

C++

3 Set C++ 2020

Γιώργος Ντάκος: \\\A.M: 1059569 /// Έτος: Γ'
24/5/2020

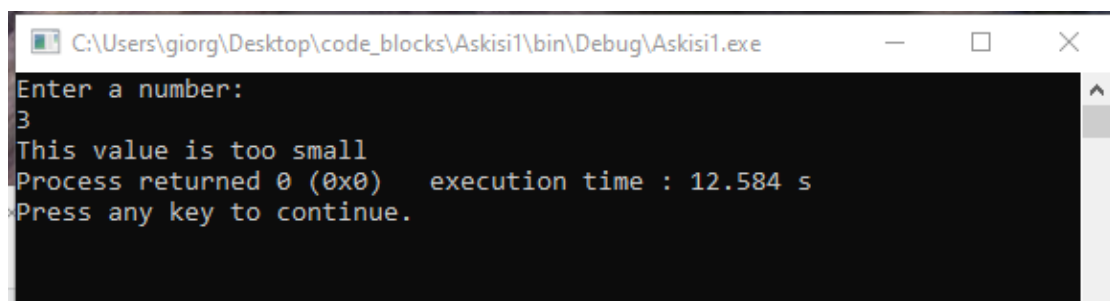
ΕΡΓΑΣΤΗΡΙΑΚΕΣ ΑΣΚΗΣΕΙΣ(C++)

Άσκηση 1

Ερώτημα 1:

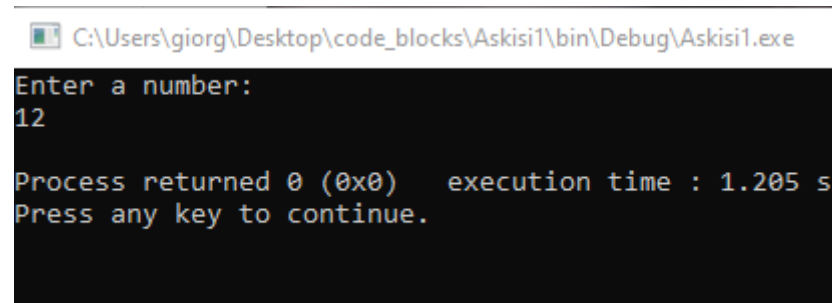
Γράφουμε το πρόγραμμα μας και μετά το κάνουμε compile και περνούμε τα έξεις αποτελέσματα για κάθε περίπτωση:

Για value=3:



```
C:\Users\giorg\Desktop\code_blocks\Askisi1\bin\Debug\Askisi1.exe
Enter a number:
3
This value is too small
Process returned 0 (0x0) execution time : 12.584 s
Press any key to continue.
```

Για value=12:



```
C:\Users\giorg\Desktop\code_blocks\Askisi1\bin\Debug\Askisi1.exe
Enter a number:
12
Process returned 0 (0x0) execution time : 1.205 s
Press any key to continue.
```

Στην αρχή του προγράμματος έχουμε την καθιερωμένη βιβλιοθήκη <iostream> που έχει τα καθιερωμένα αντικείμενα και πράξεις iostream. Στην συνέχεια με την using namespace std κάνουμε ορατά τα ονόματα του std χωρίς το πρόθεμα std::. Έπειτα ορίζουμε μια συνάρτηση την main χωρίς κάποιο όρισμα και η οποία κάνει τα εξής: Πρώτα πολλά γράφεται στο ρεύμα εξόδου cout το μήνυμα που βρίσκεται μέσα στις αγκύλες χάρης τα << και μετρά με την endl ορίζουμε το τέλος της γραμμής οπότε πάμε σε καινούργια. Μετά ορίζουμε μια

μεταβλητή τύπου `int` και με το ρεύμα εισόδου `cin` γράφουμε την τιμή που θα πάρει η `value` από το `command line`. Στην συνέχεια ακολουθεί μια συνθήκη `if` που αν η μεταβλητή `value > 10` τότε τυπώνεται το αντίστοιχο μήνυμα. Στο τέλος του προγράμματος επιστρέφουμε την τιμή `0`.

Για να εμφανίζει το μήνυμα «This is a big enough number!» όταν θα ισχύει η συνθήκη `value > 10`, κάνουμε τις παρακάτω αλλαγές που φαίνεται στην εικόνα :

```
#include <iostream>
using namespace std;
int main()
{
    cout << "Enter a number: " << endl;
    int value;
    cin >> value;
    if(value < 10)
    {
        cout << "This value is too small";
    }

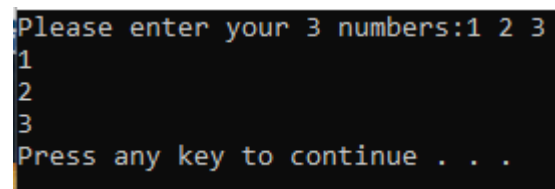
    //οι αλλαγές που χρειάζεται το 1ο ερώτημα
    else if(value > 10)
    {
        cout << "This is a big enough number!";
    }
    return 0;
}
```

Ερώτημα 2:

Στον κώδικα αυτό περιλαμβάνουμε μια νέα βιβλιοθήκη την `<cstdlib>` η οποία περιέχει συναρτήσεις κατανομής μνήμης της C. Στην συνέχεια ορίζουμε μια συνάρτηση `main` με 2 μεταβλητές την μια τύπου `int` και την άλλη τύπου πίνακα δεικτών χαρακτήρων. Αυτό που κάνει το πρόγραμμα είναι να δέχεται 3 ακέραιους αριθμούς και να μας τους τυπώνει στην συνέχεια με την σειρά που τους βάλαμε τον έναν κάτω από τον άλλο (αυτό γιατί μετά από κάθε τιμή που τυπώνει δέχεται το όρισμα `\n` που δηλώνει νέα γραμμή). Τέλος με

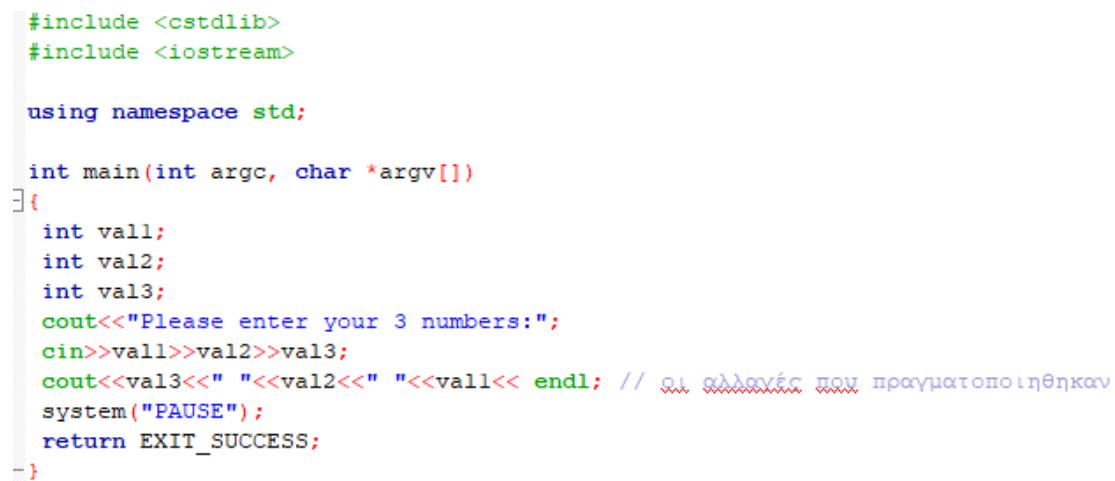
system("PAUSE") σταματά το περιβάλλον command line που τρέχει το πρόγραμμα(αυτή η εντολή ισχύει για windows) και με την σταθερά return EXIT_SUCCESS επιστρέφουμε την επιτυχή εκτέλεση του προγράμματος.

Τα αποτελέσματα:



```
Please enter your 3 numbers:1 2 3
1
2
3
Press any key to continue . . .
```

Για να εμφανίζει τους αριθμούς που δοθήκαν με αναστροφή σειρά πριν τερματίσει το πρόγραμμα κάνουμε τις παρακάτω αλλαγές που φαίνεται στην εικόνα:

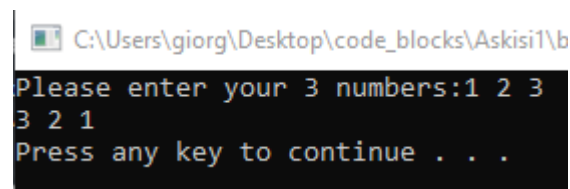


```
#include <cstdlib>
#include <iostream>

using namespace std;

int main(int argc, char *argv[])
{
    int val1;
    int val2;
    int val3;
    cout<<"Please enter your 3 numbers:";
    cin>>val1>>val2>>val3;
    cout<<val3<<" "<<val2<<" "<<val1<< endl; // οι αλλαγές που πραγματοποιήθηκαν
    system("PAUSE");
    return EXIT_SUCCESS;
}
```

Τα αποτελέσματα:



```
C:\Users\giorg\Desktop\code_blocks\Askisi1\b
Please enter your 3 numbers:1 2 3
3 2 1
Press any key to continue . . .
```

Ερώτημα 3:

Το συγκεκριμένο πρόγραμμα έχει ως ορίσματα 2 δισδιάστατους πίνακες που ο καθένας έχει από 2 γραμμές και 2 στήλες όπου δέχονται ακέραιους αριθμούς και άλλες 2

μεταβλητές. Στην συνέχεια έχει 2 for loop όπου το ένα είναι εμφωλευμένο μέσα στο άλλο οπότε το μήνυμα να εισάγουμε 2 αριθμούς θα τυπωθεί 2 φορές και επίσης με το 2^ο for θα όπου κρατάει τις στήλες του πίνακα ενώ το 1^ο for τις γραμμές θα συμπληρώσουμε όλα τα στοιχεία του πίνακα. Αυτά ισχύουν για τον πίνακα first. Τα ίδια συμβαίνουν και για τον πίνακα second. Επίσης κάθε φορά το πρόγραμμα μας δείχνει σε ποια γραμμή του πίνακα βρισκόμαστε. Στο τέλος προσθέτει τα στοιχεία του 1^{ου} πίνακα με τα στοιχεία του 2^{ου} πίνακα απλά δεν τυπώνονται τα αποτελέσματα.

Τα αποτελέσματα:

```
First set of numbers:
Enter two integers: 1
1 2
Enter two integers: 2
1 2

Second set of numbers:
Enter two more integers: 1
1 2
Enter two more integers: 2
1 2

Process returned 0 (0x0)   execution time : 46.046 s
Press any key to continue.
```

Για να πάρουμε τα δεδομένα του πίνακα 1 μετά την πρόσθεση των στοιχείων του με τα στοιχεία του δευτέρου πίνακα δημιουργούμε ένα εμφωλευμένο for όπου το πρώτο for θα είναι για τις γραμμές του πίνακα και το δεύτερο for για τις στήλες του πίνακα όπου στο δεύτερο for θα έχουμε την εκτύπωση των στοιχείων του πίνακα. Επίσης για να έχουμε δυο κοινές μεταβλητές για τους πίνακες που θα καθορίζουν τα μεγέθη των πινάκων, χρησιμοποιούμε την const μετά τον τύπο δεδομένων της κάθε σταθεράς. Παρακάτω φαίνεται ο κώδικας με τις αλλαγές:

```

1  #include<iostream>
2
3  using namespace std;
4  int main()
5  {
6      int const r=2;
7      int const c=2;
8
9      cout<<"Give the number of rows for both arrays"<<endl;
10
11     cout<<"\nGive the number of columns for both arrays"<<endl;
12
13
14     cout<<"First set of numbers:"<<endl;
15     int first[r][c], second[r][c];
16     int i,j;
17     for( i=0;i<r;i++){
18         cout<<"Enter two integers: "<<i+1<<endl;
19         for( j=0;j<c;j++){
20             cin>>first[i][j];
21         }
22     }
23     cout<<"\n\nSecond set of numbers:"<<endl;
24     for( i=0;i<r;i++){
25         cout<<"Enter two more integers: "<<i+1<<endl;
26         for( j=0;j<c;j++){
27             cin>>second[i][j];
28         }
29     }
30     for( i=0;i<r;i++){
31         for( j=0;j<c;j++){
32             first[i][j]=first[i][j]+second[i][j];
33         }
34     }
35     cout<<"\nElements of first array after the sum with the Elements of second Array:"<<endl;
36     for(i=0; i<r; i++)
37     {
38         for(j=0; j<c; j++)
39         {
40             cout<<first[i][j]<<" ";
41         }
42         cout<<"\n";
43     }
44     return 0;
45 }
46

```

Τα αποτελέσματα:

```

Give the number of columns for both arrays
First set of numbers:
Enter two integers: 1
1 2
Enter two integers: 2
1 2

Second set of numbers:
Enter two more integers: 1
1 2
Enter two more integers: 2
1 2

Elements of first array after the sum with the Elements of second Array:
2 4
2 4

Process returned 0 (0x0)   execution time : 16.531 s
Press any key to continue.

```

Για να δώσει ο χρήστης τις διαστάσεις των πινάκων κάνουμε τις εξής αλλαγές που φαίνονται παρακάτω:

```
#include<iostream>

using namespace std;
int main()
{
    int r,c;

    cout<<"Give the number of rows for both arrays"<<endl;
    cin>>r;
    cout<<"\nGive the number of columns for both arrays"<<endl;
    cin>>c;

    cout<<"First set of numbers:"<<endl;
    int first[r][c], second[r][c];
    int i,j;
    for( i=0;i<r;i++){
        cout<<"Enter two integers: "<<i+1<<endl;
        for( j=0;j<c;j++){
            cin>>first[i][j];
        }
    }
    cout<<"\n\nSecond set of numbers:"<<endl;
    for( i=0;i<r;i++){
        cout<<"Enter two more integers: "<<i+1<<endl;
        for( j=0;j<c;j++){
            cin>>second[i][j];
        }
    }
    for( i=0;i<r;i++){
        for( j=0;j<c;j++){
            first[i][j]=first[i][j]+second[i][j];
        }
    }
    cout<<"\nElements of first array after the sum with the Elements of second Array:"<<endl;
    for(i=0; i<r; i++)
    {
        for(j=0; j<c; j++)
        {
            cout<<first[i][j]<<" ";
        }
        cout<<"\n";
    }
    return 0;
}
```

Τα αποτελέσματα:

```

Give the number of rows for both arrays
2

Give the number of columns for both arrays
2
First set of numbers:
Enter two integers: 1
1 2
Enter two integers: 2
1 2


Second set of numbers:
Enter two more integers: 1
1 2
Enter two more integers: 2
1 2


Elements of first array after the sum with the Elements of second Array:
2 4
2 4


Process returned 0 (0x0)   execution time : 12.139 s
Press any key to continue.

```

Ερώτημα 4:

Δημιουργούμε την κλάση όπως μας ζητάει η άσκηση. Στην συνέχεια δηλώνουμε στην main μια μεταβλητή ογκος τύπου double η οποία θα μας παρέχει τον όγκο του κάθε αντικειμένου. Έπειτα δημιουργούμε τα αντικείμενα τα οποία είναι τύπου Kouti και μετά αρχικοποιούμε τις μεταβλητές των αντικειμένων που έχουν κληρονομήσει από την κλάση. Στο τέλος υπολογίζουμε τον όγκο ξεχωριστά για κάθε αντικείμενο και διατυπώνουμε τα ανάλογα μηνύματα.

Τα αποτελέσματα φαίνονται παρακάτω:

```

O ogkos toy koutiou A einai:38.4
O ogkos toy koutiou B einai:50


Process returned 0 (0x0)   execution time : 0.100 s
Press any key to continue.

```

Τώρα για την δεύτερη περίπτωση αφού έχουμε private τις μεταβλητές θα φτιάξουμε μια public μέθοδο τύπου double την calculateOgkos() με 3 ορίσματα τύπου double. Έπειτα θα

κάνουμε τις private μεταβλητές να ισούνται με αυτές των ορισμάτων της μεθόδου και στο τέλος της μεθόδου επιστρέφουμε τον όγκο του αντικειμένου. Μέσα στην main δημιουργούμε τα αντικείμενα KoutiA και KoutiB. Έπειτα με σωστά μηνύματα εξόδου και τις σωστές τιμές στα ορίσματα της calculateOgkos() ξεχωριστά για κάθε αντικείμενο παίρνουμε τα παρακάτω αποτελέσματα:

```
O ogkos toy koutiou A einai:38.4
O ogkos toy koutiou B einai:50

Process returned 0 (0x0)   execution time : 0.100 s
Press any key to continue.
```

Ερώτημα 5:

Προσθέτουμε την υπερφόρτωση του τελεστή + στο public κομμάτι της κλάσης το οποίο έχει ως όρισμα ένα σταθερό δείκτη τύπου Kouti. Μετά δημιουργούμε ένα αντικείμενο kouti μέσα στην υπερφόρτωση και αρχικοποιούμε τις τιμές του αντικειμένου με τον εξής τρόπο: Το αντικείμενο κουτί καλεί την κάθε private μεταβλητή μέσω της «.» και τα βάζει να ισούνται το καθένα με την πρόσθεση του αντίστοιχου μεγέθους της private με το μέγεθος που δείχνει ο δείκτης b. Με αυτό τρόπο επιτρέπουμε την πρόσθεση αντικειμένων. Μετά βγάζουμε τα ορίσματα από την calculateOgkos() και αφήνουμε μόνο να μας επιστρέφει τον όγκο του αντικειμένου. Τα μεγέθη τώρα θα τα ορίζουμε μέσα από τις 3 καινούργιες μεθόδους, setMikos(), setPlatos(), setYpsos(), όπου κάθε μια έχει το δικό της όρισμα το οποίο το βάζει να ισούνται με αντίστοιχο το μέγεθος. Μέσα στην main κάνουμε της απαραίτητες αλλαγές και καλούμε τις 3 παραπάνω μεθόδους με τα καταλληλά ορίσματα και για τα 2 αντικείμενα KoutiA και KoutiB. Μετά δημιουργούμε ένα 3^ο αντικείμενο KoutiC του οποίου τα μεγέθη θα ισούνται με την πρόσθεση των μεγεθών των 2

άλλων αντικείμενων δηλαδή θα γράψουμε την εντολή $KoutiC=KoutiA+KoutiB$. Στο τέλος βάζουμε ακόμα μια `cout` που θα μας τυπώνει τον όγκο του κουτιού `KoutiC`.

Τα αποτελέσματα:

```
O ogkos toy koutiou A einai:38.4
O ogkos toy koutiou B einai:50
O ogkos toy koutiou C einai:356.4

Process returned 0 (0x0)   execution time : 0.019 s
Press any key to continue.
```

Άσκηση 2

Ερώτημα 1:

Πρώτα πολλά εκτελούμε τον κώδικα σύμφωνα με αυτά που μας ζητάει η άσκηση και συμπεραίνουμε τα εξής: Η βιβλιοθήκη `vector` είναι μια καθιερωμένη βιβλιοθήκη προτύπων (STL) η οποία δημιουργεί μια δομή δεδομένων με τις ιδιότητες ενός δυναμικού πίνακα. Με την συνάρτηση `vec.size()` μας επιστρέφει τον αριθμό των στοιχείων που υπάρχουν στο διάνυσμα. Έτσι στην αρχή η παραπάνω συνάρτηση που την καλούμε μας επιστρέφει 0. Μετα με μια `for` εισάγουμε τα στοιχεία μας μέσα στο διάνυσμα μέσω της συνάρτησης `push.back()` όπου εισάγει ένα στοιχείο στο τέλος της δομής και εκχωρεί αυτόματα μνήμη αν αυτό είναι αναγκαίο. Οπότε μετά το `for` ξανά καλούμε την συνάρτηση `vec.size()` η οποία περιμένουμε να μας επιστρέψει άλλο μέγεθος αφού δώσαμε στοιχεία στο διάνυσμα μας μέσω της `for`. Τέλος πάλι με ένα `for`

εκτυπώνουμε τα περιεχόμενα της κάθε θέσης του διανύσματος.

Τα αποτελέσματά:

```
vector size = 0
extended vector size = 7
Vector [0] = 0
Vector [1] = 1
Vector [2] = 2
Vector [3] = 3
Vector [4] = 4
Vector [5] = 5
Vector [6] = 6

Process returned 0 (0x0)   execution time : 0.037 s
Press any key to continue.
```

Τώρα για να παίρνει τυχαίους αριθμούς το διάνυσμα μας απλά στην συνάρτηση `push.back()` αντί να βάλουμε για όρισμα το `counter` της `for` θα βάλουμε την συνάρτηση `rand()` δηλαδή θα έχουμε την εντολή `vec.push_back(rand())`.

Τα αποτελέσματα:

```
vector size = 0
extended vector size = 7
Vector [0] = 41
Vector [1] = 18467
Vector [2] = 6334
Vector [3] = 26500
Vector [4] = 19169
Vector [5] = 15724
Vector [6] = 11478

Process returned 0 (0x0)   execution time : 0.040 s
Press any key to continue.
```

Στην συνέχεια κάνοντας τις αλλαγές που μας ζητάει η άσκηση συμπεραίνουμε τα εξής: Με την χρήση του `iterator` ορίζουμε έναν δείκτη `n` τύπου `int` που δείχνει στο πρώτο στοιχείο του διανύσματος(`begin()`) και μετά με μια `while` όσο ο `iterator` είναι διάφορος του τέλους του τελευταίου στοιχείου του

διανύσματος μας τυπώνει τα περιεχόμενα της θέσης στην οποία βρίσκεται ο δείκτης v.

Τα αποτελέσματα:

```
vector size = 0
extended vector size = 7
value of v = 41
value of v = 18467
value of v = 6334
value of v = 26500
value of v = 19169
value of v = 15724
value of v = 11478

Process returned 0 (0x0)   execution time : 0.073 s
Press any key to continue.
```

Κάνοντας πάλι τώρα τις αλλαγές που μας ζητάει η άσκηση έχουμε τα εξής: Με την `resize()` αλλάζουμε το μέγεθος του διανύσματος οπότε το καλούμε με το τρόπο αυτό: `vec.resize(5)` και μετά με την χρήση πάλι της `size()` τυπώνουμε το νέο μέγεθος του διανύσματος. Τώρα αν την καλέσουμε με αυτό τον τρόπο (`resize(10,5)`) γίνεται το εξής: Το πρώτο όρισμα είναι αυτό που θα αλλάξει το μέγεθος το διανύσματος, το δεύτερο όρισμα της `resize` αυτό που μας λέει είναι ότι αν δημιουργηθούν παραπάνω θέσεις στο διάνυσμα από πριν τότε θα πάρουν την τιμή του ορίσματος αυτού:

Τα αποτελέσματα και για τις 2 περιπτώσεις:

```
vector size = 0
extended vector size = 7
reduced vector size = 5
value of v = 41
value of v = 18467
value of v = 6334
value of v = 26500
value of v = 19169

Process returned 0 (0x0)   execution time : 0.039 s
Press any key to continue.
```

```
vector size = 0
extended vector size = 7
reduced vector size = 10
value of v = 41
value of v = 18467
value of v = 6334
value of v = 26500
value of v = 19169
value of v = 15724
value of v = 11478
value of v = 5
value of v = 5
value of v = 5

Process returned 0 (0x0)   execution time : 0.039 s
Press any key to continue.
```

Ερώτημα 2:

Σύμφωνα με τα ζητούμενα της άσκησης παρατηρούμε ότι οι `setIpsos` και `getIpsos` δεν δημιουργούν κάποιο πρόβλημα με τον τρόπο που έχουν δημιουργηθεί στον κώδικα και αυτό γιατί τις έχουμε ορίσει μέσα στην κλάση και δημιουργούμε το σώμα τους καλώντας την κλάση με την μέθοδο έξω από το ορισμό της κλάσης.

Τα αποτελέσματα:

```
Ogkos gia Kouti Small: 4
Ogkos gia Kouti Big: 1560

Process returned 0 (0x0)   execution time : 0.018 s
Press any key to continue.
```

Τώρα αν κάνουμε τις αλλαγές που μας ζητάει η άσκηση μας εμφανίζει κάποια errors. Αυτό συμβαίνει διότι από την στιγμή που έχουμε δηλώσει ως `private` τις μεταβλητές της κλάσης δεν μπορούμε να τις καλέσουμε μέσα στην `main` αφού είναι προσπελάσιμες μόνο από την κλάση `Kouti`. Οπότε αυτό που κάνουμε είναι να ορίσουμε κάποιες συναρτήσεις οι οποίες θα

είναι public οπότε θα είναι προσπελάσιμες και από την main μέσα από τις οποίες θα θέτουμε τις τιμές των μεταβλητών του κουτιού αλλά και να μας επιστρέφονται αυτές οι τιμές και έτσι θα πραγματοποιήσουμε τον σκοπό της άσκησης. Ο κωδικός με τις αλλαγές θα παρατεθεί στο παράρτημα Α. Τα αποτελέσματα παραμένουν τα ίδια.

Ερώτημα 3:

Βλέπουμε ότι αυτό που κάνει ο κώδικας είναι το εξής. Έχουμε ορίσει μια κλάση Polygon με protected 2 μεταβλητές οπότε αυτές είναι προσπελάσιμες από τις υποκλάσεις της Polygon αλλά και από την ίδια την κλάση. Επίσης έχει μια μέθοδο public από την οποία μπορούμε και δίνουμε τιμές στις μεταβλητές της Polygon. Μετά έχουμε μια υποκλάση της κλάσης Polygon η οποία έχει μια public μέθοδο που μας επιστρέφει τον εμβαδόν ενός ορθογωνίου.

Τα αποτελέσματα:

```
Emvadon orthogoniou: 40
Process returned 0 (0x0)   execution time : 0.041 s
Press any key to continue.
```

Φτιάχνουμε την κλάση Triangle που κληρονομεί από την Polygon και δημιουργούμε μια μέθοδο area() που θα μας επιστρέφει το εμβαδόν του τριγώνου. Τέλος κάνουμε και απαραίτητες αλλαγές στην main και έχουμε τα εξής αποτελέσματα:

```
Emvadon orthogoniou: 40
Emvadon trigwnoy: 12
Process returned 0 (0x0)   execution time : 0.038 s
Press any key to continue.
```

Ερώτημα 4:

Κάνουμε τις αλλαγές που μας ζητάει η άσκηση και συμπεραίνουμε τα εξής: Η κλάση Rectangle κληρονομεί και από την κλάση Polygon αλλά και από την PaintCost κάτι το οποίο δεν γίνεται στην java γιατί από όσο ξέρουμε μια subclass κληρονομεί μόνο από μια υπερκλάση.

Τα αποτελέσματα:

```
Emvadon orthogoniou: 40
Synoliko kostos xrwmatos: 2800euro .
Emvadon trigwnoy: 12

Process returned 0 (0x0)   execution time : 0.018 s
Press any key to continue.
```

Ερώτημα 5:

Εκτελούμε τον κώδικα σύμφωνα με την άσκηση και παρατηρούμε ότι παίρνουμε κάποια μη αναμενόμενα αποτελέσματα δηλαδή ενώ εμείς περιμέναμε να πάρουμε το εμβαδόν του αντικειμένου rec παίρνουμε κάτι τελείως διαφορετικό που φαίνεται παρακάτω:

```
This is area as computed by the Polygon class
-----
Process exited after 0.04585 seconds with return value 0
Press any key to continue . . .
```

Αυτό συμβαίνει για το λόγο ότι έχουμε ένα αντικείμενο polygon δείκτη το οποίο δείχνει στο rec και στην συνάρτηση area() όμως επειδή έχουν κοινό όνομα με την συνάρτηση εμβαδού στην Rectangle και την Polygon ο compiler κάνει κλήση της συνάρτησης της Polygon. Αυτό λέγεται early binding (πολυμορφισμός χρόνου μεταγλώττισης). Όπως υποδηλώνει το όνομα, ο μεταγλωττιστής συσχετίζει απευθείας

μια διεύθυνση στην κλήση συνάρτησης. Αντικαθιστά την κλήση με μια οδηγία γλώσσας μηχανής που λέει στο κεντρικό πλαίσιο να μεταβεί στη διεύθυνση της λειτουργίας.

Τώρα βάζοντας την εντολή `virtual`(εικονική) στην συνάρτηση `area()` της `Polygon` παρατηρούμε ότι έχουμε διαφορετικά αποτελέσματα τα οποία φαίνονται παρακάτω:

```
This is area as computed by the Rectangle class
-----
Process exited after 0.04803 seconds with return value 0
Press any key to continue . . .
```

Με αυτό τον τρόπο υλοποιήσαμε έναν μηχανισμό late binding δηλαδή ο μεταγλώττισης προσθέτει κώδικα που προσδιορίζει το είδος του αντικειμένου κατά το χρόνο εκτέλεσης και στη συνέχεια ταιριάζει με την κλήση με τον σωστό ορισμό λειτουργίας. Αυτό έγινε δηλώνοντας τη `area()` ως εικονική συνάρτηση. Τώρα οι εικονικές συναρτήσεις μας επιτρέπουν να δημιουργήσουμε μια λίστα με δείκτες κλάσης βάσης και μεθόδους κλήσεων οποιασδήποτε από τις παραγόμενες τάξεις χωρίς καν να γνωρίσουμε το είδος του παραγομένου αντικειμένου κλάσης.

C++ SET 3:

Άσκηση 1

Ερώτημα 1:

Κανουμε τις αλλαγες που μας ζηταει(ολοκληρος ο κωδικας θα παρατεθει στο τελος).

Ερώτημα 2:

Έχοντας δημιουργήσει ως protected τον κατασκευαστή της κλάσης Animal και οι τρεις κλάσεις Bird, Fish, Dog κληρονομούν από την Animal τότε ως protected ο κατασκευαστής Animal με τα ορίσματα μπορεί να κληρονομηθεί και από τις υπόλοιπες υποκλίσεις και να αρχικοποιήσουν τις ανάλογες τιμές τους. Τώρα αν κάνουμε τον κατασκευαστή Animal private θα έχουμε πρόβλημα_διότι τότε ο κατασκευαστής θα είναι προσπελάσιμος μόνο από την ιδιά του την κλάση και όχι από τις υποκλάσεις του.

Ερώτημα 3:

Δημιουργούμε την main όπως μας ζητάει το ερώτημα και μεταγλωττίζουμε.

Ερώτημα 4:

Εκτελούμε τον κώδικα και παίρνουμε τα εξής αποτελέσματα:

```
Animal
Dog
Bird
Fish

-----
Process exited after 0.04452 seconds with return value 0
Press any key to continue . . .
```

Όπως βλέπουμε η getClass() μας έχει επιστρέψει κάθε φορά και διαφορετική κλάση. Αυτό έγινε διότι κάθε αντικείμενο είναι διαφορετικής κλάσης και καλεί την getClass της δική της κλάσης δηλαδή το αντικείμενο a που είναι τύπου Animal θα καλέσει την getClass() της Animal ενώ το d που είναι τύπου Dog θα καλέσει την getClass() της Dog. Το ίδιο ισχύει και για τα αντικείμενα b, f.

Ερώτημα 5:

Κάνοντας τις αλλαγές που μας ζητάει το ερώτημα παρατηρούμε ότι αντί να μας επιστρέψει την getClass της κλάσης Dog μας επιστρέφει την getClass της Animal. Αυτό έγινε διότι το αντικείμενο a όταν το εξισώνουμε με το αντικείμενο d αυτό που γίνεται είναι τα ορίσματα του d να μεταφερθούν στο a και όχι να αλλάξει η κλάση στην οποία ανήκει το αντικείμενο a. Οπότε και γι' αυτό πήραμε τα παρακάτω αποτελέσματα:

```
Animal
Dog
Bird
Fish
Animal
-----
Process exited after 0.04553 seconds with return value 0
Press any key to continue . . .
```

Ερώτημα 6:

Κάνοντας τις απαραίτητες αλλαγές και εκτελώντας τον κώδικα παίρνουμε τα παρακάτω αποτελέσματα:

```
This is a Animal. It has 3 legs and sound:
This is a Animal. It has 4 legs and sound: bark
This is a Animal. It has 2 legs and sound: chirp
This is a Animal. It has 0 legs and sound: blop
This is a Animal. It has 4 legs and sound: bark
-----
Process exited after 0.04726 seconds with return value 0
Press any key to continue . . .
```

Παρατηρούμε πρώτα πολλά ότι το αντικείμενο a πήρε κάποια ορίσματα δηλαδή δεν περιμέναμε να μας επιστρέψει πόδια και ήχο. Αυτό έγινε διότι από το public κατασκευαστή Animal όταν καλούμε την println απλά δίνει κάποιες default τιμές στο αντικείμενο μας. Μετά ήταν αναμενόμενο για κάθε αντικείμενο η printfInfo() να μας επιστρέψει τις ανάλογες τιμές

των legs και sound.Από την άλλη βέβαια περιμέναμε για κάθε αντικείμενο να πάρουμε και την αντίστοιχη κλάση αντί αυτού μας επιστρέφει μόνο την κλάση Animal.Αυτό γίνεται διότι η printInfo() είναι ορισμένη μέσα στην Animal οπότε η τιμή που θα επιστρέψει για την getClass θα είναι αυτή που είναι ορισμένη μέσα στην Animal.Τώρα στην τελευταία γραμμή των αποτελεσμάτων αυτό που παρατηρούμε είναι ότι το αντικείμενο a πήρε τις τιμές των ορισμάτων του αντικείμενου d και αυτό χάρης στο τελεστή < = > και γι' αυτό παίρνουμε τα παραπάνω αποτελέσματα.

Ερώτημα 7:

Κάνουμε τις αλλαγές που μας ζητάει η άσκηση και παίρνουμε τα εξής αποτελέσματα:

```
This is a Animal. It has 1 legs and sound:
This is a Animal. It has 4 legs and sound: bark
This is a Animal. It has 2 legs and sound: chirp
This is a Animal. It has 0 legs and sound: blop

-----
Process exited after 0.04789 seconds with return value 0
Press any key to continue . . .
```

Όπως βλέπουμε πάλι παίρνουμε κάποια αναμενόμενα αλλά και κάποια μη αναμενόμενα αποτελέσματα. Η πρώτη γραμμή των αποτελεσμάτων είναι συνέπεια του ότι έχουμε μια μεταβλητή a προς αντικείμενο τύπου Animal οπού του έχουμε αναθέσει να δείχνει στο αντικείμενο aa τύπου Animal το οποίο από default παίρνει στα τις τιμές που βλέπουμε. Τώρα για τις υπόλοιπες 3 γραμμές ισχύει το ίδιο δηλαδή κάθε φορά ο δείκτης δείχνει απλά σε άλλο αντικείμενο άλλης κλάσης στα οποία έχουμε εμείς αρχικοποιήσει τις τιμές legs και sounds στον κατασκευαστή της κάθε κλάσης. Τώρα ο λόγος που δεν παίρνουμε την αντίστοιχη κλάση για τα αντικείμενα είναι ότι

έχουμε κάνει early binding οπότε ο compiler παίρνει την τιμή της διεύθυνσης που ανήκει στην getClass της κλάσης Animal.

Ερώτημα 8:

Η αλλαγή που πρέπει να γίνει γενικά είναι να υλοποιήσουμε τον μηχανισμό late binding. Αυτό θα το επιτύχουμε δηλώντας την getClass() στην Animal ως virtual(εικονική). Κάνοντας την παραπάνω αλλαγή λύνουμε το πρόβλημα μας και περνούμε τα παρακάτω αποτελέσματα:

```
This is a Animal. It has 4254617 legs and sound:
This is a Dog. It has 4 legs and sound: bark
This is a Bird. It has 2 legs and sound: chirp
This is a Fish. It has 0 legs and sound: blop

-----
Process exited after 0.05029 seconds with return value 0
Press any key to continue . . .
```

Ερώτημα 9:

Κάνουμε τις αλλαγές που μας ζητάει δημιουργώντας έναν πίνακα με δείκτες τύπου Animal στον οποίο αναθέτουμε τα αντικείμενα της κάθε κλάσης. Έπειτα με for διατρέχουμε τον πίνακα η οποία για κάθε στοιχείο του πίνακα καλεί την printInfo().

```
main()
{
    int i;
    Animal aa;
    Dog d;
    Bird b;
    Fish f;
    Animal *a[]={&aa, &d, &b, &f};

    for(i=0; i<4; i++)
        a[i]->printInfo();
}
```

Τα αποτελέσματα:

```
This is a Animal. It has 4254601 legs and sound:
This is a Dog. It has 4 legs and sound: bark
This is a Bird. It has 2 legs and sound: chirp
This is a Fish. It has 0 legs and sound: blop

-----
Process exited after 0.01682 seconds with return value 0
Press any key to continue . . . _
```

Ερώτημα 10:

Χρησιμοποιούμε τα vectors και ο κώδικας που συμπληρώσαμε είναι ο παρακάτω:

```
main()
{
    int i;
    Animal *a1 = new Animal;
    Animal *a2 = new Dog;
    Animal *a3 = new Bird;
    Animal *a4 = new Fish;

    vector <Animal*> vec;
    vec.push_back(a1);
    vec.push_back(a2);
    vec.push_back(a3);
    vec.push_back(a4);

    for(i=0; i<vec.size(); i++)
        vec[i]->printInfo();

    return 0;
}
```

Γενικά τα vectors είναι πιο βολικά από τους πίνακες διότι έχουμε περισσότερες δυνατότητες όπως δεν χρειάζεται να ξέρουμε το μέγεθος τους ενώ για τους πίνακες πρέπει έχουμε άφθονες συναρτήσεις για την προσπέλαση/εισαγωγή/διαγραφή στοιχείων ενός vector ενώ στους πίνακες όχι και επίσης τα vectors μπορούν δεχθούν μια δυναμική χρήση στους κώδικες ενώ οι πίνακες όχι. Βέβαια στην συγκεκριμένη περίπτωση είναι προτιμότεροι οι πίνακες διότι ξέρουμε το μέγεθος του πίνακα και επίσης γλιτώνουμε γραμμές κώδικα.

Άσκηση 2

Ερώτημα 1:

Η λειτουργία του κώδικα αυτή είναι η εξής. Ζητάει από τον χρήστη να του δώσει ένα μοντέλο αυτοκίνητου, τα χιλιόμετρα, και τον χρόνο τον οποίο χρειάζεται να φτάσει αυτά τα χιλιόμετρα. Στην συνέχεια σε ρωτάει αν θέλουμε να βάλουμε και άλλα μοντέλα με τα χαρακτηριστικά τους. Αν δώσουμε την απάντηση γ επαναλαμβάνεται η διαδικασία και μέσα σε ένα if αν το πηλίκο των χιλιομέτρων προς τον χρόνο είναι μεγαλύτερο από το προηγούμενο πηλίκο τότε παίρνουμε ως τιμές αυτές που κάνουν το πηλίκο μεγαλύτερο. Τώρα αν δώσουμε ως απάντηση η τότε βγαίνουμε από τον βρόγχο while και μας τυπώνει τα αποτελέσματα για το ποιο είναι το πιο γρήγορο αμάξι.

Τα αποτελέσματα :

```
Please enter model and plate: Ford Focus
Please enter distance in km: 500
Please enter time spent in hours: 10
More data? (y/n) y
Please enter model and plate: Ford Fiesta
Please enter distance in km: 230
Please enter time spent in hours: 20
More data? (y/n) n
The fastest model is: Ford Focus (speed: 50km/h )

-----
Process exited after 39.19 seconds with return value 0
Press any key to continue . . . █
```

Ερώτημα 2:

Για να αποκτήσει το πρόγραμμα έναν αντικειμενοστραφή ύφος θα δημιουργήσουμε μια κλάση Vehicle που θα παίρνει ως

private τα 3 ορίσματα που το χαρακτηρίζουν(μοντέλο, χιλιόμετρα, ώρες). Στην συνέχεια στο πεδίο public θα έχει την μέθοδο input η οποία θα είναι αυτή στην οποία θα δίνω τιμές για τα ορίσματα. Έπειτα θα έχουμε μια bool τύπου μέθοδο που θα συγκρίνει 2 αντικείμενα της κλάσης και ικανοποιείται η συνθήκη της if τότε θα μας επιστρέφει true αλλιώς false. Τέλος έχουμε την μέθοδο show που μας τυπώνει τα αποτελέσματα για το πιο γρήγορο αμάξι.

Τα αποτελέσματα:

```
Please enter model and plate: Pagandi Zonda
Please enter distance in km: 300
Please enter time spent in hours: 1
Please enter model and plate: Bugati Veyron
Please enter distance in km: 360
Please enter time spent in hours: 1
More data? (y/n) y
Please enter model and plate: Lamborghini Aventador
Please enter distance in km: 330
Please enter time spent in hours: 1
More data? (y/n) n
The fastest vehicle is: Bugati Veyron (speed: 360km/h )

Process returned 0 (0x0)    execution time : 81.360 s
Press any key to continue.
```

Ερώτημα 3:

Δηλώνοντας ως friend μια μέθοδο αυτό που καταφέρνουμε είναι να έχουμε μια ειδική επιχορήγηση για πρόσβαση σε ιδιωτικά και προστατευόμενα μέλη δηλαδή να έχουμε μια μέθοδο άλλης τάξης ή μια global function. Τα αποτελέσματα είναι ιδιά με τα παραπάνω.

Ερώτημα 4:

Οι αλλαγές του κώδικα θα παρατεθούν στο παράρτημα στο τέλος της αναφοράς. Τα αποτελέσματα είναι ίδια με αυτά του 2^{ου} ερωτήματος.