BÀI TẬP THỰC HÀNH TUẦN 1

1. **Environment:**
   - Cài đặt JDK: http://www.oracle.com/technetwork/java/javase/downloads/index.html
   - Cài đặt IDE
     - NetBeans: : http://www.oracle.com/technetwork/java/javase/downloads/jdk-netbeans-jsp-142931.html
     - hoặc Eclipse: https://eclipse.org/downloads/eclipse-packages/
2. **Viết chương trình đầu tiên**
   - Yêu cầu: Viết chương trình HelloWorld
     - Tạo Java Project với tên **BTH1**
     - Trong **src**, tạo package với tên **bth1.edu.vn**
     - Trong **bth1.edu.vn,** tạo class HelloWorld.java chèn hàm ***public static void main()***
   - Code tham khảo:

```
package tdm.edu.vn;
public class HelloWorld {
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        System.out.println("Xin chào bạn, tôi là TÈO!");
    }
}
```

3. **Bài tập 1:** trong **bth1.edu.vn,** tạo class **BaiTap1.java**
   - Yêu cầu: Viết chương trình tạo một mảng số nguyên ngẫu nhiên có n phần tử (n được nhập từ bàn phím, n >0), mỗi phần tử có giá trị từ 0 →100
     - Xuất giá trị các phần tử của mảng.
     - Tìm phần tử có giá trị nhỏ nhất.
     - Đếm số lượng các phần tử là số nguyên tố.
     - Đếm số phần tử có tổng các chữ số lớn hơn 10.
     - Sắp xếp mảng tăng dần/giảm dần
     - Sắp xếp số chẵn giảm dần, số lẻ tăng dần
     - Tìm phần tử có giá trị x
   - Code tham khảo:

```
// Hàm phát sinh mảng
static void taoMang(int[] A)
{
    for(int i=0; i<A.length; i++)
        A[i]= (int) Math.round(Math.random()*100);
}
// Hàm xuất mảng
static void xuatMang(int []A)
{
    for(int i=0; i<A.length;i++)
        System.out.print(A[i] + " ");
}
```

```java
// Hàm tìm phần tử nhỏ nhất
static int timMin(int []A)
{
    int min=A[0];
    for(int i=1; i<A.length;i++)
        if (min>A[i]) min = A[i];
    return min;
}
// Hàm kiểm tra số nguyên tố
static boolean kiemTraSNT(int n)
{
    if (n<=1) return false;
    for(int i =2; i<=Math.sqrt(n);i++)
        if(n%i==0) return false;
    return true;
}
// Hàm đếm số nguyên tố
static int demSNT(int[] A)
{
    int dem = 0;
    for(int i=0; i<A.length;i++)
        if(kiemTraSNT(A[i])) dem +=1;
    return dem;
}
// Hàm sắp xếp giảm dần
static void sapXepGiam(int[] A)
{
    for(int i=0; i<(A.length-1); i++)
        for(int j=i+1; j<A.length; j++)
            if(A[i]<A[j])   hoanVi(A, i, j);
}
// Hàm sắp xếp chẵn tăng
static void sapChanTang(int[] A)
{
    for(int i=0; i<(A.length-1); i++)
        for(int j=i+1; j<A.length; j++)
            if((A[i]%2==0)&&(A[j]%2==0)&&(A[i]>A[j]))
                hoanVi(A, i, j);
}
// Hàm sắp xếp chẵn tăng
static void sapLeGiam(int[] A)
{
    for(int i=0; i<(A.length-1); i++)
        for(int j=i+1; j<A.length; j++)
            if((A[i]%2!=0)&&(A[j]%2!=0)&&(A[i]<A[j]))
                hoanVi(A, i, j);
}
// Hàm tính tổng các chữ số
static int tongChuSo( int n)
{
    int s = 0;
    while(n!=0)
    {
        s += (n%10);
        n /=10;
    }
    return s;
}
// Hàm đếm số phần tử có tổng các chữ số > 10
static int demTongChuSo(int[] A)
{
    int dem=0;
    for(int i=0; i<A.length; i++)
        if(tongChuSo(A[i])>10) dem +=1;
    return dem;
}
```

```
// Hàm tìm phần tử x trong mảng A
static void tim(int A[], int x)
{
    int vt =-1;
    for(int i=0; i<A.length; i++)
        if(A[i]==x)
        {
            vt=i;
            break;
        }
    if(vt==-1)
        System.out.println("Không tìm thấy!");
    else
        System.out.println("Tìm thấy "+ x + " tại vị trí thứ "+ vt);
}
public static void main(String[] args) {
    // TODO Auto-generated method stub
    Scanner sc = new Scanner(System.in);
    System.out.print("Nhập số nguyên n : ");
    int n = sc.nextInt();
    int[] mangSN = new int[n];
    taoMang(mangSN);
    System.out.print("Xuất mảng: ");
    xuatMang(mangSN);
    System.out.print("\nPhần tử nhỏ nhất là : "+ timMin(mangSN));
    System.out.print("\nCó "+ demSNT(mangSN) +" phần tử là số nguyên tố");
    System.out.print("\nCó "+ demTongChuSo(mangSN) +" phần tử có tổng các chữ số lớn hơn 10");
    Arrays.sort(mangSN);
    System.out.print("\nMảng được sắp xếp tăng dần: ");
    xuatMang(mangSN);
    sapXepGiam(mangSN);
    System.out.print("\nMảng được sắp xếp giảm dần: ");
    xuatMang(mangSN);
    sapChanTang(mangSN);
    System.out.print("\nMảng được sắp xếp chẵn tăng dần: ");
    xuatMang(mangSN);
    System.out.print("\nNhập phần tử cần tìm x =");
    int x=sc.nextInt();
    tim(mangSN,x);
}
```

4. Bài tập 2:

- Yêu cầu 1: Cho ma trận số nguyên cấp n x m. Cài đặt class **MaTran.java** trong **bth1.edu.vn** thực hiện các chức năng sau:
  - Nhập ma trận.
  - In ma trận.
  - Tìm phần tử nhỏ nhất.
  - Tìm phần tử lẻ lớn nhất.
  - Tìm dòng có tổng lớn nhất.
  - Tính tổng các số không phải là số nguyên tố.
- Yêu cầu 2: Tạo class **BaiTap2.java** trong **bth1.edu.vn** chèn hàm ***public static void main().*** Sử dụng lại class **MaTran** để thực hiện tính toán trên ma trận.

- Code tham khảo **MaTran.java**:

```
public class MaTran {
    private int soDong;
    private int soCot;
    private int[][] A;
    MaTran(int n, int m)
    {
        soDong = n;
        soCot =m;
        A = new int[soDong][soCot];
    }
    public void nhap()
    {
        Scanner sc = new Scanner(System.in);
        System.out.println("---Nhập ma trận---");
        for(int i=0; i<soDong;i++)
            for(int j=0;j<soCot;j++)
            {
                System.out.print("A["+i+","+j+"]=");
                A[i][j] = sc.nextInt();
            }
    }
public void xuat()
{
    System.out.print("---Xuất ma trận---");
    for(int i = 0; i<soDong; i++)
    {
        System.out.println();
        for(int j =0; j<soCot; j++)
            System.out.print(A[i][j]+ " ");
    }
}
public int timMin()
{
    int min = A[0][0];
    for(int i=0; i<soDong; i++)
        for(int j=0; j<soCot; j++)
            if(min>A[i][j]) min=A[i][j];
    return min;
}
public int timMaxLe()
{
    int max=A[0][0];
    for(int i=0; i<soDong; i++)
        for(int j=0; j<soCot; j++)
            if((A[i][j]%2!=0)&&(max<A[i][j]))
                max = A[i][j];
    return max;
}
private int tongDong(int[] mang)
{
    int s = 0;
    for(int i =0; i<mang.length; i++)
        s +=mang[i];
    return s;
}
public void timDongMax()
{
    int dong= 0;
    int max= tongDong(A[0]);
    for(int i=1;i<soDong; i++)
    {
        int s = tongDong(A[i]);
        if (s>max)
        {  dong = i; max = s;  }
    }
    System.out.print("\nDòng "+ dong + " có tổng lớn nhất là " + max);
}
```

```
    private boolean kiemTraSNT(int n)
    {
        if (n<=1) return false;
        for(int i =2; i<=Math.sqrt(n);i++)
            if(n%i==0) return false;
        return true;
    }
    public int tongKhongLaSNT()
    {
        int s= 0;
        for(int i=0; i<soDong; i++)
            for(int j=0; j<soCot; j++)
                if(!kiemTraSNT(A[i][j]))
                    s+=A[i][j];
        return s;
    }
}
```

- Code tham khảo **BaiTap2.java**:

```
package tdm.edu.vn;
import java.util.Scanner;

import tdm.edu.vn.*;
public class BaiTap2 {
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        Scanner sc = new Scanner(System.in);
        System.out.print("Nhập số dòng: ");
        int n = sc.nextInt();
        System.out.print("Nhập số cột: ");
        int m = sc.nextInt();
        MaTran mt = new MaTran(n,m);
        mt.nhap();
        mt.xuat();
        System.out.print("\nPhần tử nhỏ nhất là "+ mt.timMin());
        int max = mt.timMaxLe();
        if (max%2==0)
            System.out.print("\nKhông có phần tử lẻ");
        else
            System.out.print("\nPhần tử lẻ lớn nhất là "+ max);
        mt.timDongMax();
        System.out.println("\n Tổng các số không phải là SNT : "+ mt.tongKhongLaSNT());
    }
}
```

5. Bài tập 3:

- Yêu cầu 1 : Cho ma trận vuông số nguyên cấp n. Cài đặt class **MaTranVuong.java** trong **bth1.edu.vn** thực hiện các chức năng sau:
  - Nhập ma trận.
  - In ma trận.
  - Tổng các phần tử thuộc tam giác trên.
  - Tổng các phần tử thuộc tam giác dưới.
  - Kiểm tra xem ma trận có đối xứng qua đường chéo chính?
- Yêu cầu 2: Tạo class **BaiTap3.java** trong **bth1.edu.vn** chèn hàm ***public static void main().*** Sử dụng lại class **MaTranVuong** để thực hiện tính toán trên ma trận vuông.