Streams - strings

Εισαγωγή στα object / class

Αρχική ιδέα object

- ***Objects – αντικείμενα*** είναι κάποιες «σύνθετες» μεταβλητές που περιγράφουν ποιο «πολύπλοκα» δεδομένα από τα βασικά
  - Όπως για παράδειγμα δηλώνουμε μια μεταβλητή x τύπου int

    int x;
  - Μπορούμε να δηλώσουμε και μια μεταβλητή a τύπου π.χ. Book

    book a;
- Ο τύπος μιας μεταβλητής object λέγεται κλάση (**class**)
  - Στο παραπάνω παράδειγμα λέμε ότι το a είναι ένα object της book class

Αρχική ιδέα object

- Ένα object περιέχει πεδία (fields ή members)
  - ***δεδομένα*** (ονομάζονται ***data members*** ή ***data fields***)
  - ***functions*** (που τις ονομάζουμε ***member functions*** ή ***methods***)

- Η C++ υποστήριξε αντικείμενα από την αρχή
- Παράδειγμα αντικειμένων είναι τα strings, τα αρχεία, κ.α.
- Υπάρχουν κλάσεις που έχουν οριστεί ήδη (σε βιβλιοθήκες), μπορούμε όμως να ορίσουμε και δικά μας

Πρόσβαση στα members

- Η πρόσβαση στα members ενός object γίνεται με την χρήση του τελεστή τελεία (.)
- Για παράδειγμα αν το a είναι object της κλάσης Book και η κλάση έχει ένα data member με το όνομα size και ένα member function με το όνομα erase (που δεν παίρνει ορίσματα) η πρόσβαση σε αυτά θα γίνεται ως εξής

a.size

a.erase()

- ΠΡΟΣΟΧΗ. Η πρόσβαση γίνεται μέσω αντικειμένου ΟΧΙ κλάσης.

Χρήση των Strings

Υλοποίηση των string στην C

- Η C υλοποιούσε τα strings σαν πίνακες χαρακτήρων (χαρακτήρες αποθηκευμένους σε διαδοχικές θέσεις στην μνήμη)
- Τελικός χαρακτήρας ήταν ό \0
- Αυτή η υλοποίηση δημιουργούσε αρκετά προβλήματα
- Υποστηρίζεται ακόμα από την C++

String class

- Επιτρέπει στον χρήστη να χρησιμοποιεί τα strings όπως χρησιμοποιεί και τις τιμές των άλλων βασικών data types
- Για να μπορέσουμε να χρησιμοποιήσουμε την string πρέπει να εισάγουμε την βιβλιοθήκη

#include<string>

using namespace std;

Δημιουργώντας μεταβλητές τύπου string

- ΠΡΟΣΟΧΗ Μια σταθερά π.χ. “a single string” ΔΕΝ είναι τύπου string αλλά τύπου πίνακα χαρακτήρων (char *)
- Μπορούμε να δηλώσουμε μια μεταβλητή τύπου string και να την αρχικοποιήσουμε με μία σταθερά

```
string name1{ "nikos" };
string name2;
name2 = "tasos";
```

Λειτουργίες σε strings

- Μπορούμε να ενώσουμε μεταβλητές τύπου string και σταθερές με τον τελεστή +
- Μπορούμε να διαβάζουμε και να εκτυπώνουμε με τους τελεστές >>, <<

```
string name1{ "kyrios" };
string name2;
string tmp;

cout << "Dose to onoma sou: ";
cin >> name2;
tmp = name1 + " " + name2;
cout << tmp << endl;
```

Λειτουργίες σε strings

- Μπορεί να γίνει έλεγχος αν δύο strings είναι ίσα , άνισα (προσοχή case sensitive)

```
string name1{ "kalim" };
string name2{ "kalimera" };

cout << (name1 == name2) << endl;
cout << (name1 > name2) << endl;
```

- Μπορεί να γίνει έλεγχος αν ένα string είναι μεγαλύτερο από ένα άλλο

Μέγεθος string

- ***size*** – επιστρέφει το μέγεθος ενός string
  - ΔΕΝ είναι int αλλά string::size_t καλό είναι να δηλώνεται σαν auto
- ***empty*** – ελέγχει ένα string αν είναι άδειο

```
string s1;
string s2{ "dokimi" };

if (s1.empty()) {
    cout << "to 1o einai keno\n";
}
auto mikos = s2.size();
cout << "to 2o exei mikos: " << mikos <<endl;
```

Πρόσβαση σε χαρακτήρα ενός string

- Μπορεί να χρησιμοποιηθεί ο subscript operator []
- Δίνεται ένα index που είναι ακέραιος που καθορίζει την θέση του χαρακτήρα στο string (πρώτο μηδέν)
- Πρέπει να γίνεται έλεγχος αν είναι σωστό το index
- Μια εναλλακτική είναι να χρησιμοποιηθεί η μέθοδος at

```
string name{ "Country" };

cout << name[0] << endl;
cout << name[name.size() - 1] << endl;
cout << name.at(2) << endl;
```

Πρόσβαση σε όλα τα στοιχεία ενός string

- Πολλές φορές χρειάζεται να έχουμε πρόσβαση σε όλους τους χαρακτήρες ενός string

- Αυτό μπορεί να γίνει με μία ειδική μορφή της for που ονομάζεται ***range for***

```
string code{ "344 889 333" };

for (auto x : code)
    if (x == ' ')
        cout << '-';
    else
        cout << x;
```

Μεταβολή των στοιχείων ενός string

- Στο παράδειγμα της προηγούμενης στην μεταβλητή x γίνεται ανάθεση του κάθε χαρακτήρα του string “by value”
- Αυτό σημαίνει ότι δεν μπορούμε να μεταβάλουμε το string
- Για να μεταβάλλουμε το string πρέπει η ανάθεση να γίνει “by reference”

```
string code{ "344 889 333" };

for (auto &x : code)
    if (x == ' ')
        x= '-';

cout << code;
```

Είσοδος με την getline

- Αν προσπαθήσουμε να διαβάσουμε ένα string με την χρήση του τελεστή >> θα διαπιστώσουμε ότι «σταματάει» σε κάθε κενό χαρακτήρα
- Λύση βρίσκεται με την χρήση της συνάρτησης getline
- Μπορεί να πάρει και 3ο όρισμα τον χαρακτήρα στον οποίο θα σταματήσει την ανάγνωση από την είσοδο

```
string input;

cout << "with >> operator:\n";
cin >> input;
cout << input << endl;

cout << "with getline:\n";
getline(cin, input);
cout << input << endl;
```

Μέθοδοι της string

- ***str.insert(pos,str2)***
  - Εισάγει string str2 στην θέση pos
- ***str.erase (pos,length)***
  - Αφαιρεί length χαρακτήρες από την θέση pos
- ***str.find(str1)***
  - Επιστρέφει την θέση που βρίσκεται το str1
- ***str.find(str1,pos)***
  - Επιστρέφει την θέση που βρίσκεται το str1 ξεκινά αναζήτηση από pos

```
string input{ "another day in sun" };
string input2{ "another day in sun" };
string str{ "day" };

cout << input.insert(15, "the ") << endl;
cout << input.erase(7,4)<<endl;
cout << input2.find(str) << endl;
cout << input2.find("a", 5)<<endl;
```

Χρήστη των streams για είσοδο έξοδο

Input / output

- Η είσοδος και η έξοδος ενός προγράμματος συχνά παρουσιάζεται στην βιβλιογραφία σαν ***I/O (input/output)*** και περιλαμβάνει
  - Σαν είσοδο μία συσκευή που δίνει δεδομένα στο πρόγραμμα μας (π.χ. πληκτρολόγιο) ή ένα αρχείο από το οποίο διαβάζουμε δεδομένα
  - Σαν έξοδο μία συσκευή που εμφανίζει τις πληροφορίες από το πρόγραμμα μας (π.χ. οθόνη) ή ένα αρχείο στο οποίο αποθηκεύουμε τις πληροφορίες

- Τόσο η είσοδος όσο και η έξοδος στην C++ περιγράφονται με αντικείμενα ***stream***
  - Ουσιαστικά περιγράφουν ροή δεδομένων
  - Μπορεί να είναι input και output

Cin και cout objects

- Η βιβλιοθήκη iostream που χρησιμοποιείται από ένα πρόγραμμα με την οδηγία #include<iostream> περιγράφει δύο βασικά objects τύπου stream
  - **cin** που συνδέεται με το πληκτρολόγιο
  - **cout** που συνδέεται με την οθόνη
  - Υπάρχουν και τα ***cerr*** και ***clog***

- Μπορούμε να ορίσουμε και δικά μας object τύπου stream που να γράφουν/διαβάζουν από ένα αρχείο

Είσοδος και έξοδος σε αρχείο

- Η είσοδος και έξοδος σε αρχείο μοιάζει σε μεγάλο βαθμό με την είσοδο και έξοδο σε πληκτρολόγιο / οθόνη

- Πρέπει να δημιουργήσουμε τα αντίστοιχα αντικείμενα
- Γράφουμε / διαβάζουμε από την αρχή μέχρι το τέλος
- Δεν έχουμε την δυνατότητα να πάμε πίσω (αλλά μπορούμε να αρχίσουμε από την αρχή)

Η χρήση stream objects

Δημιουργία stream objects

Όπως και οι μεταβλητές του απλού τύπου:

- Πρέπει να δηλωθούν
- Πρέπει να αρχικοποιηθούν πριν από την χρήση τους (στην ουσία δηλαδή να γίνει η σύνδεση του object με ένα αρχείο στο σύστημα αρχείων)
- Από εκεί και πέρα μπορεί να γίνει ανάθεση νέας τιμής στο object (δηλαδή να δείχνει πλέον σε καινούργιο αρχείο)

Ορίζοντας ένα stream object

- Για να χρησιμοποιήσω τα stream objects στο πρόγραμμα μου πρέπει να κάνω τα παρακάτω

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

- Από εκεί και πέρα μπορώ να ορίσω ένα stream
  - Εισόδου

```
ifstream in_stream;
```

  - Εξόδου

```
ofstream out_stream;
```

  - Εισόδου & εξόδου

```
fstream in_out_stream;
```

Σύνδεση με ένα αρχείο

- Αφού γίνει η δήλωση ενός stream object πρέπει να γίνει και η σύνδεση του με ένα αρχείο. Αυτή γίνεται με την χρήση μίας μεθόδου με το όνομα open

in_stream.open("input.dat");

- Το ίδιο ισχύει και για τα stream εξόδου. Το όνομα του αρχείου μπορεί να είναι το full path αν το αρχείο δεν είναι στον ίδιο φάκελο με τον κώδικα της C++

Χρήση ενός stream object

- Από την στιγμή που ένα stream object εισόδου / εξόδου συνδεθεί με ένα αρχείο μπορεί να χρησιμοποιηθεί όπως και το cin, cout με την χρήση των τελεστών >> και << αντίστοιχα
- Το όνομα του αρχείου ΔΕΝ χρειάζεται πλέον
- Ανάγνωση μιας μεταβλητής από αρχείο

in_stream>>x;

- Εγγραφή της τιμής μιας μεταβλητής σε αρχείο

out_stream<<x;

Κλείσιμο stream object

- Από την στιγμή που θα τελειώσει η χρήση του αρχείου ανάγνωση/εγγραφή το αρχείο πρέπει να «κλείσει»
- Με αυτόν τον τρόπο δηλώνουμε στο λειτουργικό σύστημα ότι πλέον δεν χρησιμοποιούμε το συγκεκριμένο αρχείο και μπορούν να έχουν πρόσβαση τα άλλα προγράμματα
- Το «κλείσιμο» του αρχείου γίνεται με την μέθοδο close

in_stream.close();

Αποτυχία ανοίγματος file

- Μπορεί το open να αποτύχει:
  - Δεν υπάρχει / δεν έχουμε δικαιώματα στο αρχείο
  - Λάθος το όνομα ενός αρχείου
- Μπορούμε να χρησιμοποιήσουμε το stream σε condition όπου αν έχει αποτύχει η open θα επιστρέψει false
- Στην συνέχεια γίνεται κλίση της συνάρτησης ***exit*** που τερματίζει την εκτέλεση του προγράμματος

```cpp
int main() {
    ifstream in_stream;
    ofstream out_stream;
    int x, y;

    in_stream.open("data.txt");
    out_stream.open("results.txt");
    if (!in_stream || !out_stream) {
        cout << "Lathos sto anoigma tvn arxeivn\n";
        exit(1);
    }

    in_stream >> x >> y;
    cout << "Grafo stin othoni\n";
    cout << "Diabasa x=" << x << " y=" << y << endl;

    out_stream << "Grafo sto arxeio\n";
    out_stream << "Diabasa x=" << x << " y=" << y << endl;

    in_stream.close();
    out_stream.close();
}
```

Ιδιαιτερότητες ενός stream object

- Δεν μπορεί να γίνει αρχικοποίηση με άλλους τρόπους εκτός από την open
- Δεν μπορεί να γίνει ανάθεση
- Σαν αποτέλεσμα δεν μπορεί να γίνει πέρασμα σαν όρισμα by value ή επιστροφή από συνάρτηση by value.
- ΜΠΟΡΕΙ να γίνει ανάθεση by reference

```
int func(ifstream in);

int main() {
    ifstream in1;

    in1.open("data.txt"); // ok
    ifstream in2{ in1 };  // error

    ifstream in3;
    in3 = in2; // error no assignment by value

    ifstream& in4 = in1; // ok by reference

    int x = func(in1);
}
```

```cpp
int main() {
    ifstream infile;
    int sum;

    infile.open("mydata.txt");
    sum = countblanks(infile);

    cout << "to arxeio exei " << sum << " kena\n";
    return 0;
}
```

```cpp
int countblanks(ifstream &in)
{
    char c;
    int count = 0;

    c = in.get();
    while (!in.eof())
    {
        if (c == ' ')
            ++count;
        c = in.get();
    }
    return count;
}
```

Έξοδος σε αρχείο -Μορφοποίηση

Χειριστές - Manipulators

Είναι στοιχεία που εισάγονται στο stream με τον τελεστή <<. Πρέπει να γίνει χρήση του #include<iomanip>

- **setw(n)** ελάχιστο πλήθος των χαρακτήρων για την εμφάνιση του επόμενου στοιχείου
- **setprecision(n)** πλήθος όλων των ψηφίων στην εμφάνιση όλων των επόμενων στοιχείων
- Manipulators χωρίς arguments
  - **showpoint** εμφανίζει το δεκαδικό ψηφίο
  - **showpos** εμφανίζει το πρόσημο
  - **showbase** εμφανίζει την βάση του αριθμού

Χειριστές - Manipulators

- Μορφή αναπαράστασης των δεκαδικών (όλα)
  - ***fixed, scientific***
- Στοίχιση (επόμενο)
  - ***left, right***
- Αριθμητική βάση (όλα)
  - ***dec,hex,oct***
- Καθορισμός χαρακτήρα «γεμίσματος» κενών (όλα)
  - ***setfill(char)***

Παράδειγμα manipulator

```
int main() {
    int inta = 21;
    double doub = 356.3774322;

    cout << scientific << doub << endl;
    cout << fixed << setprecision(4) << doub<< endl;
    cout << oct << inta << endl;
    cout << setw(10) << inta << endl;
    return 0;
}
```

Σημαίες - flags

- Εφαρμόζονται στην κλάση ios και καθορίζουν την μορφοποίηση
- Υπάρχουν αυτές που ανήκουν σε ομάδες και αυτές που είναι αυτόνομες
- Μερικές παρουσιάζονται στον διπλανό πίνακα
- Στο μάθημα θα χρησιμοποιήσουμε τους manipulators

Flag	Mask
left right	adjustfield
dec, oct, hex	basefield
scientific, fixed	floatfield
showbase, showpoint, showpos	

Ανάγνωση από είσοδο

1. Ανάγνωση τιμή ανά τιμή

- Τα αρχεία που περιέχουν δεδομένα εισόδου μπορούν να έχουν οποιοδήποτε μέγεθος. Τα προγράμματα μπορεί να μην γνωρίζουν εκ των προτέρων το μέγεθος τους
- Ένας τρόπος αναγνώρισης του τέλους παρέχει ο τελεστής ανάγνωσης >>
- Η έκφραση

in_stream>>x;

- Επιστρέφει true αν μπόρεσε να διαβάσει και false αν δεν μπόρεσε

Παράδειγμα ανάγνωσης ανά τιμή

```
int main() {
    ifstream in_stream;
    int x;

    in_stream.open("intdata.txt");
    // check if open

    while (in_stream >> x) {
        cout << setw(10)<<showpos << x<<endl;
    }

    in_stream.close();
}
```

2. Ανάγνωση ανά χαρακτήρα

- Η χρήση του τελεστή ανάγνωσης (>>) έχει κάποιους περιορισμούς (π.χ. αγνοεί τα κενά)
- Αν θέλουμε να διαβάζουμε όλους τους χαρακτήρες ενός αρχείου μπορούμε να κάνουμε ανάγνωση ανά χαρακτήρα με την χρήση της μεθόδου get ενός input stream object

in_stream.get(ch);

- Με αυτόν τον τρόπο μπορούμε να διαβάσουμε όλους τους χαρακτήρες ακόμα και τα κενά

Ανίχνευση τέλους αρχείου

- Κάθε φορά που διαβάζουμε από ένα αρχείο μετακινείται ένας δείκτης που «δείχνει» ποιο είναι το επόμενο στοιχείο που θα διαβάσει
- Όταν φτάσει στο τέλος του αρχείου δεν μπορεί να διαβάσει άλλο πλέον
- Μπορούμε να ελέγξουμε αν ο δείκτης αυτός έχει φτάσει στο τέλος του αρχείου με την μέθοδο **eof.**

while(!in_stream.eof())

Παράδειγμα ανάγνωσης χαρακτήρα

```
int main() {
    ifstream in_stream;
    char c;

    in_stream.open("chardata.txt");

    while (!in_stream.eof()) {
        in_stream.get(c);
        if (c == '-')
            cout << ' ';
        else
            cout << c;
    }

}
```

Εκτύπωση ανά χαρακτήρα

- Πολλές φορές όταν διαβάζουμε ένα αρχείο ανα χαρακτήρα θέλουμε να εκτυπώσουμε τους χαρακτήρες σε αρχείο
- Αυτό μπορεί να γίνει με την μέθοδο put ενός ofstream object

out_stream.put(ch);

3. Ανάγνωση ανά γραμμή

- Σε περίπτωση που θέλουμε να διαβάσουμε ολόκληρη γραμμή χρησιμοποιούμε την getline
- Επιστρέφει false όταν φτάσει στο τέλος του αρχείου

```
int main() {
    ifstream in_stream;
    string line;
    int count{ 0 };

    in_stream.open("text.txt");
    while (getline(in_stream, line)) {
        ++count;
        cout << "Line " << count << ":" << line<<endl;
    }

}
```

4. Ανάγνωση ανά γραμμή / επεξεργασία ανά τιμή

- Η κλάση sstringstream μετατρέπει ένα string σε stream
- Πολλές φορές θέλουμε να έχουμε πρόσβαση σε μία γραμμή και μετά να την επεξεργαζόμαστε διαφορετικά
- Σε αυτό μπορεί να μας βοηθήσει η istringstream

```
string line;
int num = 0, sum = 0;
while (getline(cin, line)) {
    istringstream sline(line);
    sum = 0;
    while (sline >> num)
        sum += num;
    cout << "Sum of line is " << sum << endl;
}
```