμα 2 αριθμών.

```
#include <iostream>
using namespace std;

int athroisma(int k, int l); // dhlosh prototypoy

int main()
{
  int a,b,c;
  cout << "dose 1o arithmo:";
  cin >> a;
  cout << "dose 2o arithmo:";
  cin >> b;
  c = athroisma(a,b); // klhsh ths synarthshs
  cout <<"to athroisma tou "<<a<<"+"<<b<<" einai:"<<c<<endl;
  return 0;
} //telos ths main

int athroisma(int k, int l) // orismos synarthshs
{
  int m; //topikh metablhth
  m = k + l;
  return m ; //epistrofh timhs
} // telos synarthshs
```

2. ΔΗΛΩΣΗ ΜΕΤΑΒΛΗΤΩΝ ΜΕ ΤΟ ΙΔΙΟ ΟΝΟΜΑ *(συνάρτηση με επιστροφή τιμής)*

Το παρακάτω πρόγραμμα μετατρέπει τους βαθμούς Fahrenheit σε βαθμούς Celsius με την βοήθεια της σχέσης $^{o}C = (^{o}F - 32) * \frac{5}{9}$.

Δείτε τη έξοδο του προγράμματος στο σημείο βρασμού του νερού (212^{O} F ή 100^{O} C), στο σημείο ψύξης του νερού (32^{O} F ή 0^{O} C) και σε ένα τυχαίο σημείο.

```
#include <iostream>
using namespace std;

float ConvTemp(float); // dhlosh prototypoy

int main()
{
   float FarTemp, CelTemp;
   cout << "dose tous vathmous Fahrenheit: ";
   cin >> FarTemp;
   CelTemp = ConvTemp(FarTemp); // klhsh ths synarthshs
   cout << "\nh thermokrasia se Celsius einai: ";
   cout << CelTemp << endl;
 return 0;
} //telos ths main

float ConvTemp(float FarTemp) // orismos synarthshs
{
   float CelTemp; //topikh metablhth
   CelTemp = ((FarTemp - 32) * 5) / 9;
   return CelTemp; //epistrofh timhs
} // telos synarthshs
```

Παρατήρηση: Η μεταβλητή **CelTemp** που βρίσκεται μέσα στην συνάρτηση **CelTemp** δεν είναι ίδια με τη μεταβλητή **CelTemp** που βρίσκεται μέσα στην **main** παρόλο που έχουν το ίδιο όνομα. Επίσης στην δήλωση της συνάρτησης **CelTemp** δηλώνουμε την παράμετρο **FarTemp** που την χρησιμοποιούμε στον τύπο της μετατροπής μέσα στην **CelTemp.**

3. ΣΥΝΑΡΤΗΣΗ ΜΕ ΑΝΑΔΡΟΜΗ *(συνάρτηση με επιστροφή τιμής)*

Το παρακάτω πρόγραμμα υπολογίζει το παραγοντικό ενός αριθμού λαμβάνοντας υπόψιν ότι:

- n=0 τότε **n!=1** και αν
- n>0 τότε **n!=n*(n-1)**

```
#include <iostream>
using namespace std;

long paragontiko(long n); //dhlosh prototypou

int main()
{
  long arithmos;
  cout<<"dose enan arithmo: "<<endl;
  cin>>arithmos;

  // exodos me klhsh ths synarthshs
```

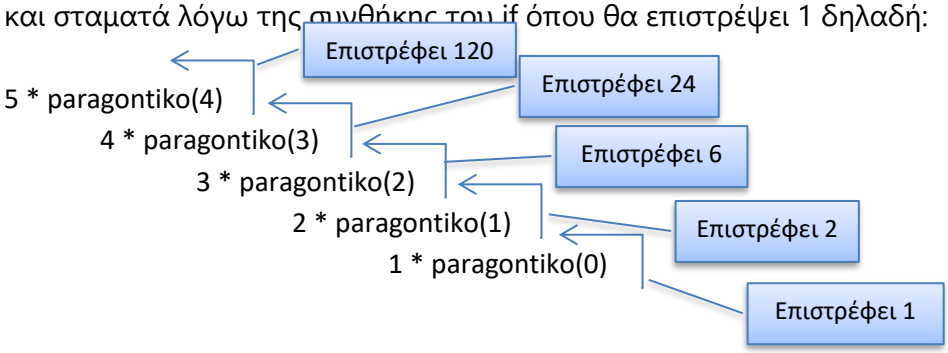
```
  cout<<arithmos<<"!=" <<paragontiko(arithmos)<<endl;

  return 0;
}//telos ths main

long paragontiko(long n) // orismos synarthshs
{
  if (n>1)
   return (n*paragontiko(n-1));
  else
   return 1;
} //telos synarthshs
```

Σύντομη εξήγηση του προγράμματος.

Πχ. Εάν το n=5 τότε η συνάρτηση καλεί τον εαυτό της με όρισμα n-1 μέχρι το παραγοντικό του 0 όπου και σταματά λόγω της συνθήκης του if όπου θα επιστρέψει 1 δηλαδή:



4. ΣΥΝΑΡΤΗΣΗ ΜΕ ΕΠΑΝΑΛΗΨΗ *(συνάρτηση με επιστροφή τιμής)*

Το παρακάτω πρόγραμμα υπολογίζει την **δύναμη** ενός αριθμού. Ο αριθμός και η δύναμη εισάγεται από το χρήστη.

Λάβετε υπόψη ότι: $\mathbf{x^n = x*x*x*.....*x}$, n φορές.

```
#include <iostream>
using namespace std;
int power(int x, int n); // dhlosh synarthshs
int main()
{
      int a,b,dynamh;
      cout<<"dose ton arithmo: ";
      cin>>a;
      cout<<"dose thn dynamh: ";
      cin>>b;
      dynamh = power(a,b); // klhsh ths synarthshs
      cout<<"to "<<a<<" sthn "<<b<<" mas kanei: "<<dynamh<<endl;
      return 0;
}//telos ths main

int power(int x, int n) // orismos synarthshs
{
   int i,p;
   p = 1;
   for (i=1; i<=n; i++)
   p = p*x;
   return p;
} //telos synarthshs
```

5. ΣΥΝΑΡΤΗΣΗ ΧΩΡΙΣ ΕΠΙΣΤΡΟΦΗ ΤΙΜΗΣ

Το παρακάτω πρόγραμμα υπολογίζει το πηλίκο 2 αριθμών.

```
#include <iostream>
using namespace std;

void phliko(int k, int l); // dhlosh synarthshs

int main()
{
   int a,b;
   cout <<"dose 1o arithmo:";
   cin >> a;
   cout <<"dose 2o arithmo:";
   cin >> b;
   phliko (a,b); // klhsh ths synarthshs
   return 0;
} // telos ths main

void phliko (int k, int l) // orismos synarthshs
{
   int m; // topikh metablhth
   m = k / l;
   cout <<"to phliko "<<k<<"/"<<l<<" einai:"<<m<<endl;
} //telos synarthshs
```

Παρατήρηση: Η συνάρτηση πηλίκο υπολογίζει αλλά και εμφανίζει το πηλίκο των μεταβλητών.

6. ΚΛΗΣΗ ΣΥΝΑΡΤΗΣΗΣ ΑΠΟ ΣΥΝΑΡΤΗΣΗ

Το παρακάτω πρόγραμμα κάνει κλήση σε μία συνάρτηση που έχουμε φτιάξει εμείς χωρίς πρωτίστως να την έχουμε δηλώσει (πρότυπο).

```
# include <iostream>
using namespace std;

/* H synarthsh Demo
typonei ena xrhsimo mhnyma */
void Demo()
{
      cout<<"Hello from Demo\n";
} //telos synarthshs

/* H synarthsh main
typonei ena mhnyma meta kalei thn Demo
meta typonei ena deytero mynhma */
int main()
{
      cout<<"Hi from main\n";
      Demo(); //klhsh synarthshs
      cout<<"We are back in main\n";
      return 0;
} //telos ths main
```

Η έξοδος του προγράμματος είναι:

Hi from main
Hello from Demo

We are back in main
Press any key to continue

Σύντομη εξήγηση του προγράμματος.

Η συνάρτηση ***Demo()*** καλείται από την ***main()*** για να τυπώσει ένα μήνυμα. Το πρόγραμμα εκτελείται γραμμή προς γραμμή έως ότου κληθεί μία συνάρτηση. Τότε το πρόγραμμα διακλαδώνεται για να εκτελεστεί η συνάρτηση και κατόπιν επιστρέφει από εκεί που σταμάτησε για να συνεχίσει την εκτέλεση του.

7. ΠΕΡΑΣΜΑ ΟΡΙΣΜΑΤΩΝ ΜΕ ΤΙΜΗ *(παράμετροι ως τοπικές μεταβλητές)*

Το παρακάτω πρόγραμμα κάνει αλλαγή θέσης 2 τιμών τοπικών μεταβλητών χωρίς να επηρεάζονται οι αρχικές μεταβλητές.

```
#include <iostream>
using namespace std;

 void swap(int x, int y); //dhlosh prototypou

 int main()
 {
    int a,b;
      cout<<"dose a: ";
      cin>>a;
      cout<<"dose b: ";
      cin>>b;
     cout<<"Main. Prin to swap, a: "<<a<<" b: "<<b<<"\n";

     swap(a,b); //klhsh synarthshs
     cout<<"Main. Meta to swap, a: "<<a<<" b: "<<b<<"\n";
     return 0;
  } //telos ths main

  void swap(int x, int y)
  {
     int temp;
     cout<<"Swap. Prin to swap, x: "<<x<<" y: "<<y<<"\n";
     temp=x;
     x=y;
     y=temp;
     cout<<"Swap. Meta to swap, x: "<<x<<" y: "<<y<<"\n";
  } //telos synarthshs
```

Σύντομη εξήγηση του προγράμματος.

Εδώ παρατηρούμε ότι οι τιμές των ***a*** και ***b*** δεν αλλάζουν, παρόλο που αλλάζουν οι τιμές των τοπικών μεταβλητών ***x*** και ***y*** και αυτό γιατί η συνάρτηση ***swap()*** μεταβιβάζει τις τιμές των a και b στις x και y για να τις χρησιμοποιήσει. Έτσι οι τιμές των a και b μένουν ανέπαφες.

8. ΠΕΡΑΣΜΑ ΟΡΙΣΜΑΤΩΝ ΜΕ ΑΝΑΦΟΡΑ

Εάν θέλουμε οι τιμές των a και b να πάρουν τις νέες τιμές των x και y μετά την εκτέλεση της συνάρτησης swap τότε πρέπει να κάνουμε **κλήση με αναφορά**

void swap(int&x, int&y); δηλαδή δίνουμε το ψευδώνυμο x στην a και y στην b. Έτσι όταν καλείται η συνάρτηση δεν μεταβιβάζουμε τιμή της αρχικής μεταβλητής, αλλά την αναφορά στην αρχική μεταβλητή.

```
#include <iostream>
using namespace std;

void swap(int &x, int &y); //dhlosh prototypou

int main()
{
   int a,b;
   cout<<"dose a: ";
   cin>>a;
   cout<<"dose b: ";
   cin>>b;
   cout<<"Main. Prin to swap, a: "<<a<<" b: "<<b<<"\n";

   swap(a,b); //klhsh synarthshs
   cout<<"Main. Meta to swap, a: "<<a<<" b: "<<b<<"\n";
   return 0;
 } //telos ths main

void swap(int &x, int &y)
{
   int temp;
   cout<<"Swap. Prin to swap, x: "<<x<<" y: "<<y<<"\n";
   temp=x;
   x=y;
   y=temp;
   cout<<"Swap. Meta to swap, x: "<<x<<" y: "<<y<<"\n";
} //telos synarthshs
```

9. ΥΠΕΡΦΟΡΤΩΣΗ ΣΥΝΑΡΤΗΣΕΩΝ

Να γραφτεί ένα πρόγραμμα που να χρησιμοποιεί μία συνάρτηση με υπερφόρτωση ώστε να κάνει τα εξής:

1. Να τυπώνει στην οθόνη 50 μηδενικά (0),
2. Να τυπώνει στην οθόνη 50 χαρακτήρες που θα καθορίζονται από την κλήση της συνάρτησης και
3. Να τυπώνει στην οθόνη χαρακτήρες που ο χαρακτήρας αλλά και ο αριθμός των χαρακτήρων να καθορίζονται στην κλήση της συνάρτησης.

Χωρίς την υπερφόρτωση θα έπρεπε να χρησιμοποιούσαμε τρεις διαφορετικές συναρτήσεις με ονόματα : ***printZero(), printCharacter(), printCharFunctionDefine()*.**

Με την υπερφόρτωση συνάρτησης χρησιμοποιούμε μία συνάρτηση με διαφορετικές παραμέτρους (ορίσματα) για κάθε διαφορετική δουλειά, πράγμα ευκολότερο από το να θυμόμαστε τα ονόματα τριών διαφορετικών συναρτήσεων.

```
#include <iostream>
using namespace std;

void printFunc(); //dhlosh prototypou 1
void printFunc(char ch); //dhlosh prototypou 2
void printFunc(char ch, int n); //dhlosh prototypou 3
int main()
{
      printFunc(); //klhsh synarthshs 1
      printFunc('/'); //klhsh synarthshs 2
      printFunc('%',40); //klhsh synarthshs 3
```

```
} //telos ths main

void printFunc() //orismos synarthshs 1
{
	int i;
	for (i=0; i<50; i++)
		cout << "0";
	cout << endl;
} //telos synarthshs

void printFunc(char ch) //orismos synarthshs 2
{
	int i;
	for (i=0; i<50; i++)
		cout << ch;
	cout << endl;
} //telos synarthshs

void printFunc(char ch, int n) //orismos synarthshs 3
{
	int i;
	for (i=0; i<n; i++)
		cout << ch;
	cout << endl;
} //telos synarthshs
```

10. ΥΠΕΡΦΟΡΤΩΣΗ ΜΕ ΠΡΟΚΑΘΟΡΙΣΜΕΝΕΣ ΤΙΜΕΣ ΠΑΡΑΜΕΤΡΩΝ

Το παρακάτω πρόγραμμα χρησιμοποιεί μία δήλωση συνάρτησης με προκαθορισμένες τιμές παραμέτρων για να κάνει 3 εργασίες.

```
#include <iostream>
using namespace std;

void printFunc(char ch='-', int n=50);

int main()
{
	//yperfortwsh synarthsewn
	printFunc();
	printFunc('/');
	printFunc('%',40);
	return 0;
}

void printFunc(char ch, int n)
{
	int i;
	for (i=0; i<n; i++)
		cout << ch;
	cout << endl;
}
```

11. ΧΡΗΣΗ ΤΟΠΙΚΩΝ ΜΕΤΑΒΛΗΤΩΝ

Το παρακάτω πρόγραμμα δείχνει την συμπεριφορά τοπικών μεταβλητών σε διαφορετικές συναρτήσεις.

```
#include <iostream>
using namespace std;

int main()
{
   int x, y; //τοπικές μεταβλητές
   x = 3;
   y = tetragono(x);
   cout << "to y ths main einai: " << y << endl; //2
   cout << "to tetragono toy" << x << "einai:" << y << endl; //3
}

int tetragono(int a)
{
   int y = 10; //τοπική μεταβλητή
   cout << "to y ths tetragono einai: " << y << endl; //1
   return(a*a);
}
```

ΕΞΟΔΟΣ ΠΡΟΓΡΑΜΜΑΤΟΣ	**Σημ**. Εδώ άλλη τιμή έχει η μεταβλητή ***y*** μέσα στην ***main()*** και άλλη μέσα στην συνάρτηση ***tetragono().***
To y ths tetragono einai 10 //1 To y ths main einai 9 //2 To tetragono toy 3 einai 9 //3	

12. ΧΡΗΣΗ ΚΑΘΟΛΙΚΩΝ ΜΕΤΑΒΛΗΤΩΝ

Το ίδιο πρόγραμμα δείχνει την συμπεριφορά καθολικών μεταβλητών σε διαφορετικές συναρτήσεις.

```
#include <iostream>
using namespace std;

int tetragono(int a);
int y = 25; //Καθολική μεταβλητή

int main()
{
   int x;
   x = 3;
   cout << "x=" <<  x << " " << "y=" << y << endl;
   y = tetragono(x); //αλλαγή τιμής της μεταβλητής y
   cout << "to y ths main einai: " << y << endl; //2
}

int tetragono(int a)
{
   //εκτύπωση της τιμής της καθολικής μεταβλητής y
   cout << "to y ths tetragono einai: " << y << endl; //1
   return(a*a);
}
```

ΕΞΟΔΟΣ ΠΡΟΓΡΑΜΜΑΤΟΣ	**Σημ**. Εδώ η μεταβλητή ***y*** έχει γίνει καθολική δηλαδή έχει οριστεί έξω από την ***main***(). Η τιμή της αλλάζει μέσα στην ***main***() αλλά δίνει την τιμή της στην κλήση της συνάρτησης ***tetragono().***
x=3, **y=25** To y ths tetragono einai 10 //1 **To y ths main einai 25** //2	