Βασικά στοιχεία της C++

Βασική δομή ενός προγράμματος

Αρχικά στοιχεία ενός προγράμματος

- Οδηγίες include

#include<iostream>

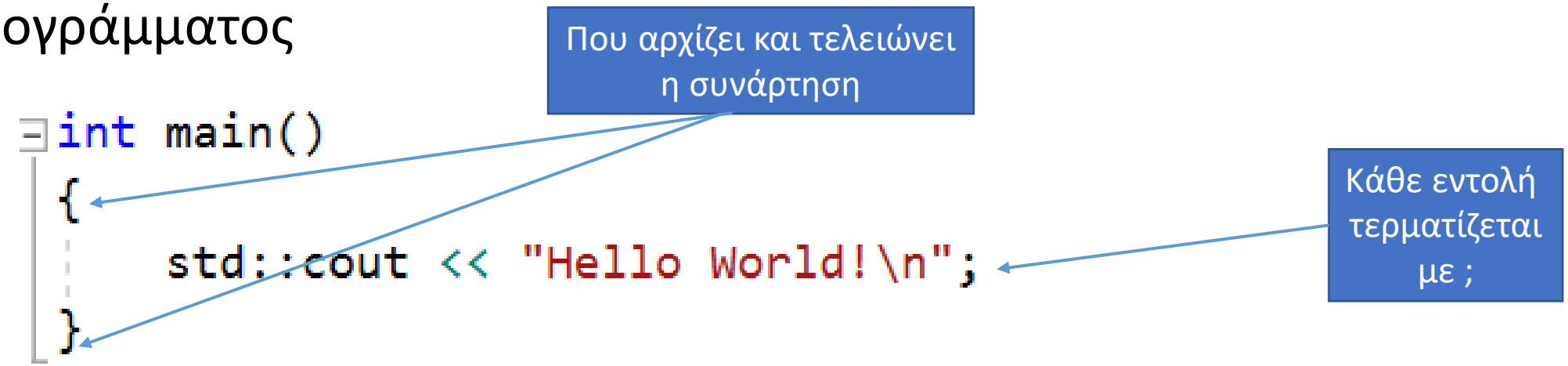
- Εισάγουν library files (αρχεία βιβλιοθήκης) στα προγράμματα. Με αυτόν το τρόπο μπορούμε να χρησιμοποιούμε αντικείμενα, συναρτήσεις, μεταβλητές και σταθερές που έχουν οριστεί σε αυτά

using namespace std;

- Αναφέρουμε τον χώρο ονομάτων που θα χρησιμοποιήσουμε (εδώ ο προκαθορισμένος της C++).
  - Με τους χώρους ονομάτων είναι δυνατή η χρησιμοποίηση των ίδιων ονομάτων πολλές φορές (π.χ. διαφορετικοί προγραμματιστές)

Συναρτήσεις

- Κάθε πρόγραμμα της C++ αποτελείται από μία ή περισσότερες συναρτήσεις (functions)
- Σε ΚΑΘΕ πρόγραμμα ΠΡΕΠΕΙ να υπάρχει μία συνάρτηση που ονομάζεται **main** και είναι αυτή από την οποία ξεκινά η εκτέλεση του προγράμματος



Διαχείριση μνήμης
Αντικείμενα - τύποι

Μνήμη

- Ένα από τα βασικά συστατικά της C και της C++ είναι ότι παρέχουν αναλυτικό «έλεγχο» της μνήμης του υπολογιστή
- Η μνήμη του υπολογιστή είναι μια «σειρά» από θέσεις στις οποίες μπορούμε να αποθηκεύσουμε/ανακτήσουμε bytes πληροφορίας

Διεύθυνση της μνήμης. Κανονικά είναι δεκαεξαδικός αριθμός και μεγάλος. Π.χ. 4FF710

100	1 0 1 1 0 1 1 1
101	1 0 1 1 0 1 0 1
102	0 0 0 1 0 1 1 1
103	1 1 0 1 0 0 0 1
104	1 0 1 0 0 1 1 1
105	0 1 0 1 0 1 1 1
106	1 1 0 1 0 1 1 0

Δεδομένα - αντικείμενα

- Τα δεδομένα τα οποία επεξεργάζεται ένα πρόγραμμα είναι αποθηκευμένα στην μνήμη
- Γενικά θεωρούμε σαν «**αντικείμενο-object**» κάποια πληροφορία που είναι αποθηκευμένη στην μνήμη
- Ένα αντικείμενο έχει
  - Μια διεύθυνση στην μνήμη
  - Έναν **τύπο**
  - Ένα συμβολικό όνομα (όχι πάντα)
  - Μία τιμή

Θα δούμε ότι ο όρος αντικείμενο έχει και άλλη ερμηνεία στην C++

Τύπος

- Ο **τύπος-type** ενός αντικειμένου είναι από τις βασικότερες έννοιες στην C++.
- **Ο τύπος καθορίζει**
  - Το «**είδος**» των δεδομένων (ακέραιοι, χαρακτήρες κτλ…)
  - Τον **χώρο** που χρειάζεται για την αποθήκευση στην μνήμη (πόσα byte)
  - Τις **πράξεις** που μπορούν να γίνουν με αυτό το αντικείμενο (πρόσθεση ή εύρεση)

Παράδειγμα τύπου αντικειμένου

- Έστω ότι έχουμε ένα αντικείμενο τύπου int που «βρίσκεται» στην θέση 100

100	1 0 1 1 0 1 1 1
101	1 0 1 1 0 1 0 1
102	0 0 0 1 0 1 1 1
103	1 1 0 1 0 0 0 1
104	1 0 1 1 0 1 1 1
105	0 1 0 1 0 1 1 1
106	1 1 0 1 0 1 1 0

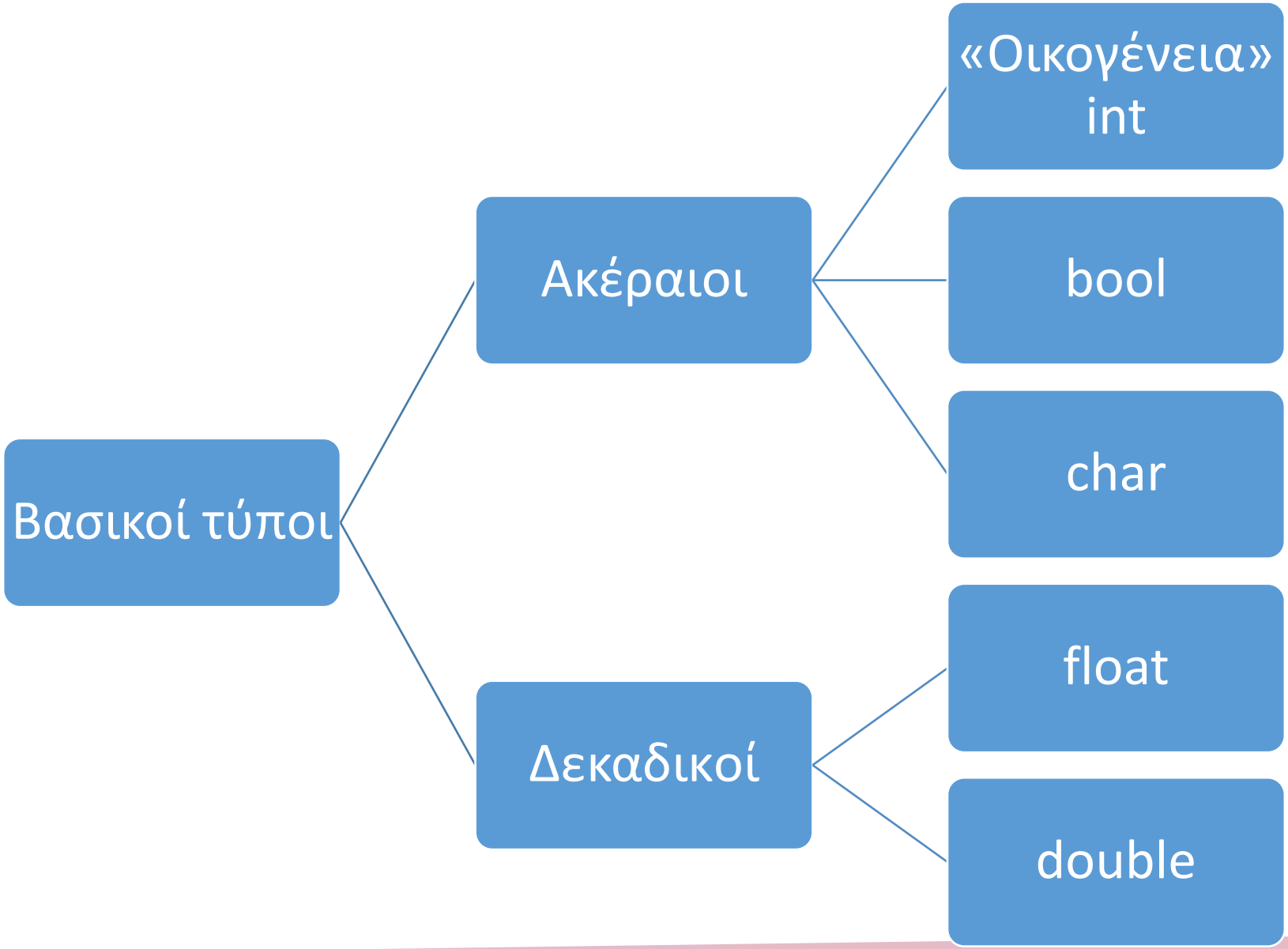
Ο τύπος του αντικειμένου καθορίζει τον «χώρο» που χρειάζεται για την αποθήκευση του.

Π.χ. Ένας ακέραιος θέλει 4 byte. Η διεύθυνση του αντικειμένου είναι η διεύθυνση του πρώτου byte.

Ο τύπος του αντικειμένου καθορίζει το «είδος» της πληροφορίας.

Π.χ. Τα παραπάνω byte τα ερμηνεύει σαν τον αριθμό -345 (παράδειγμα).

String


Βασικοί τύποι
Ακέραιοι
«Οικογένεια» int
bool
char
Δεκαδικοί
float
double

C++ και τύποι

- Η C++ είναι μια γλώσσα
  - Strongly typed
  - Statically typed

Με απλά λόγια:

Οι τύποι στην C++ θα μας απασχολήσουν πολύ!

Σταθερές

Είναι τιμές αποθηκευμένες στην μνήμη που **δεν** μεταβάλλονται.
Μπορούν να θεωρηθούν σαν «ανώνυμο» αντικείμενο

Τύπος	Παράδειγμα
Ακέραιος	3, -4, 0x3A, O34
Δεκαδικός	345.33 34.5E-2
Λογικός	true, false
Χαρακτήρα	‘a’ ‘%’ ‘5’
Συμβολοσειρά	“a string”

Variables

- Χρησιμοποιούμε ένα συμβολικό όνομα
- Κάθε variable έχει
  - ***ΟΝΟΜΑ*** το όνομα της με το οποίο το χρησιμοποιούμε στο πρόγραμμα μας
  - ***ΤΙΜΗ*** η τρέχουσα τιμή που έχει κάθε στιγμή. Μπορεί να αλλάξει
  - ***ΤΥΠΟ*** ο τύπος που είναι γνωστός at compile time
  - ***ΔΙΕΥΘΥΝΣΗ*** η περιοχή της μνήμης στην οποία είναι αποθηκευμένη εξαρτάται από τον τύπο

Symbol table

- Ο compiler διατηρεί μια «λίστα» με τις μεταβλητές του προγράμματος, τον τύπο τους και άλλα χαρακτηριστικά τους
- Με αυτό τον τρόπο μπορεί να έχει πρόσβαση στις τιμές τους

Name	Type	Location
a	int	100

Variables – ονόματα μεταβλητών

- Υπάρχουν κανόνες στα αποδεκτά ονόματα (identifiers)
  - Ξεκινάνε από χαρακτήρα
  - Μετά μπορούν να έχουν χαρακτήρες ή ψηφία
  - ΌΧΙ ειδικούς χαρακτήρες ($,%,#,^,+ , κτλ)
  - ΌΧΙ κενά (μπορεί κάτω παύλα)
  - ΌΧΙ δεσμευμένες λέξεις (if, while, for, class, κτλ)
  - Οι πεζοί με τους κεφαλαίους χαρακτήρες είναι διαφορετικοί

Δημιουργία μεταβλητών

Variables : declaration - definition

- **Declaration – δήλωση:** καθορίζεται το όνομα της μεταβλητής και το τι τύπου είναι
- **Definition – ορισμός:** αντιστοιχίζεται μια περιοχή της μνήμης, και τιμή, για την μεταβλητή αυτή
- Στην C++ όλες οι δηλώσεις είναι και ορισμοί (εκτός μια μικρή εξαίρεση)

- Μπορεί να υπάρχει και αρχικοποίηση (**initialization**)

Variable declaration

- Έχουν την μορφή

type identifier1, indentifier2 {value},… indentifiern

- Όπου ***type*** είναι κάποιος από τους τύπους, είτε βασικούς είτε αυτούς που ορίζουμε εμείς,
- ***identifier*** είναι το όνομα των μεταβλητών
- Πιθανώς να υπάρχει και μία αρχικοποίηση με τιμή ***value***

Παράδειγμα δήλωσης μεταβλητής

- Έστω ότι δηλώνουμε μια μεταβλητή τυπου int με το όνομα **data** και την αρχικοποιούμε με την τιμή 10.

1. Δεσμεύεται χώρος στην μνήμη για int και ανατίθεται η τιμή 10
2. Δημιουργείται εγγραφή για την μεταβλητή final στο symbol table

Memory

100	10
104	??????
105	???????

Symbol Table

Name	Type	Location
data	int	100

Παραδείγματα δηλώσεων

```
int x{ 0 }, y;
double p{ 3.14 };
bool a{ true };
char z{ '#' };
string name{ "chris" };
```

```
y = x;
y = y + 4;
p = 3.678;
z = 'a';
```

Ανάθεση τιμής σε μεταβλητή

- Η ανάθεση τιμής σε μεταβλητή (variable assignment) επιτρέπει την μεταβολή της τιμής που έχει μια μεταβλητή με μία καινούργια
- Πραγματοποιείται με την χρήση του τελεστή ίσον (=)

Variable assignment

- Η μορφή μιας εντολής variable assignment είναι:

lvalue = rvalue

- Αριστερά του τελεστή είναι μια **θέση στην μνήμη** (στην πιο απλή περίπτωση μια μεταβλητή)
- Δεξιά του τελεστή υπάρχει **κάτι που έχει τιμή**
  - Μια σταθερά π.χ. 21
  - Μια μεταβλητή π.χ. Name
  - Μία έκφραση (expression) που έχει μία τιμή π.χ. diameter*3.14;
- Βλέπουμε ότι το όνομα μιας μεταβλητής έχει διαφορετική σημασία αν βρίσκεται αριστερά ή δεξιά του τελεστή =

Παράδειγμα ανάθεσης τιμής σε μεταβλητή

- Έστω ότι δηλώνουμε υπάρχει ήδη μια μεταβλητή τύπου int με το όνομα data και τιμή 10.
- Αναθέτουμε σε αυτή την τιμή 25.
- Αυτό που γίνεται φαίνεται στο διπλανό σχήμα
- Ο Symbol Table δεν μεταβάλλεται

Memory πριν την ανάθεση

100	10
104	??????
105	???????

Memory μετά την ανάθεση

100	25
104	??????
105	???????

Παραδείγματα αναθέσεων

```
y = x;
y = y + 4;
p = 3.678;
z = 'a';
```

Undeclared/uninitialized variables

- Ένα κλασικό πρόβλημα είναι η χρησιμοποίηση μιας μεταβλητής η οποία δεν έχει οριστεί πότε!
- Οι μεταγλωτιστές θα εμφανίσουν μήνυμα λάθους
- Ένα μικρότερο πρόβλημα είναι η χρησιμοποίηση μιας μεταβλητής στην οποία δεν έχει ανατεθεί τιμή
  - ΔΕΝ γίνεται αρχικοποίηση σε μηδέν αυτόματα
- Οι σύγχρονοι μεταγλωτιστές φροντίζουν να εμφανίζουν μήνυμα όταν ανιχνεύουν τέτοια περίπτωση.

Μεταβλητές – scope, lifetime

- Οι μεταβλητές έχουν κάποια εμβέλεια (***scope***). Είναι το τμήμα του κώδικα στο οποίο μπορούμε να τις χρησιμοποιήσουμε
  - ***Καθολικές***: Ορίζονται έξω από συναρτήσεις και μπορούν να χρησιμοποιηθούν σε όλο τον κώδικα ενός αρχείου
  - ***Τοπικές – αυτόματες***: Ορίζονται μέσα σε ένα block και μπορούν να χρησιμοποιηθούν μόνο μέσα σε αυτό το block
- Οι μεταβλητές έχουν και κάποια διάρκεια ζωής (***lifetime***). Είναι το πόσο διατηρούν την τιμή τους
  - Οι καθολικές διατηρούν μέχρι το τέλος του προγράμματος
  - Οι τοπικές όσο είναι στην εμβέλεια εκτός αν ορίζονται σαν ***static***.

Παράδειγμα εμβέλειας

```
int a{ 9 };
{ // new scope
    int b;
    b = a + 3; //no problem
}

a = b - 1; // error b
```

Είσοδος και έξοδος

Είσοδος και έξοδος

- Η είσοδος και η έξοδος στην C++ αναπαρίστανται με την χρήση αντικειμένων τύπου ***stream.*** Η λέξη stream καθορίζει ότι οι χαρακτήρες δημιουργούνται ή καταναλώνονται ακολουθιακά ο ένας μετά τον άλλο.
- Υπάρχουν οι τύποι
  - **istream** που περιέχουν αντικείμενα που εισάγουν χαρακτήρες
  - **ostream** που περιέχουν αντικείμενα που λαμβάνουν χαρακτήρες
- Για ένα πρόγραμμα:
  - προκαθορισμένη είσοδος είναι το πληκτρολόγιο (ή αλλιώς ένα αρχείο)
  - προκαθορισμένη έξοδος είναι η οθόνη (ή αλλιώς ένα αρχείο)

iostream βιβλιοθήκη

- Η βιβλιοθήκη ***iostream*** ορίζει δύο βασικά αντικείμενα τα οποία χειρίζονται την είσοδο και την έξοδο
- ***cout*** είναι το αντικείμενο που χειρίζεται την έξοδο
- Εμφανίζονται τα στοιχεία με την χρήση του τελεστή παράθεση σε εξόδο (**<<**)

cout<<"some text"<<num<<endl

- Το ***"some text"*** είναι μία σταθερά που θα εμφανιστεί όπως είναι
- Το ***num*** είναι μία μεταβλητή όπου θα εμφανιστεί η τιμή της
- Το ***endl*** είναι ένα manipulator που ορίζει την αλλαγή γραμμής

Expressions

```
cout<<“Hello”;
```

- Η παραπάνω είναι μια εντολή expression
- Περιλαμβάνει μια πράξη ανάμεσα σε ένα αντικείμενο τύπου ostream (cout) και ένα αντικείμενο τύπου σταθεράς string (“Hello).
- Η πράξη περιγράφεται από τον τελεστή εξόδου (<<)
- Η πράξη δημιουργεί και αποτέλεσμα (θα το δούμε στην συνέχεια).

Escape sequences

- Μια σταθερά τύπου string μπορεί να περιέχει και κάποια escape sequences που δεν τυπώνονται όπως είναι αλλά εμφανίζουν κάτι ειδικό. Η μπορεί να είναι χαρακτήρες που έχουν μια ειδική σημασία στην C++ π.χ. “
- Αυτά είναι:

  \n ο χαρακτήρας αλλαγής γραμμής
  \t ο χαρακτήρας tab
  **** ο χαρακτήρας \
  \” ο χαρακτήρας “

Είσοδος – το αντικείμενο cin

- Για τον χειρισμό της εισόδου έχει οριστεί στην βιβλιοθήκη iostream το αντικείμενο **cin** και είναι τύπου **istream**
- Η είσοδος πραγματοποιείται με την χρήση του τελεστή ανάγνωση από είσοδο (>>)

cin>>x>>y;

- Τα δεδομένα πρέπει να χωρίζονται με κάποιο κενό διάστημα (κενό, αλλαγή γραμμής, tab)
- Αν δοθεί είσοδος διαφορετικού τύπου δημιουργείται πρόβλημα
- Καλό είναι να προηγείται μία έξοδο στην οθόνη που να εξηγεί ποια είναι τα δεδομένα που ζητούνται από τον χρήστη

Σχόλια

- Μπορούμε να ενσωματώσουμε στον κώδικα πληροφορίες που αγνοεί ο compiler και είναι χρήσιμες μόνο στους ανθρώπους που τον διαβάζουν
- Τα σχόλια μπαίνουν με:

  // μέχρι το τέλος της γραμμής

  /* ότι περιέχεται ανάμεσα σε αυτούς

  Τους χαρακτήρες */

Βασικοί τύποι δεδομένων

Τύπος

- Κάθε τιμή που υπάρχει στην C++ έχει έναν τύπο
- Ο τύπος καθορίζει:
  - Το πώς αποθηκεύεται αυτή η τιμή στην μνήμη
  - Το ποιες πράξεις μπορούν να γίνουν σε αυτή την τιμή και τι νόημα έχουν αυτές
- Για παράδειγμα
- 10110011 στην μνήμη τι συμβολίζει;
- επίσης πόσο είναι το x+y;

Βασικοί τύποι δεδομένων

- Η C++ παρέχει μερικούς βασικούς τύπους δεδομένων
- Αλλά μας δίνει την δυνατότητα να δημιουργήσουμε και τους δικούς μας τύπους δεδομένων (classes)

Αριθμητικοί τύποι

- Οι αριθμητικοί τύποι στην C++ χωρίζονται στις παρακάτω κατηγορίες
  - **Ακέραιοι τύποι**
  - **Δεκαδικοί τύποι**

- Επιπρόσθετα σαν αριθμητικοί τύποι αποθηκεύονται και οι
  - **Τύποι χαρακτήρων**
  - **Λογικοί**

Integers - ακέραιοι

- Ο βασικός τύπος που αποθηκεύει ακέραιες τιμές είναι ο ***int***
- Υπάρχουν και οι παραλλαγές του
  - ***short int, long int, long long int, unsigned int***
  - Αυτό που αλλάζει είναι ο αριθμός των byte που χρησιμοποιείται για την αποθήκευση καθώς και το εύρος των ακεραίων που μπορεί να αποθηκεύσει
- Η αναπαράσταση των τιμών γίνεται με ακρίβεια
- Μία σταθερά π.χ. 32 έχει τύπο int εκτός αν δηλωθεί διαφορετικά με κάποιο πρόθεμα / επίθεμα 32u

Τελεστές σε ακεραίους

- Υπάρχουν οι κλασικοί αριθμητικοί τελεστές +,-,*,/
  - ΠΡΟΣΟΧΗ Η διαίρεση είναι ακέραια
- Επίσης υπάρχει και ο πολύ σημαντικός τελεστής υπόλοιπο της ακέραια διαίρεσης (modulo) %
  - Για παράδειγμα άρτιος είναι αυτός που x%2 είναι ίσο με μηδέν

- ΠΡΟΣΟΧΗ στην προτεραιότητα των τελεστών
  - Ποιο είναι το αποτέλεσμα του 5+3*2
- Τελεστές ++, --, +=

Δεκαδικοί – floating point, double

Υπάρχουν **δύο βασικοί τύποι** για τον χειρισμό δεκαδικών τιμών

- ***float*** – μικρότερη ακρίβεια στα δεκαδικά ψηφία (4 byte)
- ***double*** – μεγαλύτερη ακρίβεια στα δεκαδικά ψηφία (8 byte)
- Υπάρχουν και οι παραλλαγές (π.χ. ***long double***)

Μία σταθερά π.χ. 3.4 έχει σαν προκαθορισμένο τον τύπο double

Τόσο οι float όσο και οι double δεν έχουν απόλυτη ακρίβεια στην αναπαράσταση των δεκαδικών τιμών

Τελεστές σε δεκαδικούς

- Ισχύουν οι ίδιοι τελεστές που ισχύουν και για τους ακεραίους.
- Αν θέλουμε να εκτελέσουμε εξειδικευμένες πράξεις μπορούμε να χρησιμοποιήσουμε μαθηματικές συναρτήσεις που είναι ορισμένες μέσα στην βιβλιοθήκη cmath (π.χ. sin, cos, power, sqrt)
- Η εμφάνιση των δεκαδικών τιμών μπορεί να γίνει με δύο τρόπους
  - Κινητή υποδιαστολή π.χ. 14.345
  - Επιστημονική μορφή π.χ. 1.4345e1

Μορφοποίηση εμφάνισης δεκαδικών

Μπορούμε να ρυθμίσουμε την μορφή εμφάνισης των δεκαδικών τιμών μέσου του αντικειμένου cout χρησιμοποιώντας τους παρακάτω χειριστές (***#include<iomanip>***)

Manipulator	Σημασία	Ισχύει για
fixed / scientific	Αν θα τυπώσει με δεκαδικά ή επιστημονική μορφή (default fixed)	Όλα τα επόμενα
setw(n)	Ο ελάχιστος αριθμός χαρακτήρων που θα τυπωθεί ο αριθμός	Μόνο το επόμενο
setprecision(n)	Ο αριθμός των χαρακτήρων για τα δεκαδικά ψηφία	Όλα τα επόμενα
dec, hex, oct	Η βάση εμφάνισης των αριθμών (δεκαδικός, δεκαεξαδικός κτλ)	Όλα τα επόμενα

Παράδειγμα μορφοποίησης

```
double x{ 34.36723 };
int y{ 12 };

cout << "x="<< scientific << x<< endl;
cout << "x=" << fixed << x << endl;
cout << "x=" <<setw(5)<<setprecision(2)<< x << endl;
cout << "y=" << y << " oct=" << oct << y << " hex=" << hex << y;
```

Αυτόματες μετατροπές τύπου

- Η C++ κάνει αυτόματες μετατροπές τύπου (ΌΧΙ ΠΑΝΤΑ) όταν δίνεται μία τιμή ενός τύπου σε μία θέση που περίμενε μια τιμή ενός διαφορετικού. Για παράδειγμα
- Δεκαδικός σε ακέραιο

int x; x=5.67;

- Αυτό που κάνει είναι να «κόψει» τα δεκαδικά ψηφία
- Ακέραιο σε δεκαδικό

double y; y=5;

- Αυτό που κάνει είναι να προσθέσει το .0

Αυτόματες μετατροπές

- Μερικές περιπτώσεις όπου πραγματοποιούνται αυτόματες μετατροπές τύπων είναι οι παρακάτω:
  - Στις αναθέσεις τιμών με την χρήση του τελεστή ίσο (=)
  - Στις πράξεις
  - Στην κλήση των συναρτήσεων όπου γίνεται πέρασμα τιμών σε παραμέτρους
  - Στην επιστροφή τιμής από συνάρτηση
- ΠΡΟΣΟΧΗ Πράξεις με int -> αποτέλεσμα int

- Αν θέλουμε εμείς να κάνουμε μετατροπή μπορούμε να χρησιμοποιήσουμε την ακόλουθη τεχνική

static_cast<int> 4.56;

Τύποι χαρακτήρα (char)

- Ο τύπος για την διαχείριση τιμών τύπου χαρακτήρα είναι ο char
  - Οι καινούργιες εκδόσεις της γλώσσας έχουν ορίσει και καινούργιους τύπους για να υποστηρίζουν όλες τις γλώσσες
- Οι σταθερές τύπου χαρακτήρα γράφονται μέσα σε απλά εισαγωγικά (π.χ. ‘a’)
- Στην πράξη αυτό που γίνεται είναι ότι ο υπολογιστής μετατρέπει τους χαρακτήρες σε ακεραίους με βάση έναν πίνακα μετατροπής που ονομάζεται κώδικας ASCII

Αριθμητική τύπων χαρακτήρα

- Η εσωτερική αναπαράσταση των τύπων χαρακτήρα σαν ακεραίους τους δίνει την δυνατότητα να πραγματοποιούν τις αριθμητικές πράξεις
- Δεν έχουν όλες νόημα
- Μερικές που έχουν νόημα είναι:
- ‘6’-’0’
- ‘C’ + 22

Τύποι αλφαριθμητικών (string)

- Αν και δεν ανήκουν στους βασικούς τύπους τα αλφαριθμητικά είναι πολύ χρήσιμα στα προγράμματα μας
- Τα αλφαριθμητικά στην C++ περιλαμβάνονται μέσα σε διπλά εισαγωγικά “this is a string”
- Η C++ μπορεί να χειρίζεται τα αλφαριθμητικά με δύο τρόπους
  - με την χρήση πινάκων χαρακτήρα (όπως η C)
  - με την χρήση της κλάσης string (απαιτεί την βιβλιοθήκη string)
- Εμείς θα χρησιμοποιήσουμε την κλάση string

Χρήση των string

- Δήλωση

string myname{“chris”};

- Ανάθεση τιμής

myname=“babis”;

- Πράξεις με string

myname=myname + “ Papadopoulos”;

myname[3];

Λογικές τιμές

- Στην C++ υπάρχει ο τύπος bool που χειρίζεται τις λογικές τιμές
- Οι τιμές που μπορεί να πάρει είναι οι true και false
- Στην πράξη αυτό που κάνει είναι αναπαριστά την true με 1 και την false με μηδέν

- Επομένως έχουμε την ακόλουθη μετατροπή
  - bool σε ακέραιο (true σε 1 και false σε 0)
  - Ακέραιο σε bool (μη μηδενική σε true και μηδενική σε false)

Ο τύπος auto

- Αν έχουμε μια δήλωση μεταβλητής η οποία συνοδεύεται με αρχικοποίηση τότε μπορούμε να αναθέσουμε στον μεταφραστή να βρει αυτός τον τύπο της μεταβλητής

- Η τιμή της αρχικοποίησης πρέπει να είναι γνωστή at compile time

auto x=4.56;

Άλλοι Τελεστές

Τελεστές σύγκρισης

- Οι τελεστές σύγκρισης ελέγχουν αν κάποια τιμή είναι μεγαλύτερη/μικρότερη από κάποια άλλη

>,<,>=,<=

- Εφαρμόζονται σε όλους τους βασικούς τύπους και το αποτέλεσμα τους είναι λογικού τύπου
- Έχουν προτεραιότητα χαμηλότερη από τους αριθμητικούς τελεστές και από τους τελεστές παράθεσης στην έξοδο – ανάγνωσης από την είσοδο

Τελεστές ισότητας ανισότητας

- Υπάρχουν και οι τελεστές ισότητας (==) / ανισότητας (!=)
- ΠΡΟΣΟΧΗ ένα πολύ «κλασικό» λάθος που κάνουν όλοι οι αρχάριοι στον προγραμματισμό στην C++ είναι η χρήση του τελεστή ανάθεσης (=) στην θέση του τελεστή ισότητας (==)
- Έχουν την παρόμοια συμπεριφορά με τους τελεστές σύγκρισης
- Η προτεραιότητα τους είναι χαμηλότερη από τους τελεστές σύγκρισης

Λογικοί τελεστές

- Οι λογικοί τελεστές εφαρμόζονται πάνω σε λογικές τιμές και έχουν σαν αποτέλεσμα και πάλι λογική τιμή
  - Μπορεί να γίνει αυτόματη μετατροπή τύπου σε λογική
- Οι λογικοί τελεστές είναι οι
  - ΚΑΙ &&
  - Η ||
  - ΌΧΙ !
- Η προτεραιότητα τους είναι η χαμηλότερη (εκτός του τελεστή ανάθεσης)

Ο τελεστής sizeof

- Ο τελεστής sizeof μπορεί να πάρει σαν όρισμα
  - Έναν τύπο
  - Ένα expression
- αυτό που επιστρέφει είναι ο χώρος που καταλαμβάνει στην μνήμη το όρισμα.
- Δεν μπορεί να εφαρμοστεί σε δυναμική μνήμη.

Μετατροπείς βασικών τύπων

Μετατροπείς βασικών τύπων

- Στην δήλωση μιας μεταβλητής όταν γράφουμε το όνομα του τύπου μπορούμε να εισάγουμε και τους μετατροπείς των τύπων που ουσιαστικά κάνουν μια μικρή μεταβολή στον βασικό τύπο της μεταβλητής
- Οι μετατροπείς είναι
  - Η λέξη const που καθορίζει ότι η μεταβλητή είναι σταθερά
  - Ο χαρακτήρας * που καθορίζει ότι είναι ένας «δείκτης» στον βασικό τύπο
  - Ο χαρακτήρας & που καθορίζει ότι είναι μία αναφορά στον βασικό τύπο

Const

- Η τιμή της μεταβλητής δεν μπορεί να αλλάξει κατά την διάρκεια εκτέλεσης του προγράμματος
- **ΠΡΕΠΕΙ να γίνεται αρχικοποίηση!**
- Στην συνέχεια μπορεί να χρησιμοποιηθεί μόνο σαν rvalue

```
const int x{ 4 },y{ 3 };
int z;

z = x * y; // ok rvalues
y = z * y; // error y as lvalue
```

Pointer

- Η τιμή της μεταβλητής είναι η διεύθυνση μιας θέσης στην μνήμη όπου είναι αποθηκευμένο ένα αντικείμενο αυτού του τύπου
- Στην δήλωση χρησιμοποιείται ο χαρακτήρας *
- Μπορεί να συνδυαστεί με το const
- Περισσότερα στο κεφάλαιο 7

```
int x{ 4 };
int* a, y{ 5 };
const int* b = &y;

a = &x;

// a points to x
// b points to y
```

Reference

- Η μεταβλητή είναι ένα «ψευδώνυμο» για μια άλλη μεταβλητή
- ΠΡΕΠΕΙ ΝΑ ΑΡΧΙΚΟΠΟΙΕΙΤΑΙ
- Δεν δημιουργείται νέο αντικείμενο επομένως δεν μπορεί να συνδυαστεί με το const

```
int x;
int& a = x, y{ 5 };
// a and x is the same
// y is a normal int

x = y;
a = 4;

cout << x << endl; // prints 4
```

Διαφορά pointer / reference