Αντικειμενοστρεφής Προγραμματισμός Ι - C++
7ο Φυλλάδιο εργαστηρίου
08/05/2023

Σκοπός: Προγραμματισμός C++ με χρήση αναφορών. Μηχανισμός κληρονομικότητας (inheritance) κλάσεων στον αντικειμενοστρεφή προγραμματισμό. Επίπεδα πρόσβασης μελών (public, protected, private), μέθοδοι δόμησης στην κληρονομικότητα, παράκαμψη ή υπερίσχυση (overriding) μεθόδων βασικής κλάσης.

Άσκηση 1η

α) Εξηγήστε αναλυτικά ποιο θα είναι το αποτέλεσμα της εκτέλεσης του παρακάτω προγράμματος. Δικαιολογήστε την απάντηση σας.

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

void f1(double& x, double& y) {
  cout << x << endl;
  cout << y << endl;

  x = x/2;
  y = 2* y;
}

int main() {
  double i=30, j=60;
  cout << i << ", " << j << endl;
  f1(i, j);
  cout << i << ", " << j << endl;

  return 0;
}
```

β) Μετατρέψτε το παραπάνω πρόγραμμα έτσι ώστε με την χρήση δεικτών και όχι αναφορών να έχετε το ίδιο αποτέλεσμα.

Άσκηση 2η

Συμπληρώστε το ακόλουθο C++ πρόγραμμα:

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

/* Προσθέτει 2 αριθμούς και επιστρέφει το αποτέλεσμα */
int add(int x, int y) {
  // Προσθέστε C++ κώδικα
}

/* Αντιστρέφει τις τιμές δύο μεταβλητών με χρήση δεικτών */
void swap(int *x, int *y) {
  int temp;
  // Προσθέστε C++ κώδικα
}
```

```
// Μειώνει κατά ένα την τιμή του x – χρήση αναφοράς
void dec(int &x) {
  // Προσθέστε C++ κώδικα
}

// Αυξάνει κατά ένα την τιμή του x – χρήση αναφοράς
void inc(int &x) {
  // Προσθέστε C++ κώδικα
}

int main() {
  /* Προσθέστε τον απαραίτητο C++ κώδικα που να επιδεικνύει την χρήση
     όλων των παραπάνω συναρτήσεων */
}
```

Άσκηση 3η

Υπάρχει λάθος στο παρακάτω πρόγραμμα; Αν υπάρχει, διορθώστε το ώστε να παράγει το αναμενόμενο αποτέλεσμα.

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

class Part {
private:
    int partnum;
public:
    Part(int partnum): partnum(partnum) {}
    int get_partnum() {
        return partnum;
    }

void set_partnum(int partnum) {
    this->partnum = partnum;
    }
};


// Πρωτότυπο συνάρτησης
void changePartNum(Part, int);

int main () {
Part p1(2345);
cout << p1.get_partnum() << endl;

changePartNum(p1, 9999); // Αλλαγή part number
cout << p1.get_partnum() << endl;
}

void changePartNum(Part p, int newpartnum) {
    p.set_partnum(newpartnum);
}
```

Άσκηση 4η

Εξηγήστε αναλυτικά την λειτουργία του παρακάτω προγράμματος. Ποιο θα είναι το αποτέλεσμα της εκτέλεσης του.

```cpp
#include <iostream>
using namespace std;

class Count {
private:
   int x;
public:
   void set_x(int x) {this->x = x;}
   void print() {
     cout << x << endl;
   }
};

int main() {
  Count counter;
  Count *counterPtr = &counter;
  Count &counterRef = counter;

  counter.set_x(1);
  counter.print();
  counterRef.set_x(2);
  counterRef.print();
  counterPtr->set_x(3);
  counterPtr->print();
  counter.print();

  return 0;
}
```

Άσκηση 5η

Υλοποιήστε κλάση στην C++ που αναπαριστά έναν άνθρωπο. Υλοποιείστε συνάρτηση μέλος της κλάσης που να δέχεται ως παράμετρο ένα αντικείμενο τύπου άνθρωπος και κάνοντας τους απαραίτητους ελέγχους, να επιστρέφει αν πρόκειται ή όχι για τον ίδιο άνθρωπο (ίδια στοιχεία με αυτόν που κάλεσε τη λειτουργία - σύγκριση αντικειμένων). Θεωρείτε ότι είναι σκόπιμο να χρησιμοποιηθεί αναφορά σε αντικείμενο ως παράμετρο σε αυτή την συνάρτηση;

Άσκηση 6η

Σχεδιάστε το διάγραμμα κλάσεων για το ακόλουθο σενάριο:
Οι εργαζόμενοι στο στρατό χωρίζονται σε πολιτικό προσωπικό, μόνιμοι στρατιωτικοί και έφεδροι στρατιώτες. Για καθένα εργαζόμενο γνωρίζουμε το όνομα και το επώνυμό του. Οι μόνιμοι έχουν επιπλέον βαθμό, θέση και καθήκοντα. Οι έφεδροι έχουν επιπλέον βαθμό, θέση και ημερομηνία απόλυσης. Το πολιτικό προσωπικό έχει επιπλέον γραφείο απασχόλησης και ημερομηνία πρόσληψης. Όλοι οι εργαζόμενοι μπορούν να ζητήσουν άδεια. Οι μόνιμοι και οι έφεδροι μπορούν να ζητήσουν μετάθεση αλλά με διαφορετική διαδικασία. Το πολιτικό προσωπικό μπορεί να κάνει αίτηση συνταξιοδότησης.

Άσκηση 7η

Ποια από τα παρακάτω δεν σχηματίζουν έγκυρα ζεύγη υπερκλάσης-υποκλάσης και γιατί;

- Τράπεζα - Λογαριασμός
- Οργανωτική Μονάδα - Τμήμα

- Λογαριασμός - Λογαριασμός_GR23456
- Άνθρωπος - Πελάτης
- Φοιτητής - Προπτυχιακός φοιτητής
- Ήπειρος - Χώρα
- Δήμος - Συνοικία

Άσκηση 8η

Στις παρακάτω δηλώσεις κλάσεων, η κλάση Sphere παράγεται από την κλάση Circle. Υπάρχει κάποιο λάθος; Αν ναι, τι προσθήκη θα έπρεπε να γίνει στην κλάση Circle ώστε το λάθος να αποκατασταθεί; Δώστε δύο εναλλακτικές λύσεις.

```
#include <iostream>
#include <cmath>
#define PI 3.141593
using namespace std;

class Circle {
    float aktina;
public:
    float emvado() {return pow(aktina, 2)*PI;}
    void set_aktina(float r) {aktina=r;}
};

class Sphere : public Circle {
public:
    float emvado() {return 4*pow(aktina, 2)*PI;}
    float ogos() {return 4/3.0*pow(aktina, 3)*PI};
};
```

Άσκηση 9η

Να δημιουργηθεί σε C++, μια κλάση με όνομα ***Employee*** και ιδιότητες το *όνομα* και το *ΑΦΜ* του εργαζόμενου. Η κλάση να περιλαμβάνει επίσης έναν *δομητή (constructor)* που να δέχεται ως ορίσματα το ονοματεπώνυμο και το ΑΦΜ και μια συνάρτηση με όνομα *display* που θα εμφανίζει τα στοιχεία ενός υπαλλήλου.

Επιπλέον ορίστε 2 ακόμα κλάσεις που θα αναπαριστούν έναν ***μισθωτό*** και έναν ***ωρομίσθιο*** υπάλληλο. Κάθε μισθωτός υπάλληλος πληρώνεται με έναν *μηνιαίο μισθό*. Αντιθέτως ο ωρομίσθιος υπάλληλος πληρώνεται ανάλογα με τις *ώρες εργασίας*. Η αμοιβή για κάθε ώρα εργασίας είναι 8€.

Για κάθε τύπο υπαλλήλου θα πρέπει να υπάρχει η δυνατότητα εμφάνισης των προσωπικών στοιχείων του υπαλλήλου και της συνολικής αμοιβής για τον τρέχοντα μήνα (συνάρτηση μέλος *display*).

Υλοποιήστε πρόγραμμα στο οποίο θα ορίζονται 2 υπάλληλοι, 1 μισθωτός και 1 ωρομίσθιος. Να εκτυπώσετε τα στοιχεία κάθε υπαλλήλου και το ποσό πληρωμής τους.

Άσκηση 10η

Να ορίσετε την κλάση ***Pet*** η οποία θα αποθηκεύει το όνομα, την ηλικία και το βάρος του ζώου. Να προσθέσετε τους απαραίτητους δομητές και συνάρτηση προβολής στοιχείων. Να ορίσετε

επίσης τη συνάρτηση *getlifespan* η οποία θα επιστρέφει ένα αλφαριθμητικό με την τιμή «άγνωστο προσδόκιμο ζωής». Στη συνέχεια να ορίσετε την κλάση ***Dog*** η οποία κληρονομεί από την κλάση Pet. Η κλάση Dog θα πρέπει να έχει μία ιδιωτική μεταβλητή μέλος με το όνομα *breed* η οποία θα περιέχει τη ράτσα του σκύλου. Να προσθέσετε τους απαραίτητους δομητές και συναρτήσεις προσπέλασης και μεταβολής για τη μεταβλητή breed. Να ορίσετε εκ νέου τη συνάρτηση *getlifespan* έτσι ώστε να επιστρέφει την τιμή «Περίπου 7 χρόνια» αν το βάρος του σκύλου είναι πάνω από 45 kg και «Περίπου 13 χρόνια» αν το βάρος του σκύλου είναι κάτω από 45 κιλά.

Στη συνέχεια να ορίσετε την κλάση ***Turtle*** η οποία θα κληρονομεί και αυτή από την κλάση Pet. Να ορίσετε εκ νέου τη συνάρτηση *getlifespan*, έτσι ώστε να επιστρέφει την τιμή «Πάρα πολλά χρόνια».

Τέλος να γράψετε ένα δοκιμαστικό πρόγραμμα το οποίο θα δημιουργεί αντικείμενα Dog και Turtle που θα ελέγχουν τις συναρτήσεις που έχουν κληρονομηθεί και αυτές που έχουν οριστεί εκ νέου.