# Κεφάλαιο IV :
# Εργαστηριακές ασκήσεις που αφορούν πίνακες και μεθόδους στη Java.

Στο παρόν κεφάλαιο παρουσιάζονται εργαστηριακές ασκήσεις οι οποίες αφορούν την χρήση πινάκων και την δημιουργία και χρήση μεθόδων στη Java. Ποιο συγκεκριμένα παρουσιάζονται ασκήσεις οι οποίες αναφέρονται σε:

- **Μονοδιάστατους πίνακες αριθμών**
- **Πίνακες χαρακτήρων**
- **Ανακατατάξεις στοιχείων ενός πίνακα**
- **Διδιαστατους πίνακες**
- **Μεθόδους**
- **Μεθόδους Recursive**
- **Overloading**

## 4.1 Λυμένες Ασκήσεις.

**4.1.1** Γράψτε ένα πρόγραμμα στο οποίο να δημιουργήσετε δέκα τυχαίους αριθμούς διπλής ακρίβειας και να τους αποθηκεύσετε σε κατάλληλο πίνακα. Στη συνέχεια χρησιμοποιείστε τον πίνακα για να τους εκτυπώσετε. Εκτυπώστε επίσης τον συνολικό αριθμό των στοιχείων του πίνακα που δημιουργήσατε χρησιμοποιώντας την μεταβλητή length.

Μια πιθανή λύση είναι η ακόλουθη:

```
import java.util.Random;

class TestArray
{
    public static void main(String[] arguments)
    {
        Random rndm = new Random();  // Dimiourgia antikeimenou rndm
        double[] x = new double[10]; // Dimiourgia pinaka me 10 stoixeia

        for(int i=0; i<10; i++)      // Apothikeysi ston pinaka 10 tixaion arithmon
            x[i]=rndm.nextDouble();

        for(int i=0; i<10; i++)      // Ektiposi tou pinaka
           System.out.println("x[" + i +"] = " + x[i]);

        System.out.println("O arithmos stoixeion tou pinaka einai : " + x.length);
    }
}
-:--  TestArray.java      (Java Abbrev)--L19--All----------------------------------------
```

Η εκτέλεση του παραπάνω προγράμματος είναι η ακόλουθη:

```
[student1@pc244 kef4]$
[student1@pc244 kef4]$
[student1@pc244 kef4]$ javac TestArray.java
[student1@pc244 kef4]$ java TestArray
x[0] = 0.503271039665935
x[1] = 0.9783879130795229
x[2] = 0.21364774265251452
x[3] = 0.29745598349172797
x[4] = 0.19127271972349125
x[5] = 0.9438726641065905
x[6] = 0.640012739787724
x[7] = 0.995834222249497
x[8] = 0.5741069262170253
x[9] = 0.1644051660305984
O arithmos stoixeion tou pinaka einai : 10
[student1@pc244 kef4]$
```

Προσέξτε πως σε έναν πίνακα με δέκα στοιχεία αυτά αριθμούνται από το 0 έως το 9.

**4.1.2** Γράψτε ένα πρόγραμμα στο οποίο να ορίσετε ένα κατάλληλο String και να αποθηκεύσετε την λέξη "Ioannina". Στην συνέχεια να ορίσετε έναν κατάλληλο πίνακα χαρακτήρων και να αποθηκεύσετε σε κάθε στοιχείο τον αντίστοιχο χαρακτήρα του String που ορίσατε (χρησιμοποιείστε την μέθοδο toCharArray() της κλάσης String). Στη συνέχεια εκτυπώστε το String και το μήκος του (χρησιμοποιώντας την μέθοδο length() της κλάσης String). Εκτυπώστε επίσης τον συνολικό αριθμό των στοιχείων του πίνακα χαρακτήρων που δημιουργήσατε (χρησιμοποιώντας την μεταβλητή length) και ένα-ένα τα στοιχεία του.

Μια πιθανή λύση είναι η ακόλουθη:

```
class CharArray
{
    public static void main(String[] arguments)
    {
        String name = "Ioannina";          // Orismos symboloseiras
        char[] a = name.toCharArray();     // Symboloseira se pinaka xaraktiron

        System.out.println("name = \"" + name + "\"");
        System.out.println("name.length = " + name.length());
        System.out.println("a.length = " + a.length);

        for(int i=0; i<a.length; i++)
           System.out.println("a[" + i +"] = " + a[i]);

    }
}
-:--  CharArray.java      (Java Abbrev)--L17--All---------------------------------
```

Η εκτέλεση του παραπάνω προγράμματος είναι η ακόλουθη:

```
[student1@pc244 kef4]$
[student1@pc244 kef4]$ javac CharArray.java
[student1@pc244 kef4]$ java CharArray
name = "Ioannina"
name.length = 8
a.length = 8
a[0] = I
a[1] = o
a[2] = a
a[3] = n
a[4] = n
a[5] = i
a[6] = n
a[7] = a
[student1@pc244 kef4]$
```

**4.1.3** Γράψτε ένα πρόγραμμα στο οποίο να δημιουργήσετε δέκα τυχαίους αριθμούς διπλής ακρίβειας και να τους αποθηκεύσετε σε κατάλληλο πίνακα. Χρησιμοποιώντας τον πίνακα να τους εκτυπώσετε. Ανακατατάξτε τον πίνακα σε φθίνουσα σειρά. Εκτυπώστε εκ νέου τον πίνακα.

Μια πιθανή λύση είναι η ακόλουθη:

```
import java.util.Random;

class TestOrder
{
    public static void main(String[] arguments)
    {
        Random rndm = new Random();  // Dimiourgia antikeimenou rndm
        double[] x = new double[10]; // Dimiourgia pinaka me 10 stoixeia

        // Apothikeysi ston pinaka 10 tixaion arithmon
        for(int i=0; i<10; i++)
            x[i]=rndm.nextDouble();

        // Ektiposi tou pinaka
        System.out.println("Pinakas 10 tixaion arithmon :");
        for(int i=0; i<10; i++)
           System.out.println("x[" + i +"] = " + x[i]);

        double temp=0.;
        for(int i=0; i<10; i++){        // Katataxi stoixeion se fthinousa seira
            for(int j=i+1; j<10; j++){  // Ypologismos megaliterou apo ta ypoloipa stoixeia
                if(x[j]>x[i]){
                    temp = x[i];        // Enalagi ton dyo stoixeion
                    x[i] = x[j];
                    x[j] = temp;
                }
            }
        }

        // Epanektiposi tou pinaka
        System.out.println();
        System.out.println("Epanektiposi stoixeion se fthinousa seira :");
        for(int i=0; i<10; i++)
           System.out.println("x[" + i +"] = " + x[i]);

    }
}
-:--  TestOrder.java      (Java Abbrev)--L38--All----------------------------------------------
```

Η εκτέλεση του παραπάνω προγράμματος είναι η ακόλουθη:

```
[student1@pc244 kef4]$
[student1@pc244 kef4]$ javac TestOrder.java
[student1@pc244 kef4]$ java TestOrder
Pinakas 10 tixaion arithmon :
x[0] = 0.36482825238871197
x[1] = 0.8313635968219164
x[2] = 0.11508507178050276
x[3] = 0.9913759815968243
x[4] = 0.793522946557893
x[5] = 0.2134634873288478
x[6] = 0.6039919196147282
x[7] = 0.5010108362697533
x[8] = 0.8345464492543351
x[9] = 0.48922620546231255

Epanektiposi stoixeion se fthinousa seira :
x[0] = 0.9913759815968243
x[1] = 0.8345464492543351
x[2] = 0.8313635968219164
x[3] = 0.793522946557893
x[4] = 0.6039919196147282
x[5] = 0.5010108362697533
x[6] = 0.48922620546231255
x[7] = 0.36482825238871197
x[8] = 0.2134634873288478
x[9] = 0.11508507178050276
[student1@pc244 kef4]$
```

**4.1.4** Γράψτε ένα πρόγραμμα στο οποίο να δημιουργήσετε τον κατάλληλο διδιάστατο πίνακα ακεραίων αριθμών και αποθηκεύσετε σε αυτόν τα παρακάτω δεδομένα:

$$A = \begin{bmatrix} 15 & 22 & 43 & 10 \\ 56 & 98 & 34 & 43 \\ 82 & 65 & 19 & 39 \end{bmatrix}$$

Εκτυπώστε τον αριθμό των στηλών και των γραμμών του (χρησιμοποιείστε την μεταβλητή length). Εκτυπώστε τα στοιχεία του υπό μορφή στηλών-γραμμών.

Μια πιθανή λύση είναι η ακόλουθη:

```
class TwoDimArray
{
    public static void main(String[] arguments)
    {
        int[][] a = { {15, 22, 43, 10},
                      {56, 98, 34, 43},
                      {82, 65, 19, 39}};

        System.out.println("a.length    = " + a.length);
        System.out.println("a[0].length = " + a[0].length);


        for(int i=0; i<a.length; i++){
            for(int j=0; j<a[0].length; j++){
                System.out.print(a[i][j] + "  ");
            }
            System.out.println();
        }
    }
}
-:--  TwoDimArray.java      (Java Abbrev)--L21--All-------------------
```

Η εκτέλεση του παραπάνω προγράμματος είναι η ακόλουθη:

```
[student1@pc244 kef4]$
[student1@pc244 kef4]$ javac TwoDimArray.java
[student1@pc244 kef4]$ java TwoDimArray
a.length    = 3
a[0].length = 4
15  22  43  10
56  98  34  43
82  65  19  39
[student1@pc244 kef4]$
```

**4.1.5** Γράψτε μια μέθοδο η οποία να υπολογίζει την τιμή της συνάρτησης $f(n) = \sqrt{3n^2 + 2n + 4}$ . Χρησιμοποιείστε την μέθοδο σε ένα πρόγραμμα για να τυπώσετε την τιμή της συνάρτησης για n=1, 2, 3….10.

Μια πιθανή λύση είναι η ακόλουθη:

```
class Fn
{
    public static void main(String[] arguments)
    {
        for(int n=1;n<=10;n++)
            System.out.println("n=" + n + " f(n)=" + fi(n));

    }


    static double fi(int i)
    {
        return Math.sqrt(3*i*i+2*i+4);
    }

}
-:--  Fn.java          (Java Abbrev)--L17--All----------------------
```

Η εκτέλεση του παραπάνω προγράμματος είναι η ακόλουθη:

```
[student1@pc244 kef4]$
[student1@pc244 kef4]$ javac Fn.java
[student1@pc244 kef4]$ java Fn
n=1 f(n)=3.0
n=2 f(n)=4.47213595499958
n=3 f(n)=6.082762530298219
n=4 f(n)=7.745966692414834
n=5 f(n)=9.433981132056603
n=6 f(n)=11.135528725660043
n=7 f(n)=12.84523257866513
n=8 f(n)=14.560219778561036
n=9 f(n)=16.278820596099706
n=10 f(n)=18.0
[student1@pc244 kef4]$
```

**4.1.6** Γράψτε μια μέθοδο η οποία να υπολογίζει την τιμή της συνάρτησης (αριθμός μεταθέσεων k προς n)

$$p(n,k) = \prod_{i=n-k+1}^{n} i = (n-k+1)(n-k+2)...)n-2)(n-1)(n)$$

Χρησιμοποιείστε την μέθοδο σε ένα πρόγραμμα για να τυπώσετε την τιμή της συνάρτησης για n=1, 2, 3….10 και k<=n.

Μια πιθανή λύση είναι η ακόλουθη:

```
class Permutation
{
    public static void main(String[] arguments)
    {
        for(int i=0; i<9; i++){
            for(int j=0; j<=i; j++)
                    System.out.print(per(i,j) + "\t");

            System.out.println();
        }
    }


    static long per(int n, int m)    // Methodos ypologismou tou p(n,m)
    {
        long result=1;
        for(int i=0; i<m; i++)
            result *= n--;

        return result;
    }
}
-:--  Permutation.java      (Java Abbrev)--L23--All-----------------------
```

Η εκτέλεση του παραπάνω προγράμματος είναι η ακόλουθη:

```
[student1@pc244 kef4]$
[student1@pc244 kef4]$ javac Permutation.java
[student1@pc244 kef4]$ java Permutation
1
1       1
1       2       2
1       3       6       6
1       4       12      24      24
1       5       20      60      120     120
1       6       30      120     360     720     720
1       7       42      210     840     2520    5040    5040
1       8       56      336     1680    6720    20160   40320   40320
[student1@pc244 kef4]$
```

**4.1.7** Γράψτε μια μέθοδο η οποία να υπολογίζει την τιμή της συνάρτησης του παραγοντικού

$n! = 1 \cdot 2 \cdot 3 \cdot 4 \cdots (n-1) \cdot n$ με 0!=1

Χρησιμοποιείστε την μέθοδο σε ένα πρόγραμμα για να τυπώσετε την τιμή της συνάρτησης για n=1, 2, 3....8.

Παρακάτω δίνονται τρεις διαφορετικές λύσεις στο πρόβλημα. Στην πρώτη μέθοδο f1() χρησιμοποιείται η εντολή for ενώ στη δεύτερη μέθοδο f2() η εντολή while. Ενδιαφέρον παρουσιάζει η τρίτη λύση με την μέθοδο f3() η οποία για να υπολογίσει το παραγοντικό καλεί τον εαυτό της. Όταν μία μέθοδος καλεί τον εαυτό της ονομάζεται **Recursive**.

```
class TestFactorial
{
    public static void main(String[] arguments)
    {
        for(int n=0;n<=8;n++)
            System.out.println(n + "! = " + f1(n)+ "\t" + f2(n)+ "\t" + f3(n));

    }

    static double f1(int n)     // Proti Methodos ypologismou paragontikou
    {
        double result=1.;
        for(int i=1;i<=n;i++)
            result*=i;

        return result;
    }

    static double f2(int n)   // Deyteri Methodos ypologismou paragontikou
    {
        double result=1.;
        while(n>1)
            result*=n--;

        return result;
    }

    static double f3(int n)   // Triti Methodos ypologismou paragontikou
    {                         // Recursive Methodos
        if (n<2)              // 0!=1 kai 1!=1
            return 1.;
        return n*f3(n-1);     // n!=n*(n-1)!
    }

}
-:--  TestFactorial.java      (Java Abbrev)--L36--All-----------------------------
```

Η εκτέλεση του παραπάνω προγράμματος είναι η ακόλουθη:

```
[student1@pc244 kef4]$
[student1@pc244 kef4]$
[student1@pc244 kef4]$ javac TestFactorial.java
[student1@pc244 kef4]$ java TestFactorial
0! = 1.0        1.0     1.0
1! = 1.0        1.0     1.0
2! = 2.0        2.0     2.0
3! = 6.0        6.0     6.0
4! = 24.0       24.0    24.0
5! = 120.0      120.0   120.0
6! = 720.0      720.0   720.0
7! = 5040.0     5040.0  5040.0
8! = 40320.0    40320.0 40320.0
[student1@pc244 kef4]$
```

**4.1.8** Δίνεται η ακόλουθη σειρά:

$$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + .... = \sum_n \frac{x^n}{n!} \quad \text{για} \qquad -\infty < x < \infty$$

Να γράψετε μια μέθοδο η οποία να υπολογίζει το παραγοντικό ενός μη μηδενικού ακεραίου αριθμού. Να χρησιμοποιήσετε την μέθοδο που αναπτύξατε σε ένα πρόγραμμα το οποίο να υπολογίζει χωριστά τα δύο μέλη της παραπάνω παράστασης. Το δεύτερο μέλος να το εκτυπώσετε για n=0,1,2…10 και να παρατηρήσετε την σύγκληση. Το x να δίνεται από το πληκτρολόγιο.

Μια πιθανή λύση είναι η ακόλουθη:

```
import java.io.*;

class Seiraex
{
    public static void main(String[] arguments) throws IOException
    {
        InputStreamReader reader = new InputStreamReader(System.in);
        BufferedReader input = new BufferedReader(reader);

        System.out.print("Dosete ton arithmo x : ");
        String arithmos = input.readLine();        // Eisagogi tou arithmou
        double x = Double.parseDouble(arithmos);   // Metaropi se double

        System.out.println("e^x =" + Math.exp(x));

        double seira=0.;
        for(int i=0;i<=10;i++) {
            seira+=Math.pow(x,i)/paragontiko(i);
            System.out.println("i=" + i + "  Seira = " + seira );
        }
    }

    static double paragontiko(int n)
    {
        double result=1.;
        for(int i=1;i<=n;i++)
            result*=i;

        return result;
    }
}
-:--  Seiraex.java      (Java Abbrev)--L31--All--------------------------------
```

Η εκτέλεση του παραπάνω προγράμματος για x=2.3 είναι η ακόλουθη:

```
[student1@pc244 kef4]$ javac Seiraex.java
[student1@pc244 kef4]$ java Seiraex
Dosete ton arithmo x : 2.3
e^x =9.974182454814718
i=0  Seira = 1.0
i=1  Seira = 3.3
i=2  Seira = 5.944999999999999
i=3  Seira = 7.972833333333332
i=4  Seira = 9.138837499999998
i=5  Seira = 9.675199416666665
i=6  Seira = 9.880804818055553
i=7  Seira = 9.948360878511902
i=8  Seira = 9.967783245893102
i=9  Seira = 9.972746739779408
i=10  Seira = 9.97388834337326
[student1@pc244 kef4]$
```

**4.1.9** Γράψτε μια μέθοδο

static void print_array(double[] x)

η οποία να τυπώνει τα στοιχεία ενός πίνακα και μία μέθοδο

static void order_array(double[] x)

η οποία να ανακατατάσσει κατ΄ αύξουσα σειρά τα στοιχεία ενός πίνακα αριθμών. Στη συνέχεια γράψτε ένα πρόγραμμα στο οποίο να δημιουργήσετε δέκα τυχαίους αριθμούς διπλής ακρίβειας και να τους αποθηκεύσετε σε κατάλληλο πίνακα. Εκτυπώστε τον πίνακα χρησιμοποιώντας την μέθοδο print_array(). Ανακατατάξτε τον πίνακα κατ΄ αύξουσα σειρά χρησιμοποιώντας την μέθοδο order_array(). Εκτυπώστε εκ νέου τον πίνακα χρησιμοποιώντας την μέθοδο print_array().

Μια πιθανή λύση είναι η ακόλουθη:

```
import java.util.Random;
class ArrayOrder
{
    public static void main(String[] arguments)
    {
        Random rndm = new Random();  // Dimiourgia antikeimenou rndm
        double[] x = new double[10]; // Dimiourgia pinaka me 10 stoixeia
        // Apothikeysi ston pinaka 10 tixaion arithmon
        for(int i=0; i<10; i++)
            x[i]=rndm.nextDouble();
        // Ektiposi tou pinaka
        System.out.println("Pinakas 10 tixaion arithmon :");
        print_array(x);
        // Anakatataxi stoixeion se ayxousa seira
        order_array(x);
        // Epanektiposi tou pinaka
        System.out.println();
        System.out.println("Epanektiposi stoixeion se ayxousa seira :");
        print_array(x);
    }

    static void print_array(double[] x)
    {
        int n = x.length;
        for(int i=0; i<n; i++)
           System.out.println("x[" + i +"] = " + x[i]);
    }

    static void order_array(double[] a)
    {
        int n =a .length;
        double temp=0.;
        for(int i=0; i<n; i++){        // Katataxi stoixeion se fthinousa seira
            for(int j=i+1; j<n; j++){  // Ypologismos mikroterou apo ta ypoloipa stoixeia
                if(a[j]<a[i]){
                    temp = a[i];        // Enalagi ton dyo stoixeion
                    a[i] = a[j];
                    a[j] = temp;
                }
            }
        }
    }
}
-:--  ArrayOrder.java      (Java Abbrev)--L44--Bot----------------------------------------
```

Η εκτέλεση του παραπάνω προγράμματος είναι η ακόλουθη:

```
[student1@pc244 kef4]$
[student1@pc244 kef4]$ javac ArrayOrder.java
[student1@pc244 kef4]$ java ArrayOrder
Pinakas 10 tixaion arithmon :
x[0] = 0.915609324314168
x[1] = 0.8287580790541387
x[2] = 0.012979582352402042
x[3] = 0.470808454276619
x[4] = 0.9105865594177187
x[5] = 0.4098547498810259
x[6] = 0.6495752966324677
x[7] = 0.8733446039851854
x[8] = 0.04223860378544764
x[9] = 0.3037211122855643

Epanektiposi stoixeion se ayxousa seira :
x[0] = 0.012979582352402042
x[1] = 0.04223860378544764
x[2] = 0.3037211122855643
x[3] = 0.4098547498810259
x[4] = 0.470808454276619
x[5] = 0.6495752966324677
x[6] = 0.8287580790541387
x[7] = 0.8733446039851854
x[8] = 0.9105865594177187
x[9] = 0.915609324314168
[student1@pc244 kef4]$
```

**4.1.10** Γράψτε μια μέθοδο

static int max(int i, int j)

η οποία να επιστέφει τον μεγαλύτερο αριθμητικά από δύο ακεραίους αριθμούς και μία μέθοδο

static int max(int i, int j, int k)

η οποία να επιστέφει τον μεγαλύτερο αριθμητικά από τρεις ακεραίους αριθμούς. Στη συνέχεια γράψτε ένα πρόγραμμα στο οποίο να δημιουργήσετε τρεις τυχαίους ακεραίους αριθμούς. Εκτυπώστε τον μεγαλύτερο από τους δύο πρώτους ακεραίους χρησιμοποιώντας την πρώτη μέθοδο max(). Στη συνέχεια εκτυπώστε τον μεγαλύτερο από τους τρεις ακεραίους χρησιμοποιώντας την δεύτερη μέθοδο max().

Στη Java είναι δυνατόν να έχουμε μεθόδους με τον ίδιο όνομα αλλά με διαφορετικό αριθμό ορισμάτων. Αυτό ονομάζεται **Overloading**. Μια πιθανή λύση είναι η ακόλουθη:

```
// Paradeigma Overloading
import java.util.Random;

class TestMax
{
    public static void main(String[] arguments)
    {
        Random rndm = new Random();  // Dimiourgia antikeimenou rndm

        int n1 = rndm.nextInt();     // Dimiourgia 1ou tixaiou akeraiou
        int n2 = rndm.nextInt();     // Dimiourgia 2ou tixaiou akeraiou
        int n3 = rndm.nextInt();     // Dimiourgia 3ou tixaiou akeraiou

        System.out.print("O megalyteros apo tous arithmous : ");
        System.out.println(n1 + "  " + n2);
        System.out.println("einai o : " + max(n1,n2));

        System.out.print("O megalyteros apo tous arithmous : ");
        System.out.println(n1 + "  " + n2 + "  " + n3);
        System.out.println("einai o : " + max(n1,n2,n3));

    }

    static int max(int i, int j)          // H methodos max me dio orismata
    {
        if(i<j) return j;
        else return i;
    }

    static int max(int i, int j, int k)  // H methodos max me tria orismata
    {
        return max(max(i,j),k);
    }
}
-:--  TestMax.java      (Java Abbrev)--L35--All----------------------------------
```

Η εκτέλεση του παραπάνω προγράμματος είναι η ακόλουθη:

```
[student1@pc244 kef4]$
[student1@pc244 kef4]$ javac TestMax.java
[student1@pc244 kef4]$ java TestMax
O megalyteros apo tous arithmous : -1582834380  205320236
einai o : 205320236
O megalyteros apo tous arithmous : -1582834380  205320236  1150216003
einai o : 1150216003
[student1@pc244 kef4]$
```

## 4.2 Ασκήσεις.

**4.2.1** Δίνετε ο πίνακας

int[] list_num = {-7, 18, 5, 9, -11, 0, 23, 13, -9, -1}

Να γράψετε ένα πρόγραμμα το οποίο να υπολογίζει το μεγαλύτερο και το μικρότερο αλγεβρικά στοιχείο καθώς και τη θέση τους μέσα στον πίνακα. Τυπώστε τα αποτελέσματα.

**4.2.2** Γράψτε ένα πρόγραμμα στο οποίο να εισάγετε από το πληκτρολόγιο δέκα αριθμούς τύπου float. Αποθηκεύστε τους σε έναν πίνακα. Υπολογίστε το άθροισμα των στοιχείων, τον μέσο, την απόκλιση κάθε στοιχείου από τον μέσο και την τυπική απόκλιση. Εκτελέστε το πρόγραμμα και τυπώστε τα αποτελέσματα για τα εξής δεδομένα: 27.5, 13.4, 53.8, 29.2, 74.2, 87.0, 39.9, 47.7, 8.1 και 63.2.

Δίνονται: Ο μέσος $\bar{x} = \frac{\sum_{i=1}^{m} x_i}{m}$, η απόκλιση από τον μέσο $d_i = x_i - \bar{x}$ και η τυπική απόκλιση $s = \sqrt{\frac{\left(d_1^2 + d_2^2 + ... + d_m^2\right)}{m(m-1)}}$

**4.2.3** Δίνονται οι πίνακες

$$A = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \end{bmatrix} \quad \text{και} \quad B = \begin{bmatrix} 10 & 11 & 12 & 13 \\ 14 & 15 & 16 & 17 \\ 18 & 19 & 20 & 21 \end{bmatrix}$$

Να γραφεί ένα πρόγραμμα το οποίο να υπολογίζει τους πίνακες $C_{ij} = A_{ij} + B_{ij}$ και $D_{ij} = A_{ij} - B_{ij}$. Να τυπωθούν σε μορφή γραμμών-στηλών οι πίνακες $C$ και $D$.

**4.2.4** Δίνετε ο πίνακας

$$A = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \end{bmatrix}$$

Να γραφεί ένα πρόγραμμα το οποίο να υπολογίζει τους πίνακες $B$ και $C$ όπου $B_{ij} = A_{ij}^2$ και $C_{ij} = \sqrt{A_{ij}}$. Να τυπωθούν σε μορφή γραμμών-στηλών οι πίνακες $B$ και $C$.

**4.2.5** Δίνονται οι πίνακες

$$A = \begin{bmatrix} 2.7 & 7.3 & 42.1 & 35 \\ 5.6 & 4.1 & 8.2 & 5.5 \\ 9 & 23 & 67.1 & 3.1 \end{bmatrix} \quad \text{και} \quad B = \begin{bmatrix} 2.4 \\ 7.1 \\ 3.4 \\ 9.2 \end{bmatrix}$$

Να γραφεί ένα πρόγραμμα το οποίο να υπολογίζει το γινόμενό τους, δηλαδή τον πίνακα $C_i = \sum_{j=1}^{4} A_{ij} * B_j$ . Να τυπωθεί σε μορφή γραμμών-στηλών ο πίνακας $C$ .

**4.2.6** Γράψτε μια μέθοδο η οποία να υπολογίζει την τιμή της συνάρτησης (αριθμός μεταθέσεων k προς n)

$$p(n,k) = \frac{n!}{k!(n-k)!}$$

Χρησιμοποιείστε την μέθοδο σε ένα πρόγραμμα για να τυπώσετε την τιμή της συνάρτησης για n=1, 2, 3….10 και k<=n.

**4.2.7** Δίνονται οι ακόλουθες σειρές τριγωνομετρικών και υπερβολικών συναρτήσεων:

$$\sin x = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + .... \qquad \text{για} \qquad -\infty < x < \infty$$

$$\cos x = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + .... \qquad \text{για} \qquad -\infty < x < \infty$$

$$\sinh x = x + \frac{x^3}{3!} + \frac{x^5}{5!} + \frac{x^7}{7!} + .... \qquad \text{για} \qquad -\infty < x < \infty$$

$$\cosh x = 1 + \frac{x^2}{2!} + \frac{x^4}{4!} + \frac{x^6}{6!} + .... \qquad \text{για} \qquad -\infty < x < \infty$$

Για κάθε μια από αυτές να αναπτύξετε ένα πρόγραμμα το οποίο να υπολογίζει χωριστά τα δύο μέρη των παραστάσεων και να επαληθεύσετε τις σχέσεις με μερικά παραδείγματα. Το x να δίνεται από το πληκτρολόγιο.

**4.2.8** Να υπολογιστούν αριθμητικά με τον **κανόνα του τραπεζίου** τα παρακάτω ορισμένα ολοκληρώματα:

$$\int_a^b \sin x dx = -\cos x\Big|_a^b$$

$$\int_a^b \cos(2x)dx = \frac{\sin(2x)}{2}\Big|_a^b$$

Να συγκρίνετε την τιμή που βρίσκετε ολοκληρώνοντας αριθμητικά με αυτή που βρίσκετε ολοκληρώνοντας αναλυτικά . Να χρησιμοποιήσετε 10000 βήματα κατά την αριθμητική ολοκλήρωση. Τα όρια της ολοκλήρωσης να δίνονται από το πληκτρολόγιο. Προστατέψτε το πρόγραμμα από λανθασμένα όρια ολοκλήρωσης. Εκτελέστε το πρόγραμμα για διάφορα όρια ολοκλήρωσης a και b.

**4.2.9** Επαναλάβετε την προηγούμενη άσκηση κάνοντας εφαρμογή του **κανόνα του Simpson** .
**Υπόδειξη:** Για τις δύο τελευταίες ασκήσεις συμβουλευτείτε την βιβλιογραφία ή το διαδίκτυο σχετικά με τον κανόνα Τραπεζίου και τον κανόνα του Simpson.