Η ΓΛΩΣΣΑ C++

Μάθημα 1:

Εισαγωγή στην C++ και σχέση με την γλώσσα C

Δημήτρης Ψούνης

www.psounis.gr

Περιεχόμενα Μαθήματος

Α. Θεωρία

1. Η Γλώσσα C++

1. Γενικά

Η Γλώσσα Προγραμματισμού C++

- Είναι μια επέκταση της γλώσσας C με
  - Την οργάνωση των δεδομένων και των συναρτήσεων σε μία ενιαία οντότητα που ονομάζεται **κλάση (class)**
  - Οργάνωση των δεδομένων και των συναρτήσεων της κλάσης, ώστε να επιτυγχάνεται απόκρυψη πληροφορίας: **ενθυλάκωση (encapsulation)**
  - Η δυνατότητα των κλάσεων να **κληρονομούνται (inheritance)** από άλλες κλάσεις και να προσθέτουν την δική τους λειτουργικότητα
  - Η δυνατότητα συναρτήσεων και κλάσεων να είναι **πολυμορφικές (polymorphism)**
    - Ένα όνομα θα μπορεί να κάνει διαφορετικές ενέργειες
- Με πολλές καινούργιες βιβλιοθήκες, οι οποίες κάνουν πιο αποδοτικό τον προγραμματισμό μεγαλύτερων projects.

Παράδειγμα:

Θέλω να κατασκευάσω ένα παιχνίδι, το οποίο θα έχει για ήρωα τον σκύλο «Πίκο» (μικρόσωμο) και τον σκύλο «Αζόρ» (μεγαλόσωμο)

- Θα ορίσω μία **κλάση** με όνομα σκύλος με όλη τη συμπεριφορά του σκύλου
- Η ενέργεια «γάβγισε» θα είναι δημόσια, ενώ η ενέργεια «εκνευρισμός» θα είναι εσωτερική (**ενθυλάκωση**)
- Την κλάση αυτή θα την **κληρονομούν** δύο υπό-κλάσεις οι οποίες θα εξειδικεύουν τα χαρακτηριστικά των μεγαλόσωμων και μικρόσωμων σκύλων.
  - Ο Πίκο θα είναι ένα **αντικείμενο** της κλάσης «Μικρόσωμος σκύλος»
- Η κλάση θα έχει μία μέθοδο «γάβγισε», που θα αντιδρά διαφορετικά, αν ο σκύλος γαβγίζει μία γάτα ή έναν άνθρωπο (**πολυμορφισμός**)

Α. Θεωρία

1. Η Γλώσσα C++

2. Ιστορία – Εκδόσεις C++

- 1979 **«C with classes»** από τον Bjarne Stroustroup
  - στα εργαστήρια της Bell και της AT & T
- 1984 Μετονομασία σε **«C++»** με αρκετές προσθήκες
- 1989 2η έκδοση της C++ με πολλές προσθήκες ακόμη στη γλώσσα
- 1998 Πρώτη standard έκδοση με πρότυπο ISO
- Από το 2011 βγαίνει μία καινούργια έκδοση κάθε 3 χρόνια συνήθως με αλλαγές που συνήθως είναι επεκτάσεις της γλώσσας. Εκδόσεις:
  - **C++11** (2011)
  - **C++14** (2014)
  - **C++17** (2017)
  - επόμενη έκδοση **C++20** (2020)

Bjarne Stroustrup

Σημείωση:

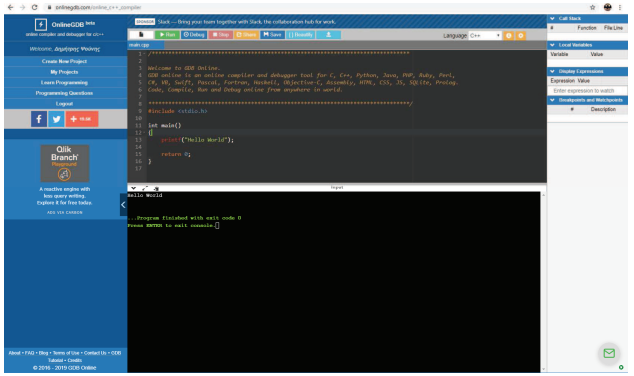
- Το όνομα C++ σημαίνει
  - **Αύξηση της C κατά 1** (με βάση τον τελεστή ++, που σημαίνει αύξηση κατά 1)
- Άρα είναι μία επέκταση της C, ωστόσο:
  - **Δεν είναι αυστηρό υπερσύνολο της**, υπάρχουν κάποια (λίγα) χαρακτηριστικά της C που δεν δουλεύουν σε C++.

Α. Θεωρία
1. Η Γλώσσα C++
3. Η αναγκαιότητα της C

- **Η C++ «πατάει» πάνω στην γλώσσα C**
  - Γι'αυτό είναι απαραίτητη καλή γνώση της C
  - Ωστόσο δεν είναι απαραίτητη η γνώση όλων των χαρακτηριστικών της C
    - Για παράδειγμα οι περισσότερες βιβλιοθήκες συναρτήσεων αλλάζουν στην C++ και γίνονται (συνήθως) πιο εύκολες στην χρήση.
  - **Τα περισσότερα βασικά χαρακτηριστικά όμως είναι ίδια.**
    - Π.χ. ο χειρισμός
      - Μεταβλητών, Τύπων Δεδομένων
      - Δομής Ελέγχου, Δομών Επανάληψης
      - Συναρτήσεων
      - Πινάκων
      - Δεικτών
    - είναι ίδιος στην C++ και είναι προαπαιτούμενος για την κατανόηση των νέων χαρακτηριστικών (κλάσεις, κληρονομικότητα κ.λπ.)
- Αλλά η μεγάλη διαφορά της C με την C++ είναι η μετατόπιση του ενδιαφέροντος στην δόμηση των προγραμμάτων από:
  - τις συναρτήσεις της C (**διαδικαστικός προγραμματισμός**)
  - στα αντικείμενα της C++ (**αντικειμενοστραφής προγραμματισμός** – object oriented programming)
- Στην συνέχεια του μαθήματος θα κάνουμε μία υπενθύμιση των βασικών χαρακτηριστικών της γλώσσας C (με κάποια νέα στοιχεία).

Α. Θεωρία
1. Η Γλώσσα C++
4. Μεταγλωττιστές

- Υπάρχουν δεκάδες μεταγλωττιστές της γλώσσας C++, είτε δωρεάν, είτε με πληρωμή.
- Στα πλαίσια αυτής της σειράς μαθημάτων θα χρησιμοποιήσουμε τον (δωρεάν) online μεταγλωττιστή **OnlineGDB**
  - Στην σελίδα https://www.onlinegdb.com/
  - Χρησιμοποιεί τον μεταγλωττιστή g++
  - Έχει μηχανισμό για debugging



Α. Θεωρία
2. Hello World!
1. Πηγαίος κώδικας

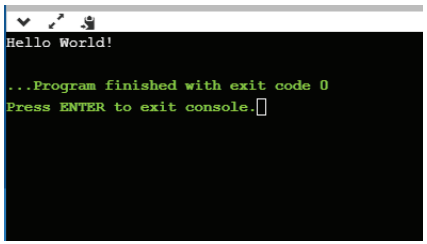
- Δημιουργούμε το νέο project CPP1.helloworld.cpp

```
/* CPP1.helloworld.cpp */

#include <iostream>

int main()
{
  std::cout<<"Hello World!";
  return 0;
}
```

- και το εκτελούμε:



Α. Θεωρία
2. Hello World!
2. Σχόλια

- Βλέπουμε την πρώτη γραμμή

```
/* CPP1.helloworld.cpp */
```

  - Είναι ένα σχόλιο. Συνηθίζεται να γράφουμε στην αρχή, το όνομα του αρχείου με κάποια επεξηγηματικά σχόλια.
- Τα σχόλια γράφονται όπως στην C και υπάρχουν δύο είδη σχολίων:
  - Σχόλια πολλών γραμμών. Ξεκινάνε με /* και τελειώνουν με */

```
/* Αυτό
είναι ένα σχόλιο πολλών
γραμμών */
```

  - Σχόλια μίας γραμμής. Ξεκινάνε με //

```
// Αυτό είναι ένα σχόλιο μίας γραμμής
```

Α. Θεωρία
2. Hello World!
3. Βιβλιοθήκη iostream

- Βλέπουμε την γραμμή

```
#include <iostream>
```

- Είναι μια οδηγία για τον προεπεξεργαστή (ξεκινά με το #)
  - Λέει πρακτικά, βρες το αρχείο (iostream.h) και βάλε τα περιεχόμενά του εδώ.
- Η βιβλιοθήκη iostream.h
  - Περιέχει τις standard κλάσεις/συναρτήσεις για τον χειρισμό εισόδου/εξόδου

Παρατηρήσεις:
- Η βιβλιοθήκη iostream.h αντικαθιστά την stdio.h της C
- Η βιβλιοθήκη είναι απαραίτητη για το αντικείμενο cout, το οποίο κάνει το χειρισμό της εξόδου στην οθόνη, εδώ της συμβολοσειράς «Hello World!»

Α. Θεωρία
2. Hello World!
4. main, block κώδικα, return

- **Η συνάρτηση main είναι το σημείο εισόδου του προγράμματος μας.**
  - Με το που ξεκινάει να τρέχει, εκτελούνται σειριακά οι εντολές μέσα στην main.

```
int main()
{
   …
   return 0;
}
```

- **Η main υποχρεωτικά επιστρεφει ακέραια τιμή.**
  - Η επιστροφή γίνεται στο λειτουργικό σύστημα που τρέχει το εκτελέσιμο.
  - Η τιμή 0 σημαίνει ότι όλα πήγαν καλά (κατά σύμβαση)
    - Ο προγραμματιστής μπορεί να διαχειριστεί μη μηδενικές τιμές για να δείξει (στο λειτουργικό) ότι κάτι δεν πήγε καλά στην εκτέλεση του προγράμματος.
- Τα άγκιστρα πάντα ορίζουν την αρχή και το τέλος της συνάρτησης main
  - και λέμε ότι ορίζεται από αυτά, ένα **μπλοκ κώδικα**.

Παρατηρήσεις:
- Η χρήση της return είναι ίδια με την C
  - Είναι μία εντολή, άρα πρέπει να ακολουθείται από ερωτηματικό.

Α. Θεωρία
2. Hello World!
5. Είσοδος / Έξοδος (1. Έξοδος με την cout)

- **Το αντικείμενο std::cout χρησιμοποιείται για την έξοδο δεδομένων στην οθόνη**

```
std::cout<<“Hello World!”;
```

  - Η πλήρης κατανόηση του, απαιτεί την γνώση κλάσεων, άλλά θα δώσουμε μια πρόχειρη ιδέα (αν δεν την κατανοούμε πλήρως, δεν πειράζει για την ώρα).
- Η **βιβλιοθήκη** iostream ορίζει μεταξύ άλλων:
  - Ένα αντικείμενο που λέγεται cout το οποίο τυπώνει στην οθόνη
  - Οι κλάσεις/αντικείμενα εισόδου/εξόδου της iostream ορίζονται σε έναν **χώρο ονομάτων** (namespace) που ονομάζεται std
    - Ο χώρος ονομάτων χρησιμοποιείται για να μην έχουν τα αντικείμενα αυτά καθολική εμβέλεια.
    - Ο χώρος ονομάτων std περιέχεται στο iostream.h
- Οπότε η έκφραση std::cout σημαίνει:
  - Βρες το αντικείμενο cout το οποίο βρίσκεται στον χώρο ονομάτων std.
- Το << θα το ονομάζουμε τελεστή εξόδου.
  - **Πρακτικά λέμε στο αντικείμενο std::cout, πάρε σαν είσοδο αυτή την συμβολοσειρά (Hello World!) και τύπωσε την στην οθόνη.**

Παρατήρηση:
- Περισσότερα για τους χώρους ονομάτων θα δούμε στη συνέχεια του μαθήματος.

Α. Θεωρία
2. Hello World!
5. Είσοδος / Έξοδος (2. Οδηγία using)

- Με την οδηγία **using**, μπορούμε να δώσουμε εντολή στο πρόγραμμα μας:
  - να ενσωματώσει στο πρόγραμμα μας ένα στοιχείο (αντικείμενο, σταθερά κ.ο.κ)
  - ώστε να μην γράφουμε κάθε φορά που βρίσκεται το αντικείμενο cout
- Έτσι ο ακόλουθος κώδικας είναι κομψότερος:

```
/* CPP1.helloworld2.cpp  using std::cout */

#include <iostream>
using std::cout;

int main()
{
  cout<<"Hello World!";
  return 0;
}
```

A. Θεωρία
2. Hello World!
5. Είσοδος / Έξοδος (2. Οδηγία using)

- και ο δρόμος του τεμπέλη:
  - Δίνουμε οδηγία στο πρόγραμμα μας να ενσωματώσει όλα τα στοιχεία του χώρου ονομάτων της std.
  - Χρησιμοποιώντας την οδηγία **using namespace std**;

```
/* CPP1.helloworld3.cpp  Ενσωμάτωση όλης της std*/

#include <iostream>
using namespace std;

int main()
{
  cout<<"Hello World!";
  return 0;
}
```


A. Θεωρία
2. Hello World!
5. Είσοδος / Έξοδος (3. Περισσότερα για την cout)

- **Ο τελεστής << λέμε ότι είναι υπερφορτωμένος (overloaded)**
  - διότι μπορεί να προβάλλει στην οθόνη, διαφορετικούς τύπους δεδομένων.
- Στα προηγούμενα παραδείγματα είδαμε πως μπορεί να προβάλλει συμβολοσειρές
  - Ας δούμε πως μπορούμε να προβάλλουμε ακέραιες και πραγματικές μεταβλητές

```
/* CPP1.overloading.cpp  Εκτύπωση διαφορετικών τύπων δεδομένων */

#include <iostream>
using namespace std;

int main()
{
  int x = 5;
  double y = 5.01;

  cout<<"To x einai ";
  cout<<x;
  cout<<" kai to y einai ";
  cout<<y;

  return 0;
}
```


A. Θεωρία
2. Hello World!
5. Είσοδος / Έξοδος (4. Περισσότερα για την cout)

- **Στην std έχει οριστεί και το std::endl**
  - Το οποίο αλλάζει γραμμή στην κονσόλα.
- Μπορούμε και να ενσωματώσουμε τον χαρακτήρα \n μέσα σε μία συμβολοσειρά της C
- Επίσης, ο τελεστής << μπορεί να γραφεί και πολλές φορές στην ίδια γραμμή μίας cout.

```
/* CPP1.endl.cpp  Χρήση του endl */

#include <iostream>
using namespace std;

int main()
{
  int x = 5;
  double y = 5.01;

  cout<<"To x einai "<<x<<endl<<"kai to y einai "<<y<<"\n";

  return 0;
}
```


A. Θεωρία
2. Hello World!
5. Είσοδος / Έξοδος (5. Είσοδος με την cin)

- **Η std::cin χρησιμοποιείται για την είσοδο από το το πληκτρολόγιο**
  - Αντικαθιστά την scanf της C
  - Λειτουργεί όπως η cout
    - Με την διαφορά ότι ο τελεστής εισόδου που έχει οριστεί είναι ο **>>**

```
/* CPP1.cin.cpp  Χρήση της cin */

#include <iostream>
using namespace std;

int main()
{
  int x;

  cout<<"Dwse x: ";
  cin>>x;

  cout<<"To x einai: "<<x;
  return 0;
}
```

Παρατήρηση:
- Οι cin και cout είναι περίπλοκα αντικείμενα. Θα τα μελετήσουμε αναλυτικά σε επόμενο μάθημα

Α. Θεωρία
3. Στοιχεία της C
1. Μεταβλητές

- Ο ορισμός των μεταβλητών γίνεται όπως στην C
- Π.χ. η δήλωση:

```
int x=5;
```

  - δηλώνει μία ακέραια μεταβλητή και της αναθέτει την τιμή 5.
- Οι τύποι δεδομένων της C υπάρχουν αυτούσιοι στην C++:
  - char, short, int, long για ακέραιες τιμές
  - float, double για πραγματικές τιμές
- Και ορίζεται ένας νέος τύπος δεδομένων:

Ο τύπος δεδομένων **bool** απεικονίζει λογικές τιμές, και συγκεκριμένα δέχεται μόνο δύο τιμές:
- **true** (λογικό αληθές). Δεσμευμένη λέξη (keyword) με τιμή 1.
- **false** (λογικό ψευδές). Δεσμευμένη λέξη (keyword) με τιμή 0.

Παρατηρήσεις:
- Κάθε αριθμητική τιμή μετατρέπεται σε λογική τιμή ώς εξής
  - Το 0, γίνεται false.
  - Κάθε άλλη τιμή, γίνεται true.
- Αναλυτικά βλ. «Γλώσσα C – Μάθημα 3: Μεταβλητές και Σταθερές»

Α. Θεωρία
3. Στοιχεία της C
1. Μεταβλητές

```
/* CPP1.bool.cpp Χρήση λογικών μεταβλητών */
#include <iostream>
using namespace std;

int main(){
   bool x = true;

   // Αριθμητική τιμή μίας λογικής μεταβλητής
   cout<<x<<endl;

   // Μετατροπή αριθμού σε λογική μεταβλητή
   x = 4;
   cout<<x<<endl;

   // Μία συνθήκη που ερμηνεύεται σε μεταβλητή bool.
   x = !(0>1);
   if (x)
      cout<<x<<endl;

   return 0;
}
```


Α. Θεωρία
3. Στοιχεία της C
2. Σταθερές

- Ισχύει ο διαχωρισμός των σταθερών σε:
  - Αριθμητικές σταθερές, π.χ. 1, 3.14
  - Συμβολικές σταθερές
    - Είτε με την οδηγία προεπεξεργαστή #define, π.χ.

```
#define PI 3.14
```

      - Η οποία αντικαθιστά κάθε εμφάνιση της PI με 3.14 πριν την μεταγλώττιση του προγράμματος
    - Είτε με την λέξη κλειδί const πριν από την δήλωση της μεταβλητής, π.χ.

```
const int x = 2;
```

      - Η οποία απαγορεύει στην μεταβλητή x να αλλάξει την τιμή της κατά την εκτέλεση του προγράμματος.

Παρατηρήσεις:
- Αναλυτικά βλ. «Γλώσσα C – Μάθημα 3: Μεταβλητές και Σταθερές»

Α. Θεωρία
3. Στοιχεία της C
3. Τελεστές και η δομή ελέγχου

- Οι **τελεστές** της C είναι ίδιοι στην C++:
  - Αριθμητικοί τελεστές:
    +,*,-,/,% και ++,--
  - Σχεσιακοί τελεστές:
    ==, >, >=, <, <=, !=
  - Λογικοί τελεστές:
    &&, ||, !
  - και οι συντομογραφίες:
    +=, -=, *=, /=, %=
- Ενώ και η σύνταξη της **δομής ελέγχου** (if) είναι ίδια.
- Και μπορούμε να χρησιμοποιήσουμε την switch

```
[προηγούμενες εντολές]
if (πρώτη συνθήκη)
{
  (εντολές1)
}
else if (δεύτερη συνθήκη)
{
  (εντολές2)
}
...
else if (συνθήκηN)
{
  (εντολέςN)
}
else
{
  (εντολέςN+1)
}
[επόμενες εντολές]
```

Παρατηρήσεις:
- Αναλυτικά βλ. «Γλώσσα C – Μάθημα 4: Τελεστές και η δομή ελέγχου»
- και τη switch από το «Γλώσσα C – Μάθημα 12: Έλεγχος Ροής Προγράμματος»

Α. Θεωρία
3. Στοιχεία της C
4. Δομές Επανάληψης

- Και οι **δομές επανάληψης** είναι ίδιες με την C:

```
[προηγούμενες εντολές]

for( αρχική; συνθήκη; βήμα)
{
   (εντολές)
}

[επόμενες εντολές]
```

```
while (Συνθήκη)
{
  (εντολές)
}
```

```
do
{
  (Εντολές)
}
while (Συνθήκη);
```

Παρατηρήσεις:
- Αναλυτικά βλ. «Γλώσσα C – Μάθημα 5: Δομές Επανάληψης»
- και τις break, continue από το «Γλώσσα C – Μάθημα 12: Έλεγχος Ροής Προγράμματος»

Α. Θεωρία
3. Στοιχεία της C
5. Συναρτήσεις

- Και οι **συναρτήσεις** είναι ίδιες με την C:

```
.....
int func(int x, int y);  <- Αυτό είναι το πρωτότυπο της συνάρτησης
                            Σημείωση: Μπορεί να γραφεί και ως int func(int, int); σε C++
main()
{
  ....
  c=func(a,b);  <- Εδώ καλούμε την συνάρτηση στην main,
  ....                          σαν μία ακόμη εντολή του προγράμματος
}

int func(int x, int y) <-Αυτό είναι το σώμα της συνάρτησης
{
  return (x*y);
}
```

Παρατηρήσεις:
- Αναλυτικά βλ. «Γλώσσα C – Μάθημα 6: Συναρτήσεις»

Α. Θεωρία
3. Στοιχεία της C
5.1. Πολυμορφισμός Συναρτήσεων

- Μια σημαντική προσθήκη στην C++ είναι ο **«πολυμορφισμός» (polymorphism)**
  - ή πως το ίδιο όνομα μπορεί να κάνει διαφορετικά πράγματα.
- Ο πολυμορφισμός συναρτήσεων στην C++ είναι μία διευκόλυνση:
  - **Το ίδιο όνομα συνάρτησης μπορεί να χρησιμοποιηθεί για να κάνει διαφορετικές δουλειές.**
- Π.χ. ενώ στην C αν θέλαμε μία συνάρτηση που να προσθέτει είτε int είτε float θα γράφαμε δύο συναρτήσεις:

```
int sumi(int x, int y);
float sumf(float x, floaty);
```

- στην C++ μπορούμε να χρησιμοποιήσουμε το ίδιο όνομα συνάρτησης:

```
int sum(int x, int y);
float sum(float x, floaty);
```

  - Ο μεταγλωττιστής αποφασίζει ποια από τις δύο συναρτήσεις θα χρησιμοποιήσει, ανάλογα με τα ορίσματα της κλήσης της συνάρτησης.

Α. Θεωρία
3. Στοιχεία της C
5.1. Πολυμορφισμός Συναρτήσεων (Παράδειγμα)

```
/* CPP1.overloading.cpp  Υπερφόρτωση
Συναρτήσεων */

#include <iostream>
using namespace std;

int sum(int x, int y);
double sum(double x, double y);

int main()
{
   std::cout<<sum(5,3);
   std::cout<<" ";
   std::cout<<sum(1.1,2.2);
   return 0;
}
```

```
int sum(int x, int y)
{
   return x+y;
}
double sum(double x, double y)
{
    return x+y;
}
```

Α. Θεωρία
3. Στοιχεία της C
5.1. Πολυμορφισμός Συναρτήσεων

- Σημαντικό!
  - Ο μεταγλωττιστής αποφασίζει ποια δήλωση να ακολουθήσει ως εξής:
    - Βλέπει την κλήση της συνάρτησης και τα ορίσματα που έχουμε βάλει σε αυτήν
    - Και καλέι εκείνη την δήλωση που ταιριάζουν τα ορίσματα.
  - Δεν λαμβάνει υπόψιν του λοιπόν, την επιστρεφόμενη τιμή.
    - Έτσι αν είχαμε τις δηλώσεις συναρτήσεων:

```
int func(int x, int y);
float func(int x, int y);
```

    - Αυτές έχουν ίδιο όνομα και ίδια ορίσματα, οπότε δεν διαφοροποιούνται και θα οδηγήσει σε λάθος μεταγλώττισης.

Παρατηρήσεις:
- Υπάρχουν και άλλα είδη πολυμορφισμού, π.χ. κλάσεων, που θα δούμε στα επόμενα μαθήματα.

Α. Θεωρία
3. Στοιχεία της C
6. Πίνακες

- Και οι **πίνακες** είναι ίδιοι με την C:
  - Παράδειγμα ορισμού πίνακα:

```
int arr[5] = {0,4,9,2,1};
```

  - Επεξεργασία κάποιου στοιχείου του πίνακα:

```
arr[2] = 4;
```

  - Παράδειγμα ορισμού διδιάστατου πίνακα

```
int arr[2][3] = {{0,1,2}, {3,4,5}};
```

  - και επεξεργασία κάποιου στοιχείου του πίνακα:

```
arr[0][1] = 4;
```

Παρατηρήσεις:
- Αναλυτικά βλ. «Γλώσσα C – Μάθημα 7: Πίνακες»

Α. Θεωρία
3. Στοιχεία της C
7. Συμβολοσειρές

- Επίσης οι συμβολοσειρές είναι ίδιες με την C
  - Παράδειγμα ορισμού συμβολοσείράς:

```
char str[80] = “A string”;
```

  - Η εκτύπωση μπορεί να γίνει με την cout (και η είσοδος με την cin)

```
cout<<str;
```

  - Ενδέχεται να μας φανεί χρήσιμη κάποια συνάρτηση του string.h
    - Προσοχή! Η ενσωμάτωση της αλλάζει και γίνεται ως εξής:

```
#include <cstring>
```

    - Παρόμοια μπορούμε να ενσωματώσουμε και άλλες βιβλιοθήκες της C π.χ. η cstdlib είναι η stdlib.h, η ctime είναι η time.h κ.ο.κ.
- Η C++ προσφέρει και έναν καινούργιο τρόπο διαχείρισης συμβολοσειρών μέσω της βιβλιοθήκης STL (αλλά θα είναι ένα από τα τελευταία μαθήματα)

Παρατηρήσεις:
- Αναλυτικά βλ.
  - «Γλώσσα C – Μάθημα 10: Συμβολοσειρές» και
  - «Γλώσσα C – Μάθημα 18: string.h»

Α. Θεωρία
3. Στοιχεία της C
7.1. Συμβολοσειρές (Παράδειγμα)

```
/* CPP1.strings.cpp  */

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

int main()
{
  char str1[80];
  char str2[80];

  cout<<"Dwse string: ";
  cin>>str1;
  strcpy(str2,str1);
  cout<<str2;

  return 0;
}
```

Α. Θεωρία

3. Στοιχεία της C

8. Δείκτες

- Οι δείκτες είναι εξίσου σημαντικοί στην C++ όπως και στην C!
  - Παράδειγμα ορισμού δείκτη:

```
int *p;
```

  - Βάζουμε τον δείκτη να δείχνει σε μια μεταβλητή (διευθυνσιοδότηση):

```
p = &x;
```

  - Και ισχύει πλέον:

```
p == &x;   // Διεύθυνση της x
```

  - Καθώς και

```
*p == x;   // Τιμή της x
```

- Και εξίσου σημαντικά είναι:
  - Η σχέση πινάκων με δείκτες
  - Η αριθμητική δεικτών
  - Τα ορίσματα συναρτήσεων (μέσω δείκτη και μέσω αναφοράς), η σταθερά NULL

Παρατηρήσεις:
- Αναλυτικά βλ. «Γλώσσα C – Μάθημα 8: Δείκτες»
- Προσοχή όμως ότι αλλάζει ο τρόπος δυναμικής δέσμευσης μνήμης (malloc – free)

Σημαντικό!

- Ο κώδικας του μαθήματος είναι διαθέσιμος στο:

https://github.com/psounis/CPP-programming

Β. Ασκήσεις

Άσκηση 1

Κατασκευάστε ένα νέο πρόγραμμα που να ζητάει από το χρήστη:
- Το όνομά του
- Το επώνυμο του
- Την ηλικία του (έτη)
- Και να τα τυπώνει έπειτα στην οθόνη σε μία γραμμή: «ΕΠΩΝΥΜΟ ΟΝΟΜΑ (ΗΛΙΚΙΑ)»

Β. Ασκήσεις

Άσκηση 2

- Κατασκευάστε μία συνάρτηση, με όνομα average, η οποία να υπολογίζει το μέσο όρο, είτε δύο ακεραίων είτε δύο πραγματικών αριθμών.
- Η main να δίνει την επιλογή στο χρήστη να εισάγει από το πληκτρολόγιο δύο ακέραιους ή δύο πραγματικούς και να υπολογίζει και να εκτυπώνει τον μέσο όρο τους.

Υπενθύμιση:
- Μέσος όρος των α+β είναι (α+β)/2.

Β. Ασκήσεις

Άσκηση 3

Το ακόλουθο πρόγραμμα από το «Γλώσσα C – Μάθημα 11» υλοποιεί την Σειριακή Αναζήτηση

```
/* break.c: Deixnei tin xrisi tis break */

#include <stdio.h>

#define N 5

main()
{
    int i;
    int A[N]= {5, 3, 2, 4, 8 }; // Pinakas dedomenwn
    int x=2; // Pros anazitisi stoixeio

    /* Psaxnoume gia to stoixeio x ston pinaka A */
    for (i=0; i<N; i++)
    {
        if (A[i]==x)
          break;
    }
    if (i<N)
          printf("\nVrethike to stoixeio %d sti thesi %d",x,i);
    else
          printf("\nDen Vrethike to stoixeio");
}
```

- Τροποποιήστε το πρόγραμμα ώστε:
  - Να είναι C++
  - Να χρησιμοποιεί λογική μεταβλητή που υποδεικνύει αν βρέθηκε το στοιχείο στον πίνακα.