ΤΕΧΝΙΚΕΣ ΑΝΤΙΚΕΙΜΕΝΟΣΤΡΑΦΟΥΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ

Αντικείμενα με πίνακες.

Constructors.

Υλοποίηση Στοίβας

Στην άσκηση αυτή θα υλοποιήσετε μια κλάση **Geometric** η οποία διαχειρίζεται μια γεωμετρική ακολουθία ακεραίων η οποία μπορεί να έχει οποιοδήποτε μέγεθος. Μια γεωμετρική ακολουθία μήκους n με βάση b, είναι οι αριθμοί $1, b, b^2, \dots, b^{n-1}$. Δηλαδή, η i τιμή της ακολουθίας έχει τιμή b^{i-1}. Η κλάση σας θα πρέπει να αποθηκεύει τις τιμές της ακολουθίας σε ένα πίνακα, και να υποστηρίζει τις εξής λειτουργίες:

1. Ένα **constructor,** ο οποίος δεν παίρνει ορίσματα και δημιουργεί μια ακολουθία με βάση το 2 και μήκος 10.
2. Ένα **constructor,** ο οποίος θα παίρνει σαν όρισμα τη βάση της ακολουθίας και το μήκος της.
3. Μια μέθοδο **print**, η οποία θα τυπώνει τις τιμές της ακολουθίας. Π.χ., για την default ακολουθία θα τυπώνει «0 1 2 4 8 16 32 64 128 256 512».
4. Την μέθοδο **equals**, η οποία θα συγκρίνει αν δύο ακολουθίες είναι ίδιες.
5. Μια μέθοδο **multiplyWith** η οποία παίρνει σαν όρισμα μία άλλη ακολουθία (ένα αντικείμενο τύπου **Geometric**) και, εφόσον έχουν το ίδιο μήκος, την πολλαπλασιάζει με την ακολουθία που καλεί την μέθοδο (πολλαπλασιάζει τις τιμές ανά όρο) και αποθηκεύει το αποτέλεσμα στο αντικείμενο που κάλεσε την μέθοδο.

Σας δίνεται η κλάση **GeometricTest,** για να τεστάρετε την κλάση σας. Πρέπει να συμπληρώσετε τα κομμάτια που λείπουν για τον έλεγχο της ισότητας. Όταν υλοποιήσετε τις μεθόδους που καλούνται στην main, βγάλετε τα σχόλια από τις αντίστοιχες εντολές για να τεστάρετε τις μεθόδους.

Μαθήματα από το lab

- Τι πληροφορία (δεδομένα) θέλουμε να κρατάει η κλάση μας?
  - Οπωσδήποτε ένα πίνακα με τις τιμές της προόδου.
  - Το μήκος της προόδου
  - Τη βάση της προόδου

- Η πληροφορία (τα δεδομένα) που θέλουμε να κρατάει η κλάση θα είναι τα πεδία της κλάσης
  - Ένα πίνακα ακεραίων sequence με τις τιμές της ακολουθίας
  - Έναν ακέραιο length με το μήκος της ακολουθίας που θα είναι και το μήκος του πίνακα
  - Ένα ακέραιο base με την βάση της ακολουθίας

```
class Geometric
{
  public Geometric(int base, int length)
  {
        int sequence[] = new int[length];
        sequence[0] = 1;
        for (int i=1; i < length; i++){
          sequence[i] = sequence[i-1]*base;
        }
  }

  public void print()
  {
        for (int i = 0; i < length; i ++){
          System.out.println(sequence[i] + " ");
        }
        System.out.println();
  }
}

class GeometricTest
{
  public static void main(String[] args){
        Geometric geom = new Geometric(3,6);
        geom.print();
  }
}
```

Σωστό ή λάθος?

Οι μεταβλητές length και sequence δεν είναι ορισμένες.

Για να μπορεί να τις βλέπει η μέθοδος print (ή οποιαδήποτε άλλη μέθοδος) θα πρέπει να είναι ορισμένες ως πεδία της κλάσης

ΛΑΘΟΣ!

```
class Geometric
{
  private int base;
  private int length;
  private int sequence[];

  public Geometric(int base, int length)
  {
        int sequence[] = new int[length];
        sequence[0] = 1;
        for (int i=1; i < length; i++){
          sequence[i] = sequence[i-1]*base;
        }
  }

  public void print()
  {
        for (int i = 0; i < length; i ++){
          System.out.println(sequence[i] + " ");
        }
        System.out.println();
  }
}

class GeometricTest
{
  public static void main(String[] args){
        Geometric geom = new Geometric(3,6);
        geom.print();
  }
}
```

Σωστό?

Ο constructor δεν αρχικοποιεί τα πεδία της κλάσης .

Οι μεταβλητές length και sequence που ορίζονται μέσα στον constructor είναι τοπικές μεταβλητές και δεν αλλάζουν την τιμή των πεδίων.

ΛΑΘΟΣ!

```
class Geometric
{
  private int base;
  private int length;
  private int sequence[];

  public Geometric(int base, int length)
  {
          this.base = base;
          this.length = length;
          sequence[0] = 1;
          for (int i=1; i < length; i++){
            sequence[i] = sequence[i-1]*base;
          }
  }

  public void print()
  {
          for (int i = 0; i < length; i ++){
            System.out.println(sequence[i] + " ");
          }
          System.out.println();
  }
}

class GeometricTest
{
  public static void main(String[] args){
          Geometric geom = new Geometric(3,6);
          geom.print();
  }
}
```

Σωστό?

Οι base και length αρχικοποιούνται σωστά.

Ο πίνακας sequence όμως όχι.

Τον έχουμε ορίσει σωστά αλλά δεν του έχουμε δώσει χώρο! Δεν έχουμε προσδιορίσει το μέγεθος του

ΛΑΘΟΣ!

```
class Geometric
{
   private int base;
   private int length;
   private int sequence[] = new int[length];

   public Geometric(int base, int length)
   {
           this.base = base;
           this.length = length;
           sequence[0] = 1;
           for (int i=1; i < length; i++){
             sequence[i] = sequence[i-1]*base;
           }
   }

   public void print()
   {
           for (int i = 0; i < length; i ++){
             System.out.println(sequence[i] + " ");
           }
           System.out.println();
   }
}

class GeometricTest
{
   public static void main(String[] args){
           Geometric geom = new Geometric(3,6);
           geom.print();
   }
}
```

Σωστό?

Θυμηθείτε ότι οι εντολές αυτές θα εκτελεστούν πριν από τις εντολές του constructor. Εκείνη τη στιγμή δεν ξέρουμε τη το μήκος της ακολουθίας και άρα δημιουργούμε ένα πίνακα μηδενικού μεγέθους!

ΛΑΘΟΣ!

```
class Geometric
{
  private int base;
  private int length;
  private int sequence[];

  public Geometric(int base, int length)
  {
          sequence = new int[length];
          sequence[0] = 1;
          for (int i=1; i < length; i++){
            sequence[i] = sequence[i-1]*base;
          }
  }

  public void print()
  {
          for (int i = 0; i < length; i ++){
            System.out.println(sequence[i] + " ");
          }
          System.out.println();
  }
}

class GeometricTest
{
  public static void main(String[] args){
          Geometric geom = new Geometric(3,6);
          geom.print();
  }
}
```

Σωστό?

Ο Constructor θα αρχικοποιήσει σωστά τον πίνακα sequence, αλλά δεν θα αλλάξει το πεδίο length μιας και χρησιμοποιεί την τοπική μεταβλητή

Το length εδώ αναφέρεται στο πεδίο και έχει τιμή μηδέν.

ΛΑΘΟΣ!

```
class Geometric
{
   private int base;
   private int length;
   private int sequence[];

   public Geometric(int base, int length)
   {
           this.base = base;
           this.length = length;
           sequence = new int[length];
           sequence[0] = 1;
           for (int i=1; i < length; i++){
             sequence[i] = sequence[i-1]*base;
           }
   }

   public void print()
   {
           for (int i = 0; i < length; i ++){
             System.out.println(sequence[i] + " ");
           }
           System.out.println();
   }
}

class GeometricTest
{
   public static void main(String[] args){
           Geometric geom = new Geometric(3,6);
           geom.print();
   }
}
```

Σωστό?

Πρώτα δηλώνουμε τα πεδία μέσα στην κλάση

Μετά δίνουμε τιμή στο μήκος και την βάση και αφού πλέον ξέρουμε το μήκος δίνουμε χώρο στον πίνακα που θα κρατάει τις τιμές.

Τώρα μπορούμε και να κάνουμε και την αρχικοποίηση του πίνακα

ΣΩΣΤΟ!

```
class Geometric
{
  private int base;
  private int length;
  private int sequence[];

  public Geometric(int base, int length){
          this.base = base;
          this.length = length;
          sequence = new int[length]
          sequence[0] = 1;
          for (int i=1; i < length; i++){
            sequence[i] = sequence[i-1]*base;
          }
  }

  public Geometric(){
          base = 2;
          length = 10;
          sequence = new int[length];
          sequence[0] = 1;
          for (int i=1; i < length; i++){
            sequence[i] = sequence[i-1]*base;
          }
  }
}

class GeometricTest{
  public static void main(String[] args){
          Geometric geom = new Geometric();
          geom.print();
  }
}
```

Default constructor και η κλήση του

```
class Geometric
{
   private int base = 2;
   private int length = 10;
   private int sequence[] = new int[length];

   public Geometric(int base, int length)
   {
          this.base = base;
          this.length = length;
          sequence = new int[length]
          sequence[0] = 1;
          for (int i=1; i < length; i++){
            sequence[i] = sequence[i-1]*base;
          }
   }

   public Geometric(){
          sequence[0] = 1;
          for (int i=1; i < length; i++){
            sequence[i] = sequence[i-1]*base;
          }
   }
}

class GeometricTest
{
   public static void main(String[] args){
          Geometric geom = new Geometric();
          geom.print();
   }
}
```

Η αρχικοποίηση των πεδίων θα γίνει στις default τιμές

Ο constructor με όρισμα θα ξανα-ορίσει ταο μήκος και την βάση και θα δώσει νέο χώρο για τον πίνακα

Ο default constructor με όρισμα θα κρατήσει τις default τιμές

Όχι και τόσο καλή υλοποίηση

```
class Geometric
{
  private int base;
  private int length;
  private int sequence[];

  public Geometric(int base, int length)
  {
        this.base = base;
        this.length = length;
        sequence = new int[length];
        createSequence()
  }

  public Geometric(){
       base = 2;
        length = 10;
        sequence = new int[length];
        createSequence();
  }

  private void createSequence(){
        sequence[0] = 1;
        for (int i=1; i < length; i++){
          sequence[i] = sequence[i-1]*base;
        }
  }
}
```

Η διαδικασία της δημιουργίας της ακολουθίας επαναλαμβάνεται σε δύο μέρη. Μπορούμε λοιπόν να ορίσουμε μία βοηθητική μέθοδο που θα την υλοποιεί και θα την καλούμε στον constructor

Πλεονεκτήμα:

- Κάνει τον κώδικα πιο απλό και κατανοητό

Εμβέλεια μεταβλητών

- Η κάθε μεταβλητή έχει εμβέλεια μέσα στο block στο οποίο ορίζεται.
  - Τις μεταβλητές-πεδία της κλάσης μπορούν να τις χρησιμοποιήσουν όλες οι μέθοδοι της κλάσης
    - Οι μεταβλητές έχουν ζωή όσο υπάρχει το αντίστοιχο αντικείμενο της κλάσης
  - Οι μεταβλητές που ορίζονται μέσα σε μία μέθοδο μπορούν να χρησιμοποιηθούν μόνο μέσα στη μέθοδο.
    - Οι μεταβλητές χάνονται όταν βγούμε από τη μέθοδο.
  - Οι παράμετροι μιας μεθόδου είναι σαν τοπικές μεταβλητές της μεθόδου.

Παράδειγμα

```
public Geometric(int base, int length)
{
      this.base = base;
      this.length = length;
      sequence = new int[length];
      sequence[0] = 1;
      for (int i=1; i < length; i++){
        sequence[i] = sequence[i-1]*base;
      }
}
```

Οι κόκκινες μεταβλητές υπάρχουν μόνο μέσα στο μπλοκ της μεθόδου

Οι μπλε μεταβλητές είναι πεδία

```
public Geometric(int base, int length)
{
      this.base = base;
      this.length = length;
      sequence = new int[length];
      sequence[0] = 1;
      for (int i=1; i < length; i++){
        sequence[i] = sequence[i-1]*base;
      }
}
```

Οι παράμετροι είναι σαν τοπικές μεταβλητές

```
public Geometric(<όρισμα1>, <όρισμα2>)
{
      base = όρισμα1;
      length = όρισμα2;
      this.base = base;
      this.length = length;
      sequence = new int[length];
      sequence[0] = 1;
      for (int i=1; i < length; i++){
        sequence[i] = sequence[i-1]*base;
      }
}
```

Η μέθοδος multiplyWith

Η μέθοδος δεν επιστρέφει κάτι μιας και το αποτέλεσμα της πρόσθεσης θα αποθηκευτεί στο αντικείμενο

Η μέθοδος παίρνει σαν όρισμα ένα αντικείμενο Geometric με το οποίο θα πολλαπλασιαστεί

```
public void multiplyWith(Geometric other)
{
      if (this.length != other.length){
            return;
      }
      for (int i=0; i < length; i++){
        this.sequence[i] *= other.sequence[i];
      }
      this.base *= other.base;
}
```

Πρέπει να ενημερώσουμε και την βάση μιας και πλέον έχει αλλάξει

Έχουμε πρόσβαση στα πεδία του other γιατί είναι της ίδιας κλάσης με το αντικείμενο που καλεί την multiplyWith

Η μέθοδος equals

Η μέθοδος equals ορίζεται πάντα έτσι

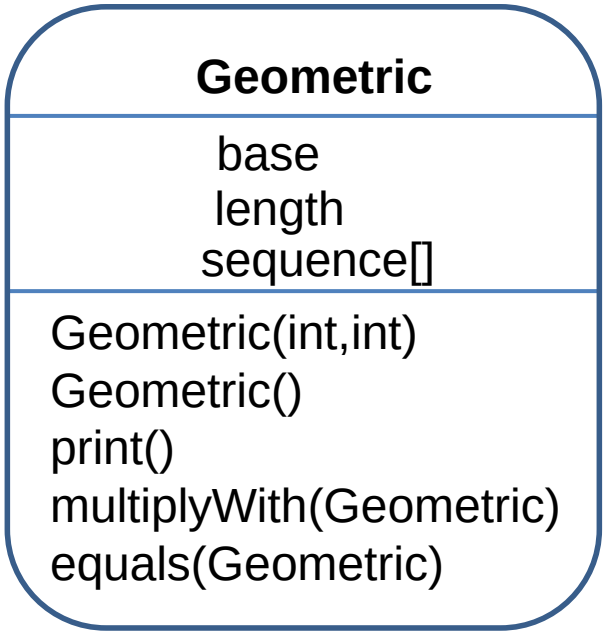
```
public boolean equals(Geometric other)
{
     if (this.length == other.length &&
                    this.base == other.base){
          return true;
     }
     return false;
 }
```

Αυτός είναι ο πιο απλός τρόπος να ελέγξετε για ισότητα. Πρέπει όμως να είσαστε προσεκτικοί να ενημερώνεται σωστά η βάση.

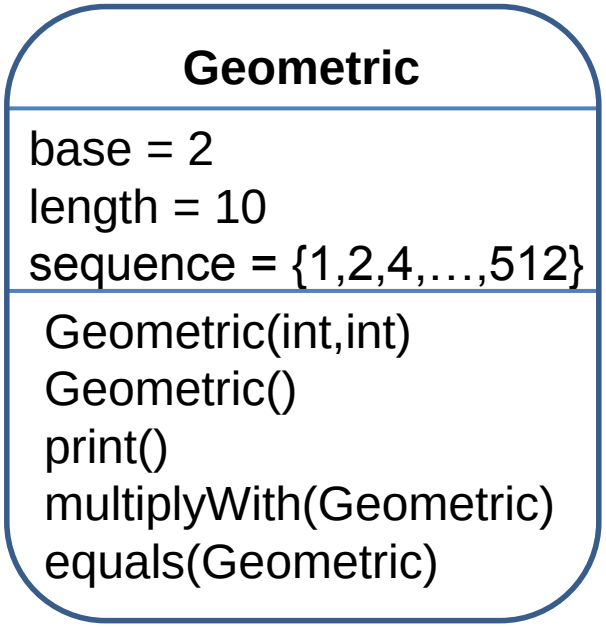
Μία άλλη επιλογή είναι να ελέγξετε τις τιμές του πίνακα μία μία ώστε να συμφωνούν.

Κλάσεις και αντικείμενα

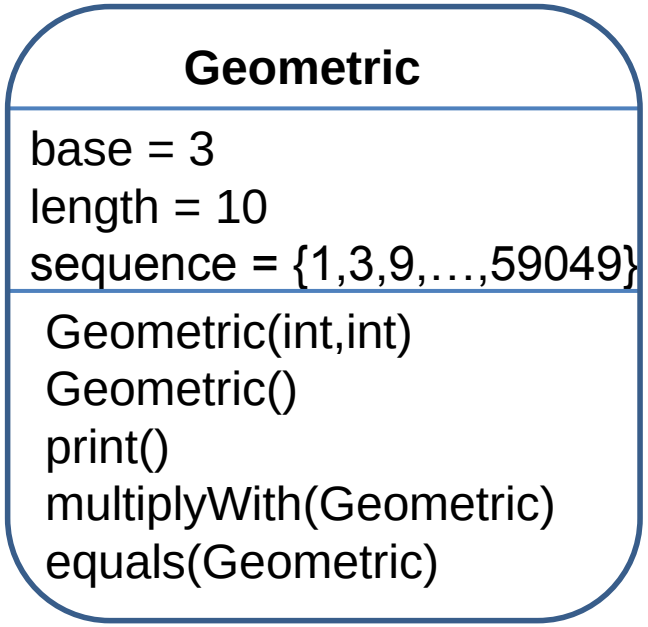
Ορισμός της κλάσης



firstSequence = new Geometric()

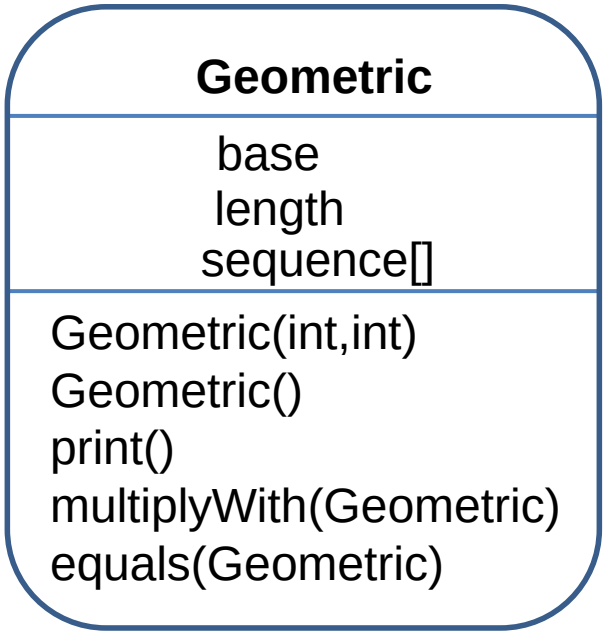


secondSequence = new Geometric(3,10)



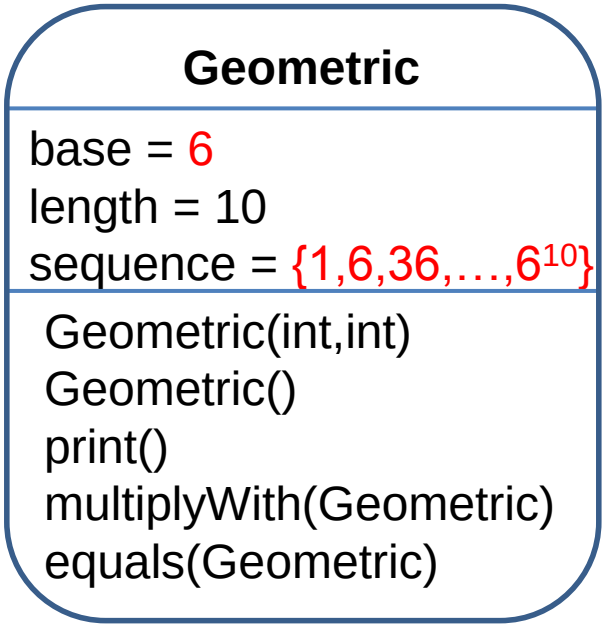
Κλάσεις και αντικείμενα

Ορισμός της κλάσης

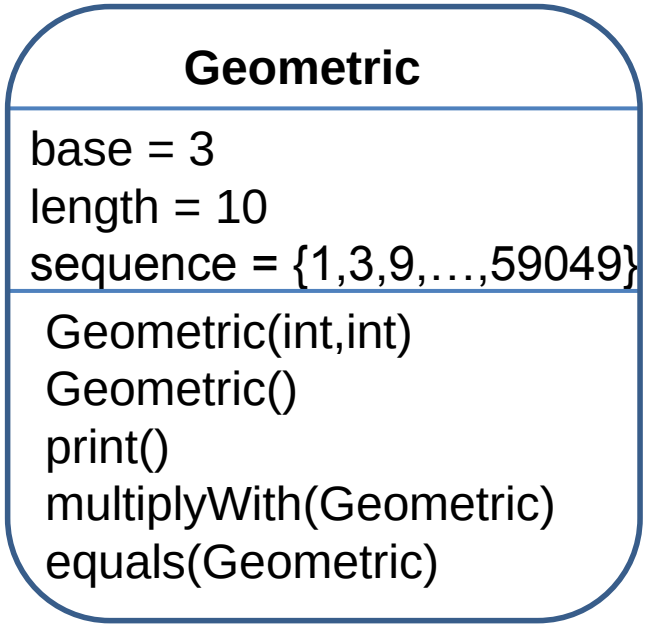



```
firstSequence.multiplyWith(secondSequence);
```

firstSequence = new Geometric()



secondSequence = new Geometric(3,10)



Η εντολή exit

Χρησιμοποιείται για σοβαρά λάθη για να σταματάει την εκτέλεση του προγράμματος.

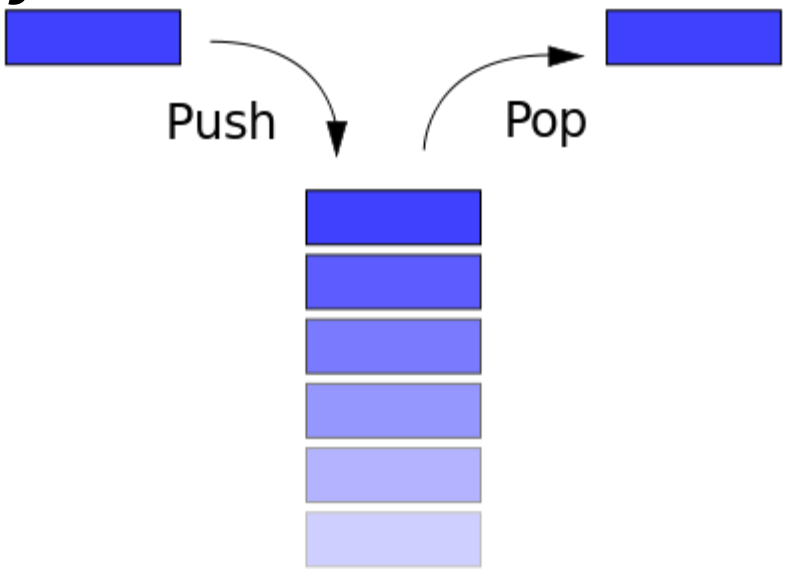
```
public Geometric(int base, int length)
{
      this.base = base;
      this.length = length;
      if (length < 0){
            System.exit(-1);
      }
      sequence = new int[length];
      createSequence()
}
```

Αν δώσουμε αρνητικό μήκος ακολουθίας το πρόγραμμα μας θα σταματήσει.

Το -1 εξυπηρετεί σαν κωδικός λάθους, μπορείτε να βάλετε όποια τιμή θέλετε.

Παράδειγμα ADT: Στοίβα (Stack)

- Η Στοίβα είναι μια συλλογή δεδομένων η οποία επιτρέπει τις εξής λειτουργίες:
  - push(element): προσθέτει ένα νέο στοιχείο στην κορυφή της στοίβας
  - pop(): αφαιρεί και επιστρέφει το στοιχείο το οποίο βρίσκεται στην κορυφή της στοίβας.
  - isEmpty(): ελέγχει αν η στοίβα είναι άδεια και επιστρέφει true ή false
- Η Στοίβα υλοποιεί την πολιτική Last-In-First-Out (LIFO) στη σειρά που μας δίνει τα στοιχεία
  - Χρήσιμο σε διάφορες εφαρμογές, π.χ., για τη δέσμευση μνήμης στην κλήση συναρτήσεων



Υλοποίηση

- Θα υλοποιήσουμε μια Στοίβα ακεραίων χρησιμοποιώντας ένα πίνακα (Στοιβα συγκεκριμένης χωρητικότητας)
  - Τι πεδία πρέπει να ορίσουμε?
  - Τι μεθόδους?

```
class Stack
{
        private int capacity;
        private int size = 0;
        private int[] elements;

        public Stack(int capacity){
                this.capacity = capacity;
                elements = new int[capacity];
        }

        public void push(int element){
                if (size == capacity){
                        System.out.println("Cannot enter any more elements");
                        return;
                }
                elements[size] = element;
                size ++;
        }

        public int pop(){
                if (size == 0){
                        System.out.println("No elements to pop");
                        return -1;
                }
                size -- ;
                return elements[size];
        }

        public boolean isEmpty(){
                return (size == 0);
        }
}
```

Εφαρμογές

- Υπολόγισε την δυαδική μορφή ενός ακεραίου.

```
class Binary
{
	public static void main(String[] args)
	{
		Stack myStack = new Stack(100);
		int number = 1973;

		while (number > 0){
			myStack.push(number%2);
			number = number/2;
		}

		while (!myStack.isEmpty()){
			System.out.print(myStack.pop());
		}
	}
}
```

Επεκτάσεις

- Πως θα ορίσουμε την μέθοδο equals?
- Πως θα ορίσουμε τη μέθοδο toString?

```
public String toString(){
        String returnString = "":
        for (int i = 0; i < size; i ++){
                returnString = returnString + elements[i] + " ";
        }
        return returnString;
}

public boolean equals(Stack other){
        if (this.size != other.size){
                return false;
        }
        for (int i = 0; i < size; i ++){
                if (this.elements[i] != other.elements[i]){
                        return false;
                }
        }
        return true;
}
```