Ελληνική Δημοκρατία
Τεχνολογικό Εκπαιδευτικό
Ίδρυμα Ηπείρου

Αντικειμενοστραφής Προγραμματισμός

Ενότητα 9 : **Τα πρότυπα (templates)**

Ιωάννης Τσούλος

Τμήμα Μηχανικών Πληροφορικής Τ.Ε

Αντικειμενοστραφής Προγραμματισμός

Ενότητα 9 : Τα πρότυπα (templates)

Ιωάννης Τσούλος
Επίκουρος Καθηγητής

Άρτα, 2015

Άδειες Χρήσης

- Το παρόν εκπαιδευτικό υλικό υπόκειται σε άδειες χρήσης Creative Commons.
- Για εκπαιδευτικό υλικό, όπως εικόνες, που υπόκειται σε άλλου τύπου άδειας χρήσης, η άδεια χρήσης αναφέρεται ρητώς.

Χρηματοδότηση

- Το έργο υλοποιείται στο πλαίσιο του Επιχειρησιακού Προγράμματος «**Εκπαίδευση και Δια Βίου Μάθηση**» και συγχρηματοδοτείται από την Ευρωπαϊκή Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο) και από εθνικούς πόρους.
- Το έργο «**Ανοικτά Ακαδημαϊκά Μαθήματα στο ΤΕΙ Ηπείρου**» έχει χρηματοδοτήσει μόνο τη αναδιαμόρφωση του εκπαιδευτικού υλικού.
- Το παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στα πλαίσια του εκπαιδευτικού έργου του διδάσκοντα.

Ευρωπαϊκή Ένωση
Ευρωπαϊκό Κοινωνικό Ταμείο
ΕΠΙΧΕΙΡΗΣΙΑΚΟ ΠΡΟΓΡΑΜΜΑ
ΕΚΠΑΙΔΕΥΣΗ ΚΑΙ ΔΙΑ ΒΙΟΥ ΜΑΘΗΣΗ
επένδυση στην κοινωνία της γνώσης
ΥΠΟΥΡΓΕΙΟ ΠΑΙΔΕΙΑΣ & ΘΡΗΣΚΕΥΜΑΤΩΝ, ΠΟΛΙΤΙΣΜΟΥ & ΑΘΛΗΤΙΣΜΟΥ
ΕΙΔΙΚΗ ΥΠΗΡΕΣΙΑ ΔΙΑΧΕΙΡΙΣΗΣ
ΕΣΠΑ 2007-2013
πρόγραμμα για την ανάπτυξη
ΕΥΡΩΠΑΪΚΟ ΚΟΙΝΩΝΙΚΟ ΤΑΜΕΙΟ
Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης

Εισαγωγικά

- Πρότυπα (templates): αποτελεί τη βάση της STL, καθώς επιτρέπουν τη χρήση πολυμορφισμού κατά τη μεταγλώττιση (compile-time) αντί κατά την εκτέλεση (runtime), γεγονός που αυξάνει κατά πολύ την απόδοσή τους, αφού ο μεταγλωττιστής μπορεί να πραγματοποιήσει αρκετές βελτιστοποιήσεις κατά την μεταγλώττιση.
- Πρότυπα μπορούν να οριστούν για συναρτήσεις ή για κλάσεις.
- Η λειτουργία είναι ή ίδια, με τη διαφορά ότι αν οριστεί μια κλάση πρότυπο, η προτυποποίηση ισχύει για κάθε μέθοδο/συνάρτηση της κλάσης.

Εισαγωγικά

- **Σκοπός προτύπων**
  - Αν σε κάποιο πρόγραμμα χρειαζόμαστε την ταξινόμηση δύο διαφορετικών συνόλων δεδομένων (π.χ. ένα πίνακα από int και ένα πίνακα από string) θα χρειαστούμε δύο συναρτήσεις που θα πραγματοποιήσουν την ταξινόμηση στους πίνακες.
  - Κάτι τέτοιο είναι περιττό, επιρρεπές σε λάθη και αιτία αύξησης του μεγέθους του προγράμματος.
  - Αν το πρόγραμμα εξελιχθεί να υποστηρίζει και άλλους τύπους δεδομένων, ακόμη και κλάσεις, θα χρειαστούμε μια συνάρτηση ταξινόμησης για κάθε τύπο! Οπωσδήποτε πρέπει να υπάρχει μια καλύτερη λύση.

Εισαγωγικά

- **Λύση**
  - Να οριστεί ένα πλαίσιο, ή πρότυπο, το οποίο θα περιγράφει τον αλγόριθμο, χρησιμοποιώντας μια αφηρημένη κλάση, ο τύπος της οποίας δε μας ενδιαφέρει παρά μόνο ορισμένα χαρακτηριστικά που είναι απαραίτητα.
  - *Π.χ.* για τη ταξινόμηση χρειαζόμαστε απλώς μια μέθοδο σύγκρισης ανάμεσα σε δύο τυχαία αντικείμενα.

    Η δήλωση ενός τέτοιου πλαισίου γίνεται με την λέξη-κλειδί `template` ακολουθούμενη από το όνομα της αφηρημένης κλάσης κλεισμένης σε <>.

Παράδειγμα κλάσης λίστας αντικειμένων


```
#include <iostream>
using namespace std;
// Ορίζουμε την κλάση list ως template που θα χρησιμοποιεί την αφηρημένη κλάση
// data_t (δεν υπάρχει πραγματικά, απλώς
// υποδηλώνει μια οποιαδήποτε κλάση).
template<class data_t> class list {
// το item θα είναι κάθε φορά τύπου data_t
data_t item;
// το next είναι δείκτης στο επόμενο αντικείμενο
// της λίστας
list *next;
public:
// ο δημιουργός (δέχεται ένα αντικείμενο data_t)
list(data_t d);
// για να προσθέσουμε ένα αντικείμενο στη λίστα χρησιμοποιούμε την add()
void add(list *node) {
node->next = this;
return;
}
// η get_next() επιστρέφει το επόμενο αντικείμενο στη λίστα
list *get_next() {
return next;
}
// η get_data επιστρέφει το αντικείμενο item. Ο τύπος θα είναι πάντα σωστός
(π.χ. // θα επιστρέφει char αν η data_t είναι char, string, κλπ.
data_t get_data() {
return item;
}
};
```

Παράδειγμα κλάσης λίστας αντικειμένων


```
// Στον ορισμό της κλάσης, απλώς δηλώσαμε τον δημιουργό
// Με τον παρακάτω τρόπο τον ορίζουμε κιόλας. Σημειώστε
// τη χρήση της λέξης template και των brackets <data_t>
// στο όνομα της κλάσης
template<class data_t> list<data_t>::list(data_t d) {
// ορισμός του item στην παράμετρό d
item = d;
// δεν έχουμε επόμενο αντικείμενο (πρώτο στοιχείο
next = 0;
}
int main() {
// Ορισμός ενός αρχικού αντικειμένου list.
// Το ρόλο της αφηρημένης κλάσης data_t παίρνει ο τύπος char
list<char> start('a');
// Δηλώνουμε και δύο δείκτες (pointers) της ίδιας κλάσης
list<char> *p, *last;
// θα χρησιμοποιήσουμε τον δείκτη last για να μετακινούμαστε
// στη λίστα, όσο μεγαλώνει
last = &start;
for (int i=1; i < 26; i++) {
// δημιούργησε ένα καινούριο κόμβο της λίστας με
// αντικείμενο το χαρακτήρα 'a' + i.
// Η δημιουργία είναι δυναμική (προσέξτε τη χρήση της new.
p = new list<char>('a' +i);
// πρόσθεσε το νέο αντικείμενο στον προηγούμενο κόμβο.
p->add(last);
// ο προηγούμενος κόμβος δεν είναι πλέον τελευταίος.
// όρισε τον τρέχοντα κόμβο να φαίνεται τελευταίος.
last = p;
}
```

Παράδειγμα κλάσης λίστας αντικειμένων


```
// ξεκίνα από την αρχή
p = &start;
while (p) {
// όσο έχουμε κόμβους στη λίστα (δηλαδή όσο η get_next()
// θα επιστρέφει μη μηδενικό αποτέλεσμα, τύπωσε το αντικείμενο
// που περιέχεται στον κόμβο.
cout << p->get_data() << ", ";
// προχώρα στον επόμενο κόμβο.
p = p->get_next();
}
cout << endl;
return 0;
}
```

Το πρόγραμμα αυτό θα τυπώσει το εξής αποτέλεσμα.

```
a, b, c, d, e, f, g, h, i, j, k, l, m, n, o, p, q, r, s, t, u, v, w, x, y, z,
```

Standard Template Library (STL)

- **Βιβλιοθήκη STL:** Σύνολο από έτοιμες κλάσεις και συναρτήσεις γενικής χρήσης που γλυτώνουν σημαντικό χρόνο από το προγραμματιστή καθώς προσφέρουν έναν εύκολο τρόπο να αντιμετωπιστούν τα περισσότερα θέματα γενικής φύσεως σε ένα πρόγραμμα.
- Έτσι ο προγραμματιστής έχει πλέον τη δυνατότητα να ασχοληθεί με την ουσία του προβλήματός του, **παρά** με το ποια ρουτίνα θα χρησιμοποιήσει για να ταξινομήσει ένα πίνακα από strings.
- Αποτελείται από τρεις κατηγορίες κλάσεων και μεθόδων: **τους containers, τους αλγόριθμους και τους iterators.**

Οι κλάσεις containers

- **Containers:** Αντικείμενα που χρησιμοποιούνται ως “δοχεία” για άλλα αντικείμενα (οποιουδήποτε είδους) και προσφέρουν συγκεκριμένα χαρακτηριστικά όσον αφορά το τρόπο πρόσβασης των αντικειμένων, τον τρόπο αποθήκευσης των αντικειμένων στη μνήμη, την αντιστοίχιση αντικειμένων σε “κλειδιά” (σχεσιακοί containers), κλπ.
- **Βασικοί containers:** vector, list, queue, stack και set, ενώ ορίζονται και οισχεσιακοί containers οι map, multimap.
- Οι containers επειδή είναι βασισμένοι στα πρότυπα, μπορούν να περιέχουν αντικείμενα οποιασδήποτε κλάσης.

Οι μέθοδοι algorithms

- Οι containers από μόνοι τους δε θα ήταν ιδιαίτερα χρήσιμοι, όχι περισσότερο από την ξεχωριστή υλοποίηση κάποιου προγραμματιστή.
- Αυτό που κάνει την STL να αποδεικνύεται αξεπέραστο εργαλείο είναι ο συνδυασμός των containers με τους αλγόριθμους (algorithms).
- Οι αλγόριθμοι είναι απλώς μέθοδοι που έχουν σχεδιαστεί να λειτουργούν πάνω στα αντικείμενα που περιέχονται στους containers, χωρίς όμως να απαιτούν κάποιο συγκεκριμένο τύπο, μόνο κάποια γενικά χαρακτηριστικά.

Οι μέθοδοι algorithms

- ***Π.χ.*** για να ταξινομήσουμε τα αντικείμενα ενός container, δεν είναι ανάγκη να απαιτήσουμε ο container να περιέχει string, απλώς να ορίζεται η πράξη της σύγκρισης για τα αντικείμενα αυτά (δηλαδή οι τελεστές <, ==, >).
- Η STL προσφέρει έτοιμους αλγόριθμους για πολλές διαδικασίες, όπως **ταξινόμηση** (sort, stable_sort), **αντιγραφή αντικειμένων από τον container** με ή χωρίς συνθήκη που πρέπει να πληρείται (copy, copy_backward, unique_copy, unique_copy_if), **ορισμός κάποιας τιμής στα αντικείμενα** (fill, fill_n), **εύρεση αντικειμένων στο container** (find, find_end, find_first_of, find_if), **καταμέτρηση αντικειμένων** με ή χωρίς συνθήκη (count, count_if), **εκτέλεση κάποιας ρουτίνας** για συγκεκριμένα αντικείμενα.

Οι μέθοδοι algorithms

- **Αλγόριθμοι διαδικασιών στην STL:**

ταξινόμηση (sort, stable_sort), **αντιγραφή αντικειμένων από τον container** με ή χωρίς συνθήκη που πρέπει να πληρείται (copy, copy_backward, unique_copy, unique_copy_if), **ορισμός κάποιας τιμής στα αντικείμενα** (fill, fill_n), **εύρεση αντικειμένων στο container** (find, find_end, find_first_of, find_if), **καταμέτρηση αντικειμένων** με ή χωρίς συνθήκη (count, count_if), **εκτέλεση κάποιας ρουτίνας** για συγκεκριμένα αντικείμενα (for_each, transform), **πράξεις συνόλων** (set_union, set_intersection), **εύρεση μεγίστων και ελαχίστων** (max_element, min_element), **διαγραφή ή αντικατάσταση αντικειμένων στο container** με ή χωρίς συνθήκη και με ή χωρίς αντιγραφή σε νέο container (remove, remove_if, remove_copy, remove_copy_if, replace, replace_if, replace_copy, replace_copy_if), κτλ.

Οι δείκτες iterators

- Κάθε αλγόριθμος δέχεται το αρχικό και το τελικό αντικείμενο του container στο οποίο θα πραγματοποιηθεί η εργασία ως παραμέτρους.
- Τα αντικείμενα αυτά δίνονται ως iterators, το τελικό στοιχείο της STL για να ολοκληρωθεί ως εργαλείο.
- **Πρακτικά οι iterators είναι pointers αλλά με κάποια ιδιαίτερη συμπεριφορά**, π.χ. υπάρχουν forward και backward iterators για μετακίνηση στα αντικείμενα ενός container μόνο προς τη μία κατεύθυνση, input και output iterators για χρήση σε streams κλπ.

Βιβλιογραφία

1. Εγχειρίδιο της C++, 2η Ελληνική έκδοση, Jesse Liberty, Γκιούρδας.
2. Μάθετε τη C++, 2η Ελληνική έκδοση, Jesse Liberty , Γκιούρδας.
3. Προγραμματισμός με τη γλώσσα C++ Μέρος Α, Αλεβίζος Θ., Έκδοση ΤΕΙ Καβάλας
4. C++ Αντικειμενοστραφής Προγραμματισμός Υπολογιστών Τομαράς Α., , Εκδόσεις Νέων Τεχνολογιών.
5. Ανακαλύψτε τη γλώσσα C, J. Purdum, Εκδόσεις Δίαυλος.
6. Εισαγωγή στο Συστηματικό Προγραμματισμό και στη γλώσσα C++, Σ. Μπαλτζής, εκδόσεις πανεπιστημίου Ιωαννίνων.
7. C++ From the beginning, Jan Skansholm, Addison Wesley.
8. The design and analysis of computer algorithms, A.V. AHO, J.E. HOPCROFT, J.D. ULLMANN, Addison Wesley 1974.
9. Structure and Interpretation of Computer Programs, H. ABELSON, G.J. SUSSMAN, J. SUSSMAN, MIT Press, Mc Graw Hill Book Company, 1985
10. The art of computer programming, D.E. KNUTH, Addison-Wesley.

Σημείωμα Αναφοράς

Σημείωμα Αδειοδότησης

[1] http://creativecommons.org/licenses/by-nc-nd/4.0/deed.el

Τέλος Ενότητας

Επεξεργασία: Ευάγγελος Καρβούνης

Άρτα, 2015

Ευρωπαϊκή Ένωση
Ευρωπαϊκό Κοινωνικό Ταμείο
ΕΠΙΧΕΙΡΗΣΙΑΚΟ ΠΡΟΓΡΑΜΜΑ
ΕΚΠΑΙΔΕΥΣΗ ΚΑΙ ΔΙΑ ΒΙΟΥ ΜΑΘΗΣΗ
επένδυση στην κοινωνία της γνώσης
ΥΠΟΥΡΓΕΙΟ ΠΑΙΔΕΙΑΣ & ΘΡΗΣΚΕΥΜΑΤΩΝ, ΠΟΛΙΤΙΣΜΟΥ & ΑΘΛΗΤΙΣΜΟΥ
ΕΙΔΙΚΗ ΥΠΗΡΕΣΙΑ ΔΙΑΧΕΙΡΙΣΗΣ
ΕΣΠΑ 2007-2013
ΕΥΡΩΠΑΪΚΟ ΚΟΙΝΩΝΙΚΟ ΤΑΜΕΙΟ
Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης

Τέλος Ενότητας

Τα πρότυπα (templates)