

**Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Εφαρμοσμένων Μαθηματικών και Φυσικών Επιστήμων
Τομέας Μαθηματικών**

**ΕΙΣΑΓΩΓΗ ΣΤΟΝ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟ
9 Σεπτεμβρίου 2005**

- Να απαντηθούν **ΟΔΑ (5)** τα θέματα.
- Καλή επιτυχία.

Θέμα 1^ο

Να γράφει η στατική μέθοδος **intArrayOddElements** η οποία δέχεται ως παράμετρο ένα μονοδιάστατο διάνυσμα από ακεραίους και επιστρέφει τον αριθμό των περιττών (δεν είναι πολλαπλάσια του 2) στοιχείων του.

Θέμα 2^ο

Να γράφει η στατική μέθοδος **intArrayShuffle** η οποία δέχεται ως παράμετρο ένα μονοδιάστατο διάνυσμα από ακεραίους και το «ανακατεύει». Ως «ανακάτεμα» ενός διανύσματος ορίζεται η αναδιάταξη του έτσι ώστε το αρχικά πρώτο στοιχείο ανταλλάσσεται με το δεύτερο, το αρχικά τρίτο στοιχείο με το τέταρτο, κοκ. Γενικότερα το i-οστό στοιχείο πριν το «ανακάτεμα» (όπου i περιττός αριθμός) ανταλλάσσεται με το (i+1)-οστό στοιχείο. Στην περίπτωση όπου το πλήθος των στοιχείων του διανύσματος είναι περιττό, το τελευταίο στοιχείο παραμένει στη θέση του.

Θέμα 3^ο

Να γράφει ο κώδικας της **στατικής** μεθόδου **relativePrimes** η οποία δέχεται ως παραμέτρους δύο θετικούς ακέραιους **m**, **n** και ελέγχει το εάν οι αριθμοί είναι «πρώτοι μεταξύ τους». Δυο θετικοί ακέραιοι είναι «πρώτοι μεταξύ τους» εάν ο μόνος κοινός διαιρέτης τους είναι η μονάδα (1). Να μην γίνει χρήση μεθόδων βιβλιοθήκης.

Θέμα 4^ο

Να γράφει η στατική μέθοδος **fill2DRandomInt** η οποία δέχεται ως παράμετρο ένα δισδιάστατο διάνυσμα από ακεραίους (**int**) (όλες οι γραμμές έχουν τον ίδιο αριθμό στοιχείων) και το γεμίζει με τυχαίες ακέραιες τιμές από το διάστημα [1..6]. Να γίνει χρήση της κλάσης **Random** του πακέτο **java.util**, η περιγραφή της οποίας δίνεται στην τελευταία σελίδα.

Θέμα 5^ο

Να σχεδιαστεί η κλάση **Counter** (μετρητής) η οποία υλοποιεί έναν μετρητή με μεταβλητό βήμα¹ (step). Η κλάση **Counter** θα πρέπει να περιλαμβάνει τις μεθόδους:

- Counter()	Κατασκευάζει ένα μετρητή με αρχική τιμή “μηδέν” και βήμα “ένα” (step=1)
- setStep(int)	Δίνει στο βήμα την τιμή της παραμέτρου
- increment()	Αυξάνει την τιμή του μετρητή κατά τον αριθμό των μονάδων που προσδιορίζει το βήμα
- getValue()	Επιστρέφει την τιμή του μετρητή
- reset()	Μηδενίζει το μετρητή
- setValue(int)	Δίνει στον μετρητή την τιμή της παραμέτρου

¹ Το βήμα είναι η ποσότητα με την οποία αυξάνεται κάθε φορά ο μετρητής.

The screenshot shows the Java API documentation for the `java.util.Random` class. The page title is "Class Random". The navigation bar includes links for Overview, Package, Class, Use, Tree, Deprecated, Index, Help, and several sub-links like PREV CLASS, NEXT CLASS, NESTED, FIELD, CONSTSTR, I, METHOD, FRAMES, NO FRAMES, and ALLCLASSES. The content area contains the class definition and its methods. To the right, there are sections for "All Implemented Interfaces:" (Serializable) and "Direct Known Subclasses:" (SecureRandom). The top right corner of the page has the text "Java™ 2 Platform Std. Ed. v1.4.2".

public class Random	Creates a new random number generator
extends Object	<code>Random()</code>
implements Serializable	<code>Random(long seed)</code>
	Creates a new random number generator using a single long seed:
Constructor Summary	
An instance of this class is used to generate a stream of pseudorandom numbers. The class uses a 48-bit seed, which is modified using a linear congruential formula. (See Donald Knuth, <i>The Art of Computer Programming, Volume 2</i> , Section 3.2.1.)	
If two instances of Random are created with the same seed, and the same sequence of method calls is made for each, they will generate and return identical sequences of numbers. In order to guarantee this property, particular algorithms are specified for the class Random. Java implementations must use all the algorithms shown here for the class Random, for the sake of absolute portability of Java code. However, subclasses of class Random are permitted to use other algorithms, so long as they adhere to the general contracts for all the methods.	
The algorithms implemented by class Random use a protected utility method that on each invocation can supply up to 32 pseudorandomly generated bits.	
Many applications will find the <code>random</code> method in class Math simpler to use.	
Since:	
JDK1.0	
See Also:	
<code>Math.random()</code> , Serialized Form	

Method Summary	
<code>protected int <u>next</u>(int bits)</code>	Generates the next pseudorandom number.
<code>boolean <u>nextBoolean</u>()</code>	Returns the next pseudorandom, uniformly distributed boolean value from this random number generator's sequence.
<code>void <u>nextBytes</u>(byte[] bytes)</code>	Generates random bytes and places them into a user-supplied byte array.
<code>double <u>nextDouble</u>()</code>	Returns the next pseudorandom, uniformly distributed double value between 0.0 and 1.0 from this random number generator's sequence.
<code>float <u>nextFloat</u>()</code>	Returns the next pseudorandom, uniformly distributed float value between 0.0 and 1.0 from this random number generator's sequence.
<code>double <u>nextGaussian</u>()</code>	Returns the next pseudorandom, Gaussian ("normally") distributed double value with mean 0.0 and standard deviation 1.0 from this random number generator's Sequence.
<code>int <u>nextInt</u>()</code>	Returns the next pseudorandom, uniformly distributed int value from this random number generator's sequence.
<code>int <u>nextInt</u>(int n)</code>	Returns a pseudorandom, uniformly distributed int value between 0 (inclusive) and the specified value (exclusive), drawn from this random number generator's Sequence.
<code>long <u>nextLong</u>()</code>	Returns the next pseudorandom, uniformly distributed long value from this random number generator's sequence.
<code>void <u>setSeed</u>(long seed)</code>	Sets the seed of this random number generator using a single long seed.

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait