**Ανοικτά Ακαδημαϊκά Μαθήματα**

**Τεχνολογικό Εκπαιδευτικό Ίδρυμα Αθήνας**

**Αντικειμενοστρεφής Προγραμματισμός (Ε)**

**Ενότητα 5:** Δείκτες και διαχείριση μνήμης- Λύσεις

Κλειώ Σγουροπούλου

Τμήμα Μηχανικών Πληροφορικής ΤΕ

|  |  |
| --- | --- |
| Το περιεχόμενο του μαθήματος διατίθεται με άδεια Creative Commons εκτός και αν αναφέρεται διαφορετικά | 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. |

# Αντικειμενοστρεφής Προγραμματισμός- Εργαστήριο 5- Λύσεις

# Άσκηση 1

#include <iostream>

#include <string>

using namespace std;

class Person

{

char \*name;

char \*surname;

int age;

public:

Person(char \*\_name, char \*\_surname, int \_age)

{

cout << "constructor called" << endl; name = new char[strlen(\_name)+1]; strcpy(name, \_name);

surname = new char[strlen(\_surname)+1]; strcpy(surname, \_surname);

age = \_age;

}

Person()

{

cout << "default constructor called" << endl; name = new char[4];

strcpy(name, "N/A"); surname = new char[4]; strcpy(surname, "N/A");

age = 0;

}

~Person()

{

cout << "destructor called" << endl;

delete []name;

delete []surname;

}

void display()

{

cout << "Name: " << name << endl;

cout << "Surname: " << surname << endl;

cout << "Age: " << age << endl;

}

void setName(char \*\_name)

{

delete []name;

name = new char[strlen(\_name)+1]; strcpy(name, \_name);

}

void setSurname(char \*\_surname)

{

delete []surname;

surname = new char[strlen(\_surname)+1];

strcpy(surname, \_surname);

}

void setAge(int \_age)

{

age = \_age;

}

};

int \_tmain(int argc, \_TCHAR\* argv[])

{

Person p[10]; //καλείται 10 φορές ο default constructor

p[0].setName("John");

p[0].setSurname("Newman");

p[0].setAge(33);

p[0].display();

system("pause");

return 0;

}

# Ασκήσεις 2 & 3

Επειδή η κλάση μας δεν περιέχει copy constructor, θα υπάρξει πρόβλημα όταν περάσουμε ένα αντικείμενο Person ως παράμετρο σε συνάρτηση. Αν περάσουμε την παράμετρο «κατά τιμή» (by value) και όχι μέσω pointer ή reference (by reference), τότε δημιουργείται αντίγραφο του αντικειμένου.

Τα αντίγραφο αυτό θα δημιουργηθεί από τον default copy constructor, ο οποίος θα κάνει «ρηχή αντιγραφή» του αντικειμένου, με αποτέλεσμα το αρχικό αντικείμενο και το αντίγραφό του να περιέχουν pointers που δείχνουν στα ίδια strings (ή αλλιώς, στις ίδιες θέσεις μνήμης).

Όταν τελειώσει η συνάρτηση, τότε θα κληθεί ο destructor του Person και θα καταστρέψει το αντίγραφο. Εκτός από αυτό όμως θα σβήσει και τα strings που περιέχονται στο αντίγραφο. Καθώς όμως τα strings αυτά είναι κοινά για τα δύο αντικείμενα, το αρχικό αντικείμενο θα περιέχει πλέον pointers που δεν δείχνουν σε κάποια strings αλλά σε θέσεις μνήμης που δεν τους ανήκουν.

Ανάλογα με την πλατφόρμα και το σύστημα του καθενός, το πρόγραμμα ενδέχεται να τερματίσει ανώμαλα οποιαδήποτε στιγμή. Η λύση για το πρόβλημα αυτό είναι η υλοποίηση copy constructor που θα κάνει αντιγραφή αντικειμένων σε βάθος. Επίσης, θα πρέπει να υλοποιήσουμε και τον assignment operator,ώστε να αποφύγουμε τα προβλήματα και στην περίπτωση αντιγραφή αντικειμένων μέσω του τελεστή “=”.

#include <iostream>

#include <string>

using namespace std;

class Person

{

char \*name;

char \*surname;

int age;

public:

Person(char \*\_name, char \*\_surname, int \_age)

{

cout << "constructor called" << endl;

name = new char[strlen(\_name)+1];

strcpy(name, \_name);

surname = new char[strlen(\_surname)+1];

strcpy(surname, \_surname);

age = \_age;

}

Person()

{

cout << "default constructor called" << endl;

name = new char[4];

strcpy(name, "N/A"); surname = new char[4]; strcpy(surname, "N/A"); age = 0;

}

~Person()

{

cout << "destructor called" << endl;

delete []name;

delete []surname;

}

Person(const Person &p)

{

cout << "copy constructor called" << endl; name = new char[strlen(p.name)+1]; strcpy(name, p.name);

surname = new char[strlen(p.surname)+1]; strcpy(surname, p.surname);

age = p.age;

}

Person& operator=(const Person &p)

{

cout << "assignment operator called" << endl;

delete[]name;

delete []surname;

name = new char[strlen(p.name)+1]; strcpy(name, p.name);

surname = new char[strlen(p.surname)+1];

strcpy(surname, p.surname);

age = p.age;

return \*this;

}

void display()

{

cout << "Name: " << name << endl;

cout << "Surname: " << surname << endl;

cout << "Age: " << age << endl;

}

void setName(char \*\_name)

{

delete []name;

name = new char[strlen(\_name)+1];

strcpy(name, \_name);

}

void setSurname(char \*\_surname)

{

delete []surname;

surname = new char[strlen(\_surname)+1];

strcpy(surname, \_surname);

}

void setAge(int \_age)

{

age = \_age;

}

int getAge()

{

return age;

}

};

int calculateBirthYear(Person p)

{

int currentYear = 2012;

int birthYear = currentYear - p.getAge();

return birthYear;

}

int \_tmain(int argc, \_TCHAR\* argv[])

{

Person p;

p.setName("John"); p.setSurname("Newman"); p.setAge(33);

p.display();

cout << "Birth year: " << calculateBirthYear(p); Person q;

q = p;

q.display();

system("pause");

return 0;

}

|  |
| --- |
| **Ανοικτά Ακαδημαϊκά Μαθήματα**  **Τεχνολογικό Εκπαιδευτικό Ίδρυμα Αθήνας** |
| **Τέλος Ενότητας** |
| **Χρηματοδότηση**   * Το παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στα πλαίσια του εκπαιδευτικού έργου του διδάσκοντα. * Το έργο «**Ανοικτά Ακαδημαϊκά Μαθήματα στο ΤΕΙ Αθήνας**» έχει χρηματοδοτήσει μόνο τη αναδιαμόρφωση του εκπαιδευτικού υλικού. * 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. |

**Σημειώματα**

**Σημείωμα Αναφοράς**

Copyright ΤΕΙ Αθήνας, Κλειώ Σγουροπούλου, 2014. Κλειώ Σγουροπούλου. «Αντικειμενοστρεφής Προγραμματισμός». Ενότητα 5: «Δείκτες και διαχείριση μνήμης- Λύσεις ασκήσεων». Έκδοση: 1.0. Αθήνα 2014. Διαθέσιμο από τη δικτυακή διεύθυνση: [ocp.teiath.gr](https://ocp.teiath.gr/).

**Σημείωμα Αδειοδότησης**

Το παρόν υλικό διατίθεται με τους όρους της άδειας χρήσης Creative Commons Αναφορά, Μη Εμπορική Χρήση Παρόμοια Διανομή 4.0 [1] ή μεταγενέστερη, Διεθνής Έκδοση. Εξαιρούνται τα αυτοτελή έργα τρίτων π.χ. φωτογραφίες, διαγράμματα κ.λ.π., τα οποία εμπεριέχονται σε αυτό και τα οποία αναφέρονται μαζί με τους όρους χρήσης τους στο «Σημείωμα Χρήσης Έργων Τρίτων».

[](file:///C:\Users\pantelis\Downloads\%5b1%5d%20http:\creativecommons.org\licenses\by-nc-sa\4.0\)

[1] http://creativecommons.org/licenses/by-nc-sa/4.0/

Ως **Μη Εμπορική** ορίζεται η χρήση:

* που δεν περιλαμβάνει άμεσο ή έμμεσο οικονομικό όφελος από την χρήση του έργου, για το διανομέα του έργου και αδειοδόχο
* που δεν περιλαμβάνει οικονομική συναλλαγή ως προϋπόθεση για τη χρήση ή πρόσβαση στο έργο
* που δεν προσπορίζει στο διανομέα του έργου και αδειοδόχο έμμεσο οικονομικό όφελος (π.χ. διαφημίσεις) από την προβολή του έργου σε διαδικτυακό τόπο

Ο δικαιούχος μπορεί να παρέχει στον αδειοδόχο ξεχωριστή άδεια να χρησιμοποιεί το έργο για εμπορική χρήση, εφόσον αυτό του ζητηθεί.

**Διατήρηση Σημειωμάτων**

* Οποιαδήποτε αναπαραγωγή ή διασκευή του υλικού θα πρέπει να συμπεριλαμβάνει:
* Το Σημείωμα Αναφοράς
* Το Σημείωμα Αδειοδότησης
* Τη δήλωση Διατήρησης Σημειωμάτων
* Το Σημείωμα Χρήσης Έργων Τρίτων (εφόσον υπάρχει) μαζί με τους συνοδευόμενους υπερσυνδέσμους.