ΜΑΘΗΜΑ 17

ΚΛΑΣΕΙΣ ΚΑΙ ΔΕΙΚΤΕΣ

Γιώργος Διάκος - Full Stack Developer

3. Κλάσεις που περιέχουν δείκτες

1. Παράδειγμα κλάσης που περιέχει δείκτη

- Ένας δείκτης είναι απλά μία μεταβλητή
  - Συνεπώς μπορεί να είναι μέλος σε κλάση
- Βλέπουμε ένα απλό παράδειγμα:
  - Στο οποίο δεσμεύουμε δυναμικά στον κατασκευαστή το χώρο για έναν ακέραιο
  - και τον αποδεσμεύουμε στον καταστροφέα.

```
/* CPP3.class_with_pointer */
#include <iostream>
using namespace std;

class dummy {
  public:
    dummy();
    ~dummy();
    void set_val(int in_val);
    int get_val();
  private:
    int *p_val;
};
```

3. Κλάσεις που περιέχουν δείκτες

1. Παράδειγμα κλάσης που περιέχει δείκτη

```
int main()
{
   dummy ob;

   ob.set_val(3);

   cout<<endl<<ob.get_val()<<endl;

   return 0;
}
dummy::dummy()
{
   p_val = new int;
   if (!p_val)  cout<<"Error allocating memory";

   cout<<"Constructing...";
}
```

```
dummy::~dummy()
{
   delete p_val;
   cout<<"Destructing...";
}

void dummy::set_val(int in_val)
{
   *p_val = in_val;
}

int dummy::get_val()
{
   return *p_val;
}
```

3. Κλάσεις που περιέχουν δείκτες

2. …και ένα πρόβλημα (χωρίς λύση για την ώρα)

- Ας δούμε και ένα πρόβλημα που μπορεί να προκύψει όταν μία κλάση περιέχει δείκτες που κάνουν δυναμική διαχείριση μνήμης
- Το πρόβλημα αυτό προκύπτει όταν:
  - Έχουμε μία κλάση που κάνει δυναμική δέσμευση μνήμης
  - Κάνουμε αντιγραφή του αντικειμένου (π.χ. ob1 = ob2)
- Μεταγλωττίστε και εκτελέστε το ακόλουθο πρόγραμμα:

```
/* CPP3.assignment_problem Δυναμική Δέσμευση και Τελεστής Ανάθεσης */
#include <iostream>
using namespace std;

class dummy {
  public:
    dummy();
    ~dummy();
    void set_val(int in_val);
    int get_val();
  private:
    int *p_val;
};
```

3. Κλάσεις που περιέχουν δείκτες

2. …και ένα πρόβλημα (χωρίς λύση για την ώρα)

```
int main()
{
   dummy ob1;
   ob1.set_val(3);
   dummy ob2;
   ob2 = ob1;

   cout<<ob1.get_val()<<endl;
   cout<<ob2.get_val()<<endl;

   return 0;
}
dummy::dummy()
{
   p_val = new int;
   if (!p_val)  cout<<"Error allocating memory";

   cout<<"Constructing...";
}
```

```
dummy::~dummy()
{
   delete p_val;
   cout<<"Destructing...";
}

void dummy::set_val(int in_val)
{
   *p_val = in_val;
}

int dummy::get_val()
{
   return *p_val;
}
```

- Σημαντικό! Στη C++ οι μεταβλητές μπορούν να δηλωθούν σε οποιοδήποτε μέρος του προγράμματος (αλλά θα το αποφεύγουμε…)

3. Κλάσεις που περιέχουν δείκτες

2. …και ένα πρόβλημα (χωρίς λύση για την ώρα)

- Η εκτέλεση του προγράμματος οδηγεί σε σφάλμα κατά το χρόνο εκτέλεσης:

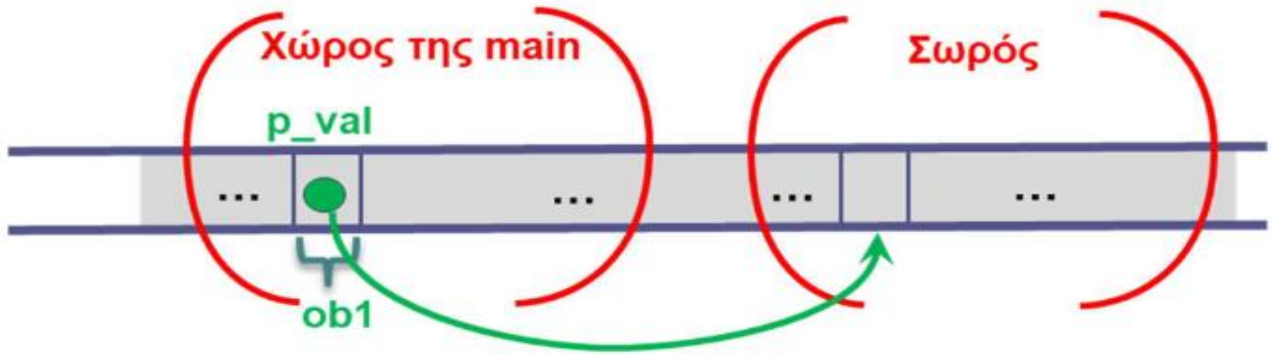
```
Constructing...
Constructing...
3
3
Destructing...
*** Error in `./a.out': double free or corruption (fasttop): 0x0000000001edbc20 ***
Aborted (core dumped)
```

- Για να καταλάβουμε γιατί συμβαίνει αυτό, θα δούμε την κατάσταση της μνήμης βήμα – βήμα κατά την εκτέλεση της main

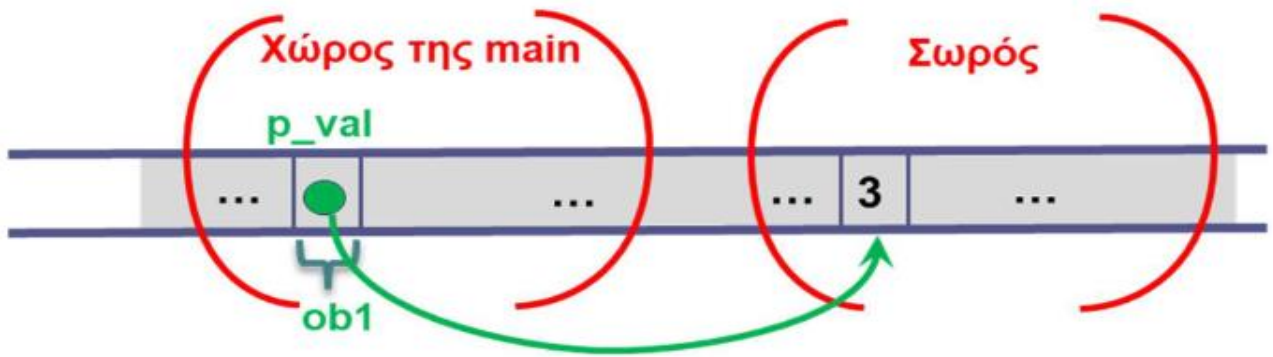
3. Κλάσεις που περιέχουν δείκτες

2. …και ένα πρόβλημα (χωρίς λύση για την ώρα)

- dummy ob1; // Δεσμεύει το χώρο για το ob1.



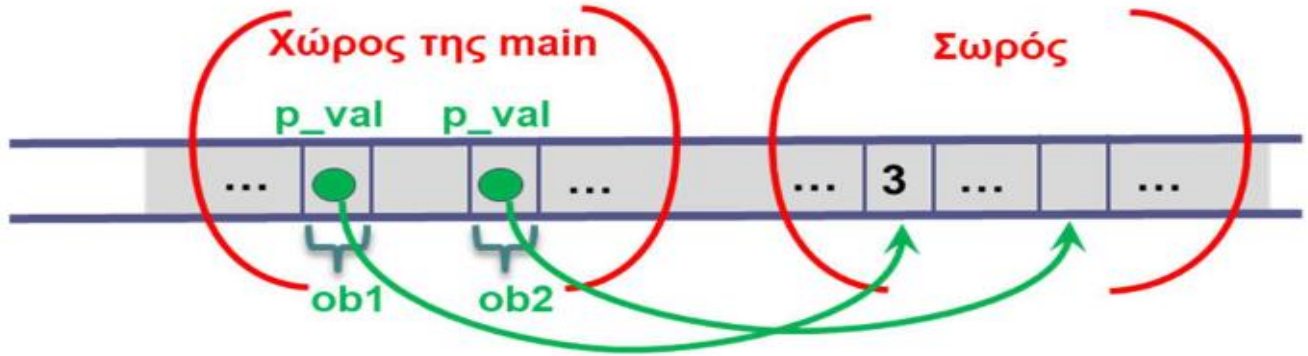
- ob1.set_val(3); // αποθηκεύει την τιμή 3 στο χώρο μνήμης



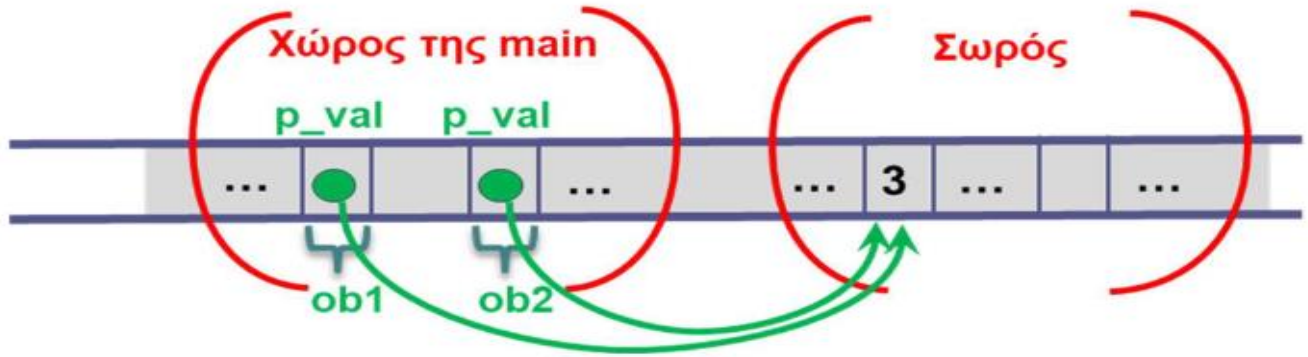
3. Κλάσεις που περιέχουν δείκτες

2. …και ένα πρόβλημα (χωρίς λύση για την ώρα)

- dummy ob2; // Δεσμεύει το χώρο για το ob2.



- ob2=ob1 // Δεν κάνει ακριβώς αυτό που θέλουμε…



3. Κλάσεις που περιέχουν δείκτες

2. …και ένα πρόβλημα (χωρίς λύση για την ώρα)

- Τώρα με την ολοκλήρωση του προγράμματος, θα τρέξουν οι δύο destructors.
  - Συνεπώς ο **ίδιος χώρος μνήμης θα αποδεσμευτεί δύο φορές**.
  - Και αυτό οδηγεί σε σφάλμα στο χρόνο εκτέλεσης.
- Ενώ ισχύει ότι έχουμε κάνει και το βασικό λάθος
  - Με την ολοκλήρωση του προγράμματος, έχει ξεμείνει χώρος μνήμης, τον οποίο δεσμεύσαμε δυναμικά και δεν τον αποδεσμεύσαμε ποτέ (**memory leak**)
- Το πρόβλημα αυτό προκύπτει όταν γίνεται bit by bit αντιγραφή αντικειμένων (shallow copy)
  - όπως στην περίπτωση που είδαμε,
  - αλλά και σε άλλες περιπτώσεις (π.χ. όταν περνάμε ένα αντικείμενο μέσω τιμής σε μία συνάρτηση)

- Θα αντιμετωπίσουμε το πρόβλημα αυτό, στα επόμενα μαθήματα:
  - Θα μάθουμε τις αναφορές (references)
    - και μέσω αυτών θα ορίσουμε τον κατασκευαστή αντιγράφου (copy constructor)
  - επίσης θα μάθουμε την υπερφόρτωση του =
    - και θα ορίσουμε τι πρέπει να γίνεται όταν έχουμε ανάθεση αντικειμένου

4. Δυναμική Δέσμευση Μνήμης για Πίνακες

1. Μονοδιάστατοι Πίνακες

- Οι τελεστές new και delete χρησιμοποιούνται και για την δέσμευση μνήμης για πίνακες
- Δέσμευση Μνήμης:
  - Η σύνταξη του τελεστή new είναι:

```
ptr = new type [n];
```

  - Δεσμεύει χώρο μνήμης για n αντικείμενα τύπου type και επιστρέφει έναν δείκτη σε αυτόν το χώρο
  - Το n δεν χρειάζεται να είναι σταθερά.

- Αποδέσμευση Μνήμης:
  - Η σύνταξη του τελεστή delete είναι:

```
delete [] ptr;
```

4. Δυναμική Δέσμευση Μνήμης για Πίνακες

2. Παράδειγμα δέσμευσης μνήμης για μονοδιάστατο πίνακα

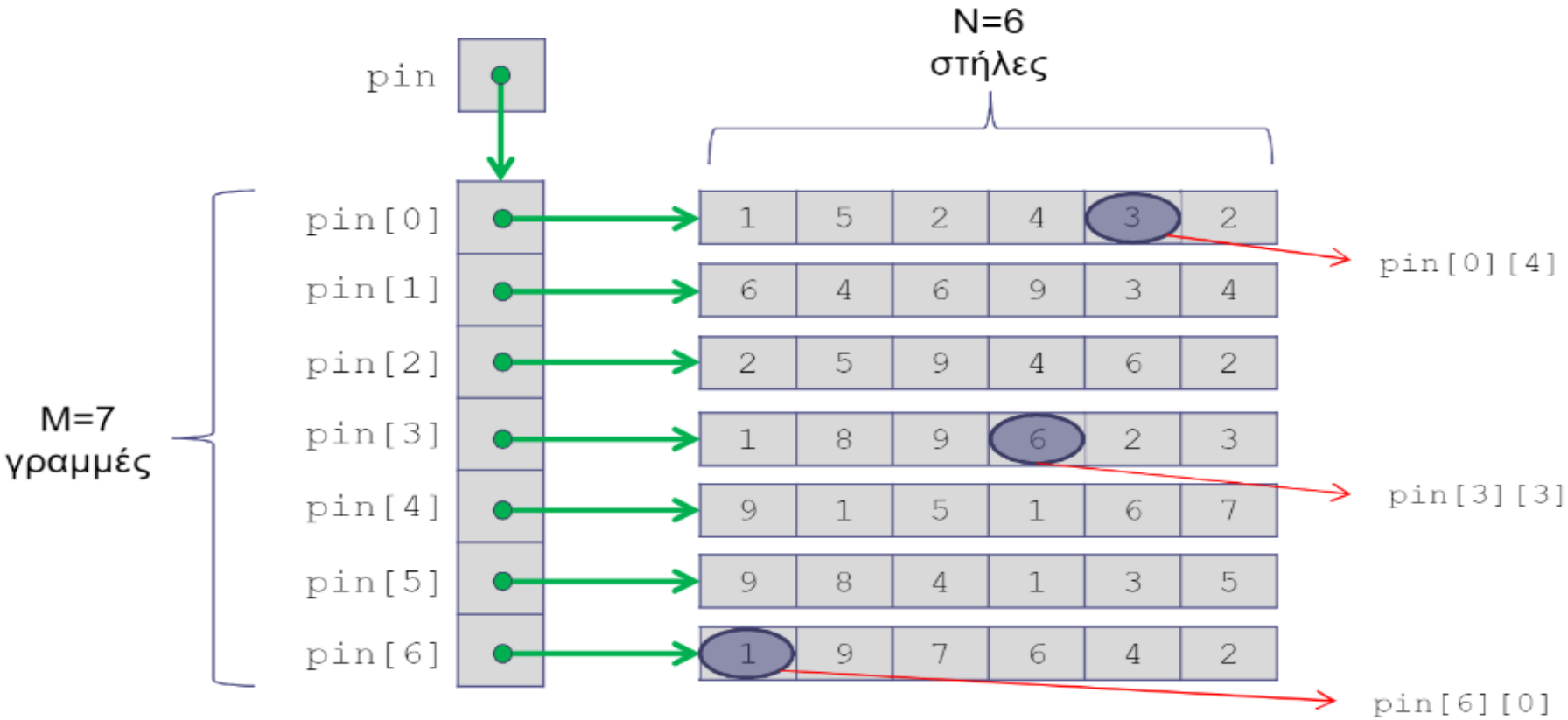
- Βλέπουμε και ένα παράδειγμα δέσμευσης μνήμης για έναν πίνακα n ακεραίων.

```
/* CPP3.1d_dynamic_array Μονοδιάστατος
πίνακας με δυναμική δέσμευση μνήμης */
#include <iostream>
using namespace std;

int main()
{
   int *arr;
   int n=4;

   /* Δέσμευση Μνήμης */
   arr = new int [n];
   if (!arr) cout<<"Error allocating memory!";

   /* Κάποια δουλειά στον πίνακα */
   for (int i=0; i<n; i++)
      arr[i]=i*i;

   for (int i=0; i<n; i++)
      cout<<arr[i]<<" ";

   /* Αποδέσμευση Μνήμης */
   delete [] arr;

   return 0;
}
```

4. Δυναμική Δέσμευση Μνήμης για Πίνακες

3. Διδιάστατοι Πίνακες

- Ένας διδιάστατος MxN πίνακας π.χ. ακεραίων είναι:
  - ένας πίνακας Μ δεικτών σε ακέραιο
    - όπου κάθε δείκτης είναι ένας πίνακας από Ν ακεραίους
- Σχηματικά (π.χ. για M=7, N=6):



4. Δυναμική Δέσμευση Μνήμης για Πίνακες

3. Διδιάστατοι Πίνακες

- Από τα παραπάνω:
  - Η δέσμευση μνήμης θα γίνει με τις εντολές:

```
ptr = new type * [M];
for (i=0; i<M; i++)
   ptr[i] = new type [N];
```

    - Προσοχή, ότι το ptr είναι διπλός δείκτης
      - αφου θέλουμε να δείχνει στην διεύθυνση μιας μεταβλητής που είναι διεύθυνση ενός τύπου.
      - Συνεπώς θα δηλωθεί ως:

```
type **ptr;
```

  - Η αποδέσμευση θα γίνει εντολές:

```
for (i=0; i<M; i++)
   delete [] ptr[i];
delete [] ptr;
```

4. Δυναμική Δέσμευση Μνήμης για Πίνακες

4. Παράδειγμα δέσμευσης μνήμης για διδιάστατο πίνακα

- Βλέπουμε και ένα παράδειγμα δέσμευσης μνήμης για έναν πίνακα n ακεραίων.

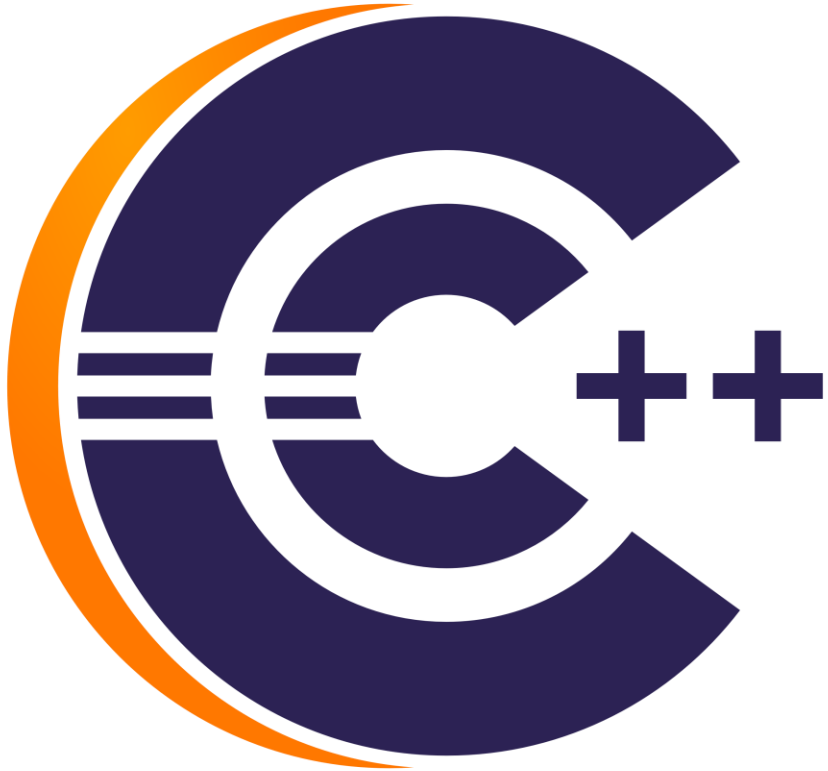
```
/* CPP3.2d_dynamic_array Διδιάστατος πίνακας με
δυναμική δέσμευση μνήμης */
#include <iostream>
using namespace std;

int main()
{
   int **arr;
   int i,j, n=3, m=5;

   /* Δέσμευση Μνήμης */
   arr = new int * [n];
   if (!arr) cout<<"Error allocating memory!";
   for (i=0; i<n; i++)
   {
      arr[i] = new int [m];
      if (!arr[i]) cout<<"Error allocating memory!";
   }
```

```
   /* Κάποια δουλειά στον πίνακα */
   for (i=0; i<n; i++)
      for (j=0; j<m; j++)
         arr[i][j]=i*j;

   for (i=0; i<n; i++)
   {
      for (j=0; j<m; j++)
         cout<<arr[i][j];
      cout<<endl;
   }

   /* Αποδέσμευση Μνήμης */
   for (i=0; i<n; i++)
      delete [] arr[i];
   delete [] arr;

   return 0;
}
```

ΑΣΚΗΣΗ 1: ΔΙΠΛΟΙ ΔΕΙΚΤΕΣ

Για να πειραματιστούμε με τους δείκτες

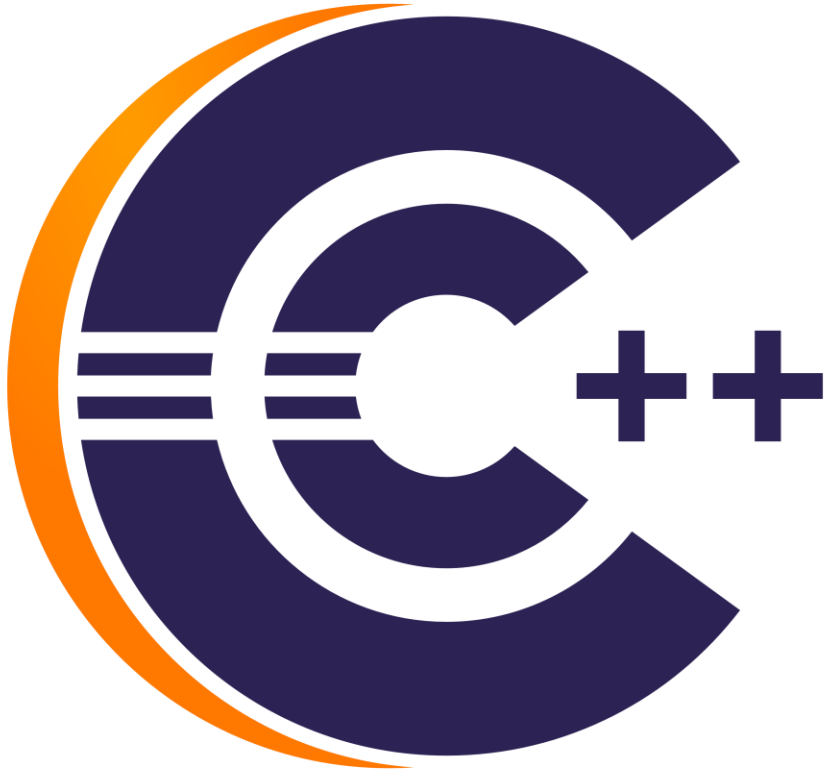
- Δήλωσε μια ακέραια μεταβλητή χ
- Τύπωσε την τιμή και την διεύθυνση της
- Δήλωσε εναν δείκτη σε ακέραιο p
- Θέσε τον να δείχνει στην χ
- Τύπωσε την τιμή και την διεύθυνση του
- Τύπωσε την τιμή του χ , μέσω του p
- Δήλωσε εναν δείκτη σε δείκτη σε ακέραιο , με όνομα pp
- Θέσε τον να δείχνει στον p
- Τύπωσε την τιμή και την διεύθυνση του
- Τύπωσε την τιμή του χ , μέσω του pp
- Τύπωσε την τιμή του p , μέσω του pp



ΑΣΚΗΣΗ 2: ΚΛΑΣΗ ΔΥΝΑΜΙΚΟΣ ΠΙΝΑΚΑΣ

Κατασκεύασε μια κλάση (ARRAY) που να περιτυλλίσει την έννοια του μονοοδιάστατου πίνακα ως εξής :

- Να έχει ως μέλη εναν δυναμικό πίνακα (δείκτης) ακεραίων , καθώς και την διάσταση του πίνακα
- Ο κατασκευαστής να παίρνει ως όορισμα την διάσταση του πίνακα και να δεσμεύει δυναμικά τον χώρο μνήμης που απαιτείται.
- Να έχει accessors για ενα στοιχείο του πίνακα
- Να τυπώνουν μήνυμα λάθους , σε περίπτωση πρόσβασης εκτός των ορίων του πίνακα
- Μια μέθοδο print που να τυπώσνει τα στοιχεία του πίνακα
- Η συνάρτηση main
- Να κατασκευάζει έναν πίνακα της κλάσης με 10 θέσεις
- Να αρχικοποιεί τα στοχεία του πίνακα , ώστε κάθε στοιχείο να έχει το τετράγωνο της αντίστοιχης θέσης
- Να τυπώνει τον πίνακα
- Ποιο πρόβλημα υπάρχει με την υλοποίηση; Πότε περιμένουμε ότι το πρόγραμμα αυτό θα έχει πρόβλημα και θα <<σκάσει>> κατά τον χρόνο εκτέλεσης;



ΤΕΛΟΣ ΜΑΘΗΜΑΤΟΣ

ΚΛΑΣΕΙΣ ΚΑΙ ΔΕΙΚΤΕΣ

Γιώργος Διάκος - Full Stack Developer