BÀI TẬP JAVA CÓ LỜI GIẢI

1. Bài tập java cơ bản

Trong phần này, bạn phải nắm được các kiến thức về:

- Các mệnh đề if-else, switch-case.
- Các vòng lặp for, while, do-while.
- Các từ khóa break và continue trong java.
- Các toán tử trong java.
- Mảng (array) trong java.
- File I/O trong java.
- Xử lý ngoại lệ trong java.

Bài 01:

Viết chương trình tìm tất cả các số chia hết cho 7 nhưng không phải bội số của 5, nằm trong đoạn 10 và 200 (tính cả 10 và 200). Các số thu được sẽ được in thành chuỗi trên một dòng, cách nhau bằng dấu phẩy.

Gợi ý:

- Sử dụng vòng lặp for

Code mẫu:

```java
import java.util.ArrayList;
import java.util.List;

public class Bai01 {
    public static void main(String[] args) {
        List<Integer> list = new ArrayList<Integer>();
        for (int i = 10; i < 201; i++) {
            if ((i % 7 == 0) && (i % 5 != 0)) {
                list.add(i);
            }
        }
        // hiển thị list ra màn hình
        showList(list);
    }

    public static void showList(List<Integer> list) {
        if (list != null && !list.isEmpty()) {
            int size = list.size();
            for (int i = 0; i < size - 1; i++) {
                System.out.print(list.get(i) + ", ");
            }
            System.out.println(list.get(size - 1));
        }
    }
}
```

Kết quả:

```
14, 21, 28, 42, 49, 56, 63, 77, 84, 91, 98, 112, 119, 126, 133, 147, 154, 161, 168,
182, 189, 196
```

Bài 02:

Viết một chương trình tính giai thừa của một số nguyên dương n. Với n được nhập từ bàn phím. Ví dụ, n = 8 thì kết quả đầu ra phải là 1*2*3*4*5*6*7*8 = 40320.

Gợi ý:

- Sử dụng đệ quy hoặc vòng lặp để tính giai thừa.

Code mẫu: sử dụng đệ quy

```java
import java.util.Scanner;

public class GiaiThuaDemo2 {
    private static Scanner scanner = new Scanner(System.in);
    /**
     * main
     *
     * @author viettuts.vn
     * @param args
     */
    public static void main(String[] args) {
        System.out.print("Nhập số nguyên dương n = ");
        int n = scanner.nextInt();
        System.out.println("Giai thừa của " + n + " là: " + tinhGiaithua(n));
    }

    /**
     * tinh giai thua
     *
     * @author viettuts.vn
     * @param n: so nguyen duong
     * @return giai thua cua so n
     */
    public static long tinhGiaithua(int n) {
        if (n > 0) {
            return n * tinhGiaithua(n - 1);
        } else {
            return 1;
        }
    }
}
```

Kết quả:

```
Nhập số nguyên dương n = 8
Giai thừa của 8 là: 40320
```

Bài 03:

Hãy viết chương trình để tạo ra một map chứa (i, i*i), trong đó i là số nguyên từ 1 đến n (bao gồm cả 1 và n), n được nhập từ bàn phím. Sau đó in map này ra màn hình. Ví dụ: Giả sử số n là 8 thì đầu ra sẽ là: {1: 1, 2: 4, 3: 9, 4: 16, 5: 25, 6: 36, 7: 49, 8: 64}.

Gợi ý:

- Sử dụng vòng lặp for để lặp i từ 1 đến n.

Code mẫu:

```java
import java.util.HashMap;
import java.util.Map;
import java.util.Scanner;

public class Bai03 {
    private static Scanner scanner = new Scanner(System.in);

    public static void main(String[] args) {
        System.out.print("Nhập số nguyên dương n = ");
        int n = scanner.nextInt();

        Map<Integer, Integer> map = new HashMap<Integer, Integer>();
        for (int i = 1; i < n + 1; i++) {
            map.put(i, i * i);
        }
        System.out.println(map);
    }
}
```

Kết quả:

```
Nhập số nguyên dương n = 10
{1=1, 2=4, 3=9, 4=16, 5=25, 6=36, 7=49, 8=64, 9=81, 10=100}
```

Bài 04:

Viết chương trình giải phương trình bậc 2: ax2 + bx + c = 0.

Code mẫu:

```java
import java.util.Scanner;

/**
 * Giải phương trình bậc 2
 *
 * @author viettuts.vn
 */
public class PhuongTrinhBac2 {
    private static Scanner scanner = new Scanner(System.in);
    /**
     * main
     *
     * @param args
     */
    public static void main(String[] args) {
        System.out.print("Nhập hệ số bậc 2, a = ");
        float a = scanner.nextFloat();
        System.out.print("Nhập hệ số bậc 1, b = ");
        float b = scanner.nextFloat();
        System.out.print("Nhập hằng số tự do, c = ");
        float c = scanner.nextFloat();
        giaiPTBac2(a, b, c);
    }

    /**
     * Giải phương trình bậc 2: ax2 + bx + c = 0
     *
     * @param a: hệ số bậc 2
     * @param b: hệ số bậc 1
     * @param c: số hạng tự do
     */
```

```
    public static void giaiPTBac2(float a, float b, float c) {
        // kiểm tra các hệ số
        if (a == 0) {
            if (b == 0) {
                System.out.println("Phương trình vô nghiệm!");
            } else {
                System.out.println("Phương trình có một nghiệm: "
                        + "x = " + (-c / b));
            }
            return;
        }
        // tính delta
        float delta = b*b - 4*a*c;
        float x1;
        float x2;
        // tính nghiệm
        if (delta > 0) {
            x1 = (float) ((-b + Math.sqrt(delta)) / (2*a));
            x2 = (float) ((-b - Math.sqrt(delta)) / (2*a));
            System.out.println("Phương trình có 2 nghiệm là: "
                    + "x1 = " + x1 + " và x2 = " + x2);
        } else if (delta == 0) {
            x1 = (-b / (2 * a));
            System.out.println("Phương trình có nghiệm kép: "
                    + "x1 = x2 = " + x1);
        } else {
            System.out.println("Phương trình vô nghiệm!");
        }
    }
}
```

Kết quả:

```
Nhập hệ số bậc 2, a = 2
Nhập hệ số bậc 1, b = 1
Nhập hằng số tự do, c = -1
Phương trình có 2 nghiệm là: x1 = 0.5 và x2 = -1.0
```

Bài 05:

Viết chương trình chuyển đổi một số tự nhiên ở hệ số 10 thành một số ở hệ cơ số B (1 <= B <= 32) bất kỳ. Giả sử hệ cơ số cần chuyển là 2 <= B <= 16. Số đại diện cho hệ cơ số B > 10 là A = 10, B = 11, C = 12, D = 13, E = 14, F = 15.

Gợi ý:

- Tham khảo bảng ASCII để chuyển đổi kiểu char thành String. Hàm chr(55 + m) trong ví dụ sau:
- Nếu m = 10 trả về chuỗi "A".
- Nếu m = 11 trả về chuỗi "B".
- Nếu m = 12 trả về chuỗi "C".
- Nếu m = 13 trả về chuỗi "D".
- Nếu m = 14 trả về chuỗi "E".

- Nếu m = 15 trả về chuỗi "F".

Code mẫu:

```
import java.util.Scanner;

public class ConvertNumber {
    public static final char CHAR_55 = 55;
    private static Scanner scanner = new Scanner(System.in);

    /**
     * main
     *
     * @author viettuts.vn
     * @param args
     */
    public static void main(String[] args) {
        System.out.print("Nhập số nguyên dương n = ");
        int n = scanner.nextInt();
        System.out.println("So " + n + " trong he co so 2 = "
                + ConvertNumber.convertNumber(n, 2));
        System.out.println("So " + n + " trong he co so 16 = "
                + ConvertNumber.convertNumber(n, 16));
    }

    /**
     * chuyen doi so nguyen n sang he co so b
     *
     * @author viettuts.vn
     * @param n: so nguyen
     * @param b: he co so
     * @return he co so b
     */
```

```
    public static String convertNumber(int n, int b) {
        if (n < 0 || b < 2 || b > 16 ) {
            return "";
        }

        StringBuilder sb = new StringBuilder();
        int m;
        int remainder = n;

        while (remainder > 0) {
            if (b > 10) {
                m = remainder % b;
                if (m >= 10) {
                    sb.append((char) (CHAR_55 + m));
                } else {
                    sb.append(m);
                }
            } else {
                sb.append(remainder % b);
            }
            remainder = remainder / b;
        }
        return sb.reverse().toString();
    }
}
```

Kết quả:

```
Nhập số nguyên dương n = 15
So 15 trong he co so 2 = 1111
So 15 trong he co so 16 = F
```

Bài 06: Dãy số Fibonacci được định nghĩa như sau: F0 = 0, F1 = 1, F2 = 1, Fn = F(n-1) + F(n-2) với n >= 2. Ví dụ: 0, 1, 1, 2, 3, 5, 8, ... Hãy viết chương trình tìm n số Fibonacci đầu tiên.

Code mẫu:

```java
import java.util.Scanner;

/**
* Tính dãy số Fibonacci bằng phương pháp đệ quy
*
* @author viettuts.vn
*/
public class FibonacciExample2 {
    private static Scanner scanner = new Scanner(System.in);
   /**
    * main
    *
    * @param args
    */
   public static void main(String[] args) {
       System.out.print("Nhập số nguyên dương n = ");
       int n = scanner.nextInt();
       System.out.println(n + " số đầu tiên của dãy số fibonacci: ");
       for (int i = 0; i < n; i++) {
           System.out.print(fibonacci(i) + " ");
       }
   }

   /**
    * Tính số fibonacci thứ n
    *
    * @param n: chỉ số của số fibonacci tính từ 0
    *           vd: F0 = 0, F1 = 1, F2 = 1, F3 = 2
    * @return số fibonacci thứ n
    */
   public static int fibonacci(int n) {
       if (n < 0) {
           return -1;
       } else if (n == 0 || n == 1) {
           return n;
       } else {
           return fibonacci(n - 1) + fibonacci(n - 2);
       }
   }
}
```

Kết quả:

```
Nhập số nguyên dương n = 12
12 số đầu tiên của dãy số fibonacci:
0 1 1 2 3 5 8 13 21 34 55 89
```

Bài 07:

Viết chương trình tìm ước số chung lớn nhất (USCLN) và bội số chung nhỏ nhất (BSCNN) của hai số nguyên dương a và b nhập từ bàn phím.

Gợi ý:

Sử dụng giải thuật Euclid.

Code mẫu:

```java
import java.util.Scanner;

public class USCLL_BSCNN_1 {
    private static Scanner scanner = new Scanner(System.in);
    /**
     * main
     *
     * @param args
     */
    public static void main(String[] args) {
        System.out.print("Nhập số nguyên dương a = ");
        int a = scanner.nextInt();
        System.out.print("Nhập số nguyên dương b = ");
        int b = scanner.nextInt();
        // tính USCLN của a và b
        System.out.println("USCLN của " + a + " và " + b
                + " là: " + USCLN(a, b));
        // tính BSCNN của a và b
        System.out.println("BSCNN của " + a + " và " + b
                + " là: " + BSCNN(a, b));
    }

    /**
     * Tìm ước số chung lớn nhất (USCLN)
     *
     * @param a: số nguyên dương
     * @param b: số nguyên dương
     * @return USCLN của a và b
     */

    public static int USCLN(int a, int b) {
        if (b == 0) return a;
        return USCLN(b, a % b);
    }

    /**
     * Tìm bội số chung nhỏ nhất (BSCNN)
     *
     * @param a: số nguyên dương
     * @param b: số nguyên dương
     * @return BSCNN của a và b
     */
    public static int BSCNN(int a, int b) {
        return (a * b) / USCLN(a, b);
    }
}
```

Kết quả:

```
Nhập số nguyên dương a = 10
Nhập số nguyên dương b = 24
USCLN của 10 và 24 là: 2
BSCNN của 10 và 24 là: 120
```

Bài 08:

Viết chương trình liệt kê tất cả các số nguyên tố nhỏ hơn n. Số nguyên dương n được nhập từ bàn phím.

Code mẫu:

```java
import java.util.Scanner;

/**
 * Chương trình liệt kê tất cả các số nguyên tố nhỏ hơn n.
 *
 * @author viettuts.vn
 */
public class BaiTap08 {
    private static Scanner scanner = new Scanner(System.in);

    /**
     * main
     *
     * @param args
     */
    public static void main(String[] args) {
        System.out.print("Nhập n = ");
        int n = scanner.nextInt();
        System.out.printf("Tất cả các số nguyên tố nhỏ hơn %d là: \n", n);
        if (n >= 2) {
            System.out.print(2);
        }
        for (int i = 3; i < n; i+=2) {
            if (isPrimeNumber(i)) {
                System.out.print(" " + i);
            }
        }
    }

    /**
     * check so nguyen to
     *
     * @author viettuts.vn
     * @param n: so nguyen duong
     * @return true la so nguyen so,
     *         false khong la so nguyen to
     */
```

```
     */
    public static boolean isPrimeNumber(int n) {
        // so nguyen n < 2 khong phai la so nguyen to
        if (n < 2) {
            return false;
        }
        // check so nguyen to khi n >= 2
        int squareRoot = (int) Math.sqrt(n);
        for (int i = 2; i <= squareRoot; i++) {
            if (n % i == 0) {
                return false;
            }
        }
        return true;
    }
}
```

Kết quả:

```
Nhập n = 100
Tất cả các số nguyên tố nhỏ hơn 100 là:
2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89 97
```

Bài 09:

Viết chương trình liệt kê n số nguyên tố đầu tiên trong java. Số nguyên dương n được nhập từ bàn phím.

Code mẫu:

```java
import java.util.Scanner;

/**
 * Chương trình liệt kê n số nguyên tố đầu tiên.
 *
 * @author viettuts.vn
 */
public class BaiTap09 {
    private static Scanner scanner = new Scanner(System.in);

    /**
     * main
     *
     * @param args
     */
    public static void main(String[] args) {
        System.out.print("Nhập n = ");
        int n = scanner.nextInt();
        System.out.printf("%d số nguyên tố đầu tiên là: \n", n);
        int dem = 0; // đếm số số nguyên tố
        int i = 2;   // tìm số nguyên tố bắt đầu từ số 2
        while (dem < n) {
            if (isPrimeNumber(i)) {
                System.out.print(i + " ");
                dem++;
            }
            i++;
        }
    }

    /**
     * check so nguyen to
     *
     * @author viettuts.vn
     * @param n: so nguyen duong
     * @return true la so nguyen so,
     *         false khong la so nguyen to
     */


    public static boolean isPrimeNumber(int n) {
        // so nguyen n < 2 khong phai la so nguyen to
        if (n < 2) {
            return false;
        }
        // check so nguyen to khi n >= 2
        int squareRoot = (int) Math.sqrt(n);
        for (int i = 2; i <= squareRoot; i++) {
            if (n % i == 0) {
                return false;
            }
        }
        return true;
    }
}
```

Kết quả:

```
Nhập n = 10
10 số nguyên tố đầu tiên là:
2 3 5 7 11 13 17 19 23 29
```

Bài 10:

Viết chương trình liệt kê tất cả số nguyên tố có 5 chữ số trong java.

Code mẫu:

```java
public class BaiTap10 {

    /**
     * main
     *
     * @param args
     */
    public static void main(String[] args) {
        int count = 0;
        System.out.println("Liệt kê tất cả số nguyên tố có 5 chữ số:");
        for (int i = 10001; i < 99999; i+=2) {
            if (isPrimeNumber(i)) {
                System.out.println(i);
                count++;
            }
        }
        System.out.println("Tổng các số nguyên tố có 5 chữ số là: " + count);
    }

    /**
     * check so nguyen to
     *
     * @author viettuts.vn
     * @param n: so nguyen duong
     * @return true la so nguyen so,
     *         false khong la so nguyen to
     */
    public static boolean isPrimeNumber(int n) {
        // so nguyen n < 2 khong phai la so nguyen to
        if (n < 2) {
            return false;
        }
        // check so nguyen to khi n >= 2
        int squareRoot = (int) Math.sqrt(n);
        for (int i = 2; i <= squareRoot; i++) {
            if (n % i == 0) {
                return false;
            }
        }
        return true;
    }
}
```

Kết quả:

```
Liệt kê tất cả số nguyên tố có 5 chữ số:
10007
10009
10037
...
99971
99989
99991
Tổng các số nguyên tố có 5 chữ số là: 8363
```

Bài 11:

Viết chương trình phân tích số nguyên n thành các thừa số nguyên tố trong java. Ví dụ: 100 = 2x2x5x5.

Code mẫu:

```java
import java.util.ArrayList;
import java.util.List;
import java.util.Scanner;

/**
 * Chương trình phân tích số nguyên n thành các thừa số nguyên tố.
 * Ví dụ: 12 = 2 x 2 x 3.
 *
 * @author viettuts.vn
 */
public class BaiTap11 {
    private static Scanner scanner = new Scanner(System.in);

    /**
     * main
     *
     * @param args
     */
    public static void main(String[] args) {
        System.out.print("Nhập số nguyên dương n = ");
        int n = scanner.nextInt();
        // phân tích số nguyên dương n
        List<Integer> listNumbers = phanTichSoNguyen(n);
        // in kết quả ra màn hình
        System.out.printf("Kết quả: %d = ", n);
        int size = listNumbers.size();
        for (int i = 0; i < size - 1; i++) {
            System.out.print(listNumbers.get(i) + " x ");
        }
        System.out.print(listNumbers.get(size - 1));
    }

    /**
     * Phân tích số nguyên thành tích các thừa số nguyên tố
     *
     * @param positiveInt
     * @return
     */
```

```
    public static List<Integer> phanTichSoNguyen(int n) {
        int i = 2;
        List<Integer> listNumbers = new ArrayList<Integer>();
        // phân tích
        while (n > 1) {
            if (n % i == 0) {
                n = n / i;
                listNumbers.add(i);
            } else {
                i++;
            }
        }
        // nếu listNumbers trống thì add n vào listNumbers
        if (listNumbers.isEmpty()) {
            listNumbers.add(n);
        }
        return listNumbers;
    }
}
```

Kết quả:

```
Nhập số nguyên dương n = 100
Kết quả: 100 = 2 x 2 x 5 x 5
```

Bài 12:

Viết chương trình tính tổng của các chữ số của môt số nguyên n trong java. Số nguyên dương n được nhập từ bàn phím. Với n = 1234, tổng các chữ số: 1 + 2 + 3 + 4 = 10

Code mẫu:

```java
import java.util.Scanner;

/**
 * Chương trình tính tổng của các chữ số của một số nguyên dương n.
 * Tổng của các chữ số của 6677 là 6 + 6 + 7 + 7 = 26.
 *
 * @author viettuts.vn
 */
public class BaiTap12 {
    private static Scanner scanner = new Scanner(System.in);
    public static int DEC_10 = 10;

    /**
     * main
     *
     * @param args
     */
    public static void main(String[] args) {
        System.out.print("Nhập số nguyên dương n = ");
        int n = scanner.nextInt();
        System.out.printf("Tổng của các chữ số "
                + "của %d là: %d", n, totalDigitsOfNumber(n));
    }

    /**
     * Tính tổng của các chữ số của một số nguyên dương
     *
     * @param n: số nguyên dương
     * @return
     */
    public static int totalDigitsOfNumber(int n) {
        int total = 0;
        do {
            total = total + n % DEC_10;
            n = n / DEC_10;
        } while (n > 0);
        return total;
    }
}
```

Kết quả:

```
Nhập số nguyên dương n = 6677
Tổng của các chữ số của 6677 là: 26
```

Bài 13:

Viết chương trình kiểm tra một số n là số thuận nghịch trong java. Số nguyên dương n được nhập từ bàn phím.

Code mẫu:

```java
package vn.viettuts.baitap;

import java.util.Scanner;

/**
 * Chương trình liệt kê tất cả các số thuận nghịch có 6 chữa số.
 *
 * @author viettuts.vn
 */
public class BaiTap13 {
    private static Scanner scanner = new Scanner(System.in);
    /**
     * main
     *
     * @param args
     */
    public static void main(String[] args) {
        System.out.print("Nhập số nguyên dương n = ");
        int n = scanner.nextInt();
        System.out.println(n + " là số thuận nghịch: " + isThuanNghich(n));
        System.out.print("Nhập số nguyên dương m = ");
        int m = scanner.nextInt();
        System.out.println(n + " là số thuận nghịch: " + isThuanNghich(m));
    }

    /**
     * Kiểm tra số thuận nghịch
     *
     * @param n: số nguyên dương
     * @return true là số thuận nghịch
     *         false không là số thuận nghịch
     */

    public static boolean isThuanNghich(int n) {
        // chuyển đổi số n thành một chuỗi String
        String numberStr = String.valueOf(n);
        // kiểm tra tính thuận nghịch
        int size = numberStr.length();
        for (int i = 0; i < (size/2); i++) {
            if (numberStr.charAt(i) != numberStr.charAt(size - i - 1)) {
                return false;
            }
        }
        return true;
    }
}
```

Kết quả:

```
Nhập số nguyên dương n = 123321
123321 là số thuận nghịch: true
Nhập số nguyên dương m = 123451
123321 là số thuận nghịch: false
```

Bài 14:

Viết chương trình liệt kê các số Fibonacci nhỏ hơn n là số nguyên tố trong java. N là số nguyên dương được nhập từ bàn phím.

Code mẫu:

```
import java.util.Scanner;

/**
 *  Chương trình liệt kê các số Fibonacci nhỏ hơn n là số nguyên tố.
 *  Với n được nhập từ bàn phím.
 *
 * @author viettuts.vn
 */
public class BaiTap14 {
    private static Scanner scanner = new Scanner(System.in);
    /**
     * main
     *
     * @param args
     */
    public static void main(String[] args) {
        System.out.print("Nhập số tự nhiên n = ");
        int n = scanner.nextInt();
        System.out.printf("Các số fibonacci nhỏ hơn %d và "
                + "là số nguyên tố: ", n);
        int i = 0;
        while (fibonacci(i) < 100) {
            int fi = fibonacci(i);
            if (isPrimeNumber(fi)) {
                System.out.print(fi + " ");
            }
            i++;
        }
    }

    /**
     * Tính số fibonacci thứ n
     *
     * @param n: chỉ số của số fibonacci tính từ 0
     *           vd: F0 = 0, F1 = 1, F2 = 1, F3 = 2
     * @return số fibonacci thứ n
     */
```

```
    public static int fibonacci(int n) {
        if (n < 0) {
            return -1;
        } else if (n == 0 || n == 1) {
            return n;
        } else {
            return fibonacci(n - 1) + fibonacci(n - 2);
        }
    }

    /**
     * check so nguyen to
     *
     * @author viettuts.vn
     * @param n: so nguyen duong
     * @return true la so nguyen so,
     *         false khong la so nguyen to
     */
    public static boolean isPrimeNumber(int n) {
        // so nguyen n < 2 khong phai la so nguyen to
        if (n < 2) {
            return false;
        }
        // check so nguyen to khi n >= 2
        int squareRoot = (int) Math.sqrt(n);
        for (int i = 2; i <= squareRoot; i++) {
            if (n % i == 0) {
                return false;
            }
        }
        return true;
    }
}
```

Kết quả:

```
Nhập số tự nhiên n = 100
Các số fibonacci nhỏ hơn 100 và là số nguyên tố: 2 3 5 13 89
```

2. Bài tập chuỗi trong Java

Bài 1: Nhập một sâu ký tự. Đếm số từ của sâu đó (mỗi từ cách nhau bởi một khoảng trắng có thể là một hoặc nhiều dấu cách, tab, xuống dòng). Ví dụ " hoc java co ban den nang cao " có 7 từ.

Lời giải:

Thuật toán:

1. Nếu chuỗi đã cho input = null thì trả về -1. Kết thúc tại đây.
2. Ngược lại, duyệt từ phần tử đầu tiên đến phần tử cuối cùng của chuỗi.

3. Nếu ký tự hiện tại là ký tự chữ (ký tự khác space và tab và xuống dòng) thì ta tìm được một từ. Và đánh dấu từ đó đã được đếm (notCounted = false;). Đến khi gặp ký tự space hoặc tab hoặc xuống dòng thì đánh dấu từ đó đã đếm xong (notCounted = true;) để đếm từ tiếp theo.

Các bạn xin hãy lưu ý: một bài toán có hàng trăm nghìn cách giải, ngoài cách này ra các bạn có thể tự nghĩ cho mình những cách giải khác hoặc tìm trên google!

File: StringExample1.java

```
public class StringExample1 {
    public static final char SPACE = ' ';
    public static final char TAB = '\t';
    public static final char BREAK_LINE = '\n';

    /**
     * main
     *
     * @param args
     */
    public static void main(String[] args) {
        String str = "hoc java     co ban den nang cao"
                + "      \n test";
        System.out.print("Số từ của chuỗi đã cho là: "
                + countWords(str));
    }

    /**
     * Đếm số từ của một chuỗi,
     * giả sử các từ được ngăn cách nhau bởi một hoặc nhiều
     * dấu 'space', tab '\t' và xuống dòng '\n'
     *
     * @param input - chuỗi ký tự
     * @return số từ của chuỗi ký tự input
     */
    public static int countWords(String input) {
        if (input == null) {
            return -1;
        }
        int count = 0;
        int size = input.length();
        boolean notCounted = true;
        for (int i = 0; i < size; i++) {
            if (input.charAt(i) != SPACE && input.charAt(i) != TAB
                    && input.charAt(i) != BREAK_LINE) {
                if(notCounted) {
                    count++;
                    notCounted = false;
                }
            } else {
                notCounted = true;
            }
        }
        return count;
    }
}
```

Kết quả:

```
Số từ của chuỗi đã cho là: 8
```

Bài 2: Viết chương trình java liệt kê số lần xuất hiện của các từ trong một chuỗi.

Lời giải

Trong bài này chúng tôi sử dụng StringBuilder thay vì String trong java để build một từ và sử dụng TreeMap để lưu các từ tìm được và số lần xuất hiện của chúng trong chuỗi đã cho.

File: StringExample2.java

```
import java.util.Map;
import java.util.TreeMap;

/**
 * Chương trình liệt kê số lần xuất hiện của các từ của một chuỗi
 * trong java
 *
 * @author viettuts.vn
 */
public class StringExample2 {
    public static final char SPACE = ' ';
    public static final char TAB = '\t';
    public static final char BREAK_LINE = '\n';

    /**
     * main
     *
     * @param args
     */
    public static void main(String[] args) {
        String str = "hoc java     co ban den nang cao"
                + "      \n hoc c++ co ban den nang cao.";
        System.out.println("---------------------------------");
        System.out.println(str);
        System.out.println("---------------------------------");
        // liệt kê số lần xuất hiện của các từ trong chuỗi trên
        System.out.println("Liệt kê số lần xuất hiện của các từ: ");
        Map<String, Integer> wordMap = countWords(str);
        for (String key : wordMap.keySet()) {
            System.out.print(key + " " + wordMap.get(key) + "\n");
        }
    }

    /**
     * Đếm số từ của một chuỗi,
     * giả sử các từ được ngăn cách nhau bởi một hoặc nhiều
     * dấu 'space', tab '\t' và xuống dòng '\n'
     *
     * @param input - chuỗi ký tự
     * @return số từ của chuỗi ký tự input
     */
```

```
    public static Map<String, Integer> countWords(String input) {
        // khởi tạo wordMap
        Map<String, Integer> wordMap = new TreeMap<String, Integer>();
        if (input == null) {
            return wordMap;
        }
        int size = input.length();
        StringBuilder sb = new StringBuilder();
        for (int i = 0; i < size; i++) {
            if (input.charAt(i) != SPACE && input.charAt(i) != TAB
                    && input.charAt(i) != BREAK_LINE) {
                // build một từ
                sb.append(input.charAt(i));
            } else {
                // thêm từ vào wordMap
                addWord(wordMap, sb);
                sb = new StringBuilder();
            }
        }
        // thêm từ cuối cùng tìm được vào wordMap
        addWord(wordMap, sb);
        return wordMap;
    }

    /**
     * Thêm từ vào wordMap
     *
     * @param wordMap: map chứa các từ và số lần xuất hiện
     * @param sb: từ cần thêm vào wordMap
     */
    public static void addWord(Map<String, Integer> wordMap, StringBuilder sb) {
        String word = sb.toString();
        if (word.length() == 0) {
            return;
        }
        if (wordMap.containsKey(word)) {
            int count = wordMap.get(word) + 1;
            wordMap.put(word, count);
        } else {
            wordMap.put(word, 1);
        }
    }
}
```

Kết quả:

```
---------------------------------
hoc java     co ban den nang cao
 hoc c++ co ban den nang cao.
---------------------------------
Liệt kê số lần xuất hiện của các từ:
ban 2
c++ 1
cao 1
cao. 1
co 2
den 2
hoc 2
java 1
nang 2
```

Lưu ý: trong một số trường hợp phải thao tác nhiều với một chuỗi bạn nên sử dụng StringBuilder thay vì sử dụng String nhé.

Bài 3: Viết chương trình java kiểm tra xem chuỗi s1 chứa chuỗi s2 không?

Lời giải

Để kiểm tra chuỗi s1 chứa chuỗi s2 hay không, bạn có thể sử dụng phương thức contains() trong java.

File: StringExample3.java

```
public class StringExample3 {
    public static void main(String[] args) {
        String str1 = "hoc java co ban den nang cao.";
        String str2 = "java co ban";
        System.out.println(str1.contains(str2));
    }
}
```

Kết quả:

```
true
```

3. Bài tập mảng trong Java

Bài 1: Viết chương trình Java nhập một mảng số nguyên a0, a1, a2, …, an-1. Liệt kê các phần tử xuất hiện trong mảng đúng 1 lần.

Lời giải

Trong bài này chúng tôi sử dụng TreeMap để lưu các từ tìm được và số lần xuất hiện của chúng trong mảng đã cho.

File: BaiTap19.java

```java
import java.util.Map;
import java.util.Scanner;
import java.util.TreeMap;

/**
 * Chương trình liệt kê số lần xuất hiện các phần tử trong một mảng
 * nhập từ bàn phím trong java.
 *
 * @author viettuts.vn
 */
public class BaiTap19 {
    public static Scanner scanner = new Scanner(System.in);

    /**
     * main
     *
     * @param args
     */
    public static void main(String[] args) {
        System.out.print("Nhập số phần tử của mảng: ");
        int n = scanner.nextInt();
        // khởi tạo arr
        int [] arr = new int [n];
        System.out.println("Nhập các phần tử của mảng: ");
        for (int i = 0; i < n; i++) {
            System.out.printf("a[%d] = ", i);
            arr[i] = scanner.nextInt();
        }
        // tìm số lần xuất hiện của các phần tử
        Map<Integer, Integer> map = new TreeMap<Integer, Integer>();
        for (int i = 0; i < n; i++) {
            addElement(map, arr[i]);
        }
        System.out.print("Các phần tử xuất hiện 1 lần: ");
        for (Integer key : map.keySet()) {
            if (map.get(key) == 1) {
                System.out.print(key + " ");
            }
        }
    }

    /**
     * Thêm từ vào map
     *
     * @param wordMap: map chứa các từ và số lần xuất hiện
     * @param sb: từ cần thêm vào wordMap
     */


    public static void addElement(Map<Integer, Integer> map, int element) {
        if (map.containsKey(element)) {
            int count = map.get(element) + 1;
            map.put(element, count);
        } else {
            map.put(element, 1);
        }
    }
}
```

Kết quả:

```
Nhập số phần tử của mảng: 6
Nhập các phần tử của mảng:
a[0] = 1
a[1] = 2
a[2] = 3
a[3] = 1
a[4] = 2
a[5] = 5
Các phần tử xuất hiện 1 lần: 3 5
```

Bài 2: Nhập một mảng số nguyên a0, a1, a2, …, an-1. Liệt kê các phần tử xuất hiện trong mảng đúng 2 lần.

Lời giải

Trong bài này chúng tôi sử dụng TreeMap để lưu các từ tìm được và số lần xuất hiện của chúng trong mảng đã cho.

File: BaiTap20.java

```java
import java.util.Map;
import java.util.Scanner;
import java.util.TreeMap;

/**
 * Chương trình liệt kê số lần xuất hiện các phần tử trong một mảng
 * nhập từ bàn phím trong java.
 *
 * @author viettuts.vn
 */
public class BaiTap20 {
    public static Scanner scanner = new Scanner(System.in);

    /**
     * main
     *
     * @param args
     */
    public static void main(String[] args) {
        System.out.print("Nhập số phần tử của mảng: ");
        int n = scanner.nextInt();
        // khởi tạo arr
        int [] arr = new int [n];
        System.out.println("Nhập các phần tử của mảng: ");
        for (int i = 0; i < n; i++) {
            System.out.printf("a[%d] = ", i);
            arr[i] = scanner.nextInt();
        }
        // tìm số lần xuất hiện của các phần tử
        Map<Integer, Integer> map = new TreeMap<Integer, Integer>();
        for (int i = 0; i < n; i++) {
            addElement(map, arr[i]);
        }
        System.out.print("Các phần tử xuất hiện 1 lần: ");
        for (Integer key : map.keySet()) {
            if (map.get(key) == 2) {
                System.out.print(key + " ");
            }
        }
    }

    /**
     * Thêm từ vào map
     *
     * @param wordMap: map chứa các từ và số lần xuất hiện
     * @param sb: từ cần thêm vào wordMap
     */


    public static void addElement(Map<Integer, Integer> map, int element) {
        if (map.containsKey(element)) {
            int count = map.get(element) + 1;
            map.put(element, count);
        } else {
            map.put(element, 1);
        }
    }
}
```

Kết quả:

```
Nhập số phần tử của mảng: 8
Nhập các phần tử của mảng:
a[0] = 1
a[1] = 3
a[2] = 4
a[3] = 6
a[4] = 5
a[5] = 3
a[6] = 1
a[7] = 1
Các phần tử xuất hiện 2 lần: 3
```

Bài 3: Viết chương trình nhập một mảng số nguyên a0, a1, a2, …, an-1. Liệt kê số lần xuất hiện của các phần tử trong một mảng đã cho.

Lời giải

Trong bài này chúng tôi sử dụng TreeMap để lưu các từ tìm được và số lần xuất hiện của chúng trong mảng đã cho.

File: BaiTap21.java

```java
import java.util.Map;
import java.util.Scanner;
import java.util.TreeMap;

/**
 * Chương trình liệt kê số lần xuất hiện các phần tử trong một mảng
 * nhập từ bàn phím trong java.
 *
 * @author viettuts.vn
 */
public class BaiTap21 {
    public static Scanner scanner = new Scanner(System.in);

    /**
     * main
     *
     * @param args
     */
    public static void main(String[] args) {
        System.out.print("Nhập số phần tử của mảng: ");
        int n = scanner.nextInt();
        // khởi tạo arr
        int [] arr = new int [n];
        System.out.print("Nhập các phần tử của mảng: \n");
        for (int i = 0; i < n; i++) {
            System.out.printf("a[%d] = ", i);
            arr[i] = scanner.nextInt();
        }
        // tìm số lần xuất hiện của các phần tử
        Map<Integer, Integer> map = new TreeMap<Integer, Integer>();
        for (int i = 0; i < n; i++) {
            addElement(map, arr[i]);
        }
        System.out.print("Các phần tử xuất hiện 2 lần: \n");
        for (Integer key : map.keySet()) {
            System.out.printf("%d xuất hiện %d lần.\n", key, map.get(key));
        }
    }

    /**
     * Thêm từ vào map
     *
     * @param wordMap: map chứa các từ và số lần xuất hiện
     * @param sb: từ cần thêm vào wordMap
     */


    public static void addElement(Map<Integer, Integer> map, int element) {
        if (map.containsKey(element)) {
            int count = map.get(element) + 1;
            map.put(element, count);
        } else {
            map.put(element, 1);
        }
    }
}
```

Kết quả:

```
Nhập số phần tử của mảng: 10
Nhập các phần tử của mảng:
a[0] = 1
a[1] = 2
a[2] = 3
a[3] = 4
a[4] = 1
a[5] = 2
a[6] = 2
a[7] = 5
a[8] = 6
a[9] = 7
Các phần tử xuất hiện 2 lần:
1 xuất hiện 2 lần.
2 xuất hiện 3 lần.
3 xuất hiện 1 lần.
4 xuất hiện 1 lần.
5 xuất hiện 1 lần.
6 xuất hiện 1 lần.
7 xuất hiện 1 lần.
```

Bài 4: Viết chương trình Java nhập một mảng số nguyên a0, a1, a2, …, an-1. Hãy sắp xếp mảng theo thứ tự tăng dần.

Lời giải

Sau đây là chương trình Java sắp xếp mảng theo thứ tự tăng dần:

File: BaiTap22.java

```java
import java.util.Scanner;

/**
 * Chương trình sắp xếp mảng số nguyên theo thứ tự tăng dần.
 *
 * @author viettuts.vn
 */
public class BaiTap24 {
    public static Scanner scanner = new Scanner(System.in);

    /**
     * main
     *
     * @param args
     */
    public static void main(String[] args) {
        System.out.print("Nhập số phần tử của mảng: ");
        int n = scanner.nextInt();
        // khởi tạo arr
        int [] arr = new int [n];
        System.out.print("Nhập các phần tử của mảng: \n");
        for (int i = 0; i < n; i++) {
            System.out.printf("a[%d] = ", i);
            arr[i] = scanner.nextInt();
        }
        // sắp xếp dãy số theo thứ tự tăng dần
        sortASC(arr);
        System.out.println("Dãy số được sắp xếp tăng dần: ");
        show(arr);
    }

    /**
     * sắp xếp mảng số nguyên theo thứ tự tăng dần
     *
     * @param arr: mảng các số nguyên
     * @param n: số phần tử của mảng
     */

    public static void sortASC(int [] arr) {
        int temp = arr[0];
        for (int i = 0 ; i < arr.length - 1; i++) {
            for (int j = i + 1; j < arr.length; j++) {
                if (arr[i] > arr[j]) {
                    temp = arr[j];
                    arr[j] = arr[i];
                    arr[i] = temp;
                }
            }
        }
    }

    /**
     * in các phần tử của mảng ra màn hình
     *
     * @param arr: mảng các số nguyên
     * @param n: số phần tử của mảng
     */
    public static void show(int [] arr) {
        for (int i = 0; i < arr.length; i++) {
            System.out.print(arr[i] + " ");
        }
    }
}
```

Kết quả:

```
Nhập số phần tử của mảng: 7
Nhập các phần tử của mảng:
a[0] = 1
a[1] = 2
a[2] = 5
a[3] = 6
a[4] = 3
a[5] = 1
a[6] = 9
Dãy số được sắp xếp tăng dần:
1 1 2 3 5 6 9
```

Bài 4: Đề bài: Viết chương trình Java nhập một mảng số nguyên a0, a1, a2, …, an-1. Hãy sắp xếp mảng theo thứ tự giảm dần.

Lời giải

Sau đây là chương trình Java sắp xếp mảng theo thứ tự giảm dần:

File: BaiTap22.java

```java
import java.util.Scanner;

/**
 * Chương trình sắp xếp mảng số nguyên theo thứ tự giảm dần.
 *
 * @author viettuts.vn
 */
public class BaiTap23 {
    public static Scanner scanner = new Scanner(System.in);

    /**
     * main
     *
     * @param args
     */
    public static void main(String[] args) {
        System.out.print("Nhập số phần tử của mảng: ");
        int n = scanner.nextInt();
        // khởi tạo arr
        int [] arr = new int [n];
        System.out.print("Nhập các phần tử của mảng: \n");
        for (int i = 0; i < n; i++) {
            System.out.printf("a[%d] = ", i);
            arr[i] = scanner.nextInt();
        }
        // sắp xếp dãy số theo thứ tự giảm dần
        sortDESC(arr);
        System.out.println("Dãy số được sắp xếp giảm dần: ");
        show(arr);
    }

    /**
     * sắp xếp mảng số nguyên theo thứ tự giảm dần
     *
     * @param arr: mảng các số nguyên
     * @param n: số phần tử của mảng
     */
```

```
    public static void sortDESC(int [] arr) {
        int temp = arr[0];
        for (int i = 0 ; i < arr.length - 1; i++) {
            for (int j = i + 1; j < arr.length; j++) {
                if (arr[i] < arr[j]) {
                    temp = arr[j];
                    arr[j] = arr[i];
                    arr[i] = temp;
                }
            }
        }
    }

    /**
     * in các phần tử của mảng ra màn hình
     *
     * @param arr: mảng các số nguyên
     * @param n: số phần tử của mảng
     */
    public static void show(int [] arr) {
        for (int i = 0; i < arr.length; i++) {
            System.out.print(arr[i] + " ");
        }
    }
}
```

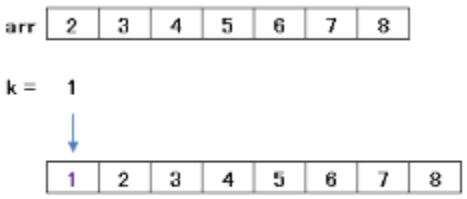
Kết quả:

```
Nhập số phần tử của mảng: 7
Nhập các phần tử của mảng:
a[0] = 1
a[1] = 2
a[2] = 7
a[3] = 6
a[4] = 3
a[5] = 3
a[6] = 5
Dãy số được sắp xếp giảm dần:
7 6 5 3 3 2 1
```

Bài 6: Viết chương trình Java nhập một mảng số nguyên a0, a1, a2, …, an-1. Hãy sắp xếp mảng theo thứ tự tăng dần, sau đó chèn phần tử k vào mà vẫn đảm bảo mảng là tăng dần.

Lời giải

Chèn phần tử vào mảng trong java.



File: BaiTap24.java

```java
import java.util.Scanner;

/**
 * Chương trình sắp xếp mảng theo thứ tự tăng dần,
 * sau đó chèn phần tử k vào mà vẫn đảm bảo mảng là tăng dần.
 *
 * @author viettuts.vn
 */
public class BaiTap24 {
    public static Scanner scanner = new Scanner(System.in);

    /**
     * main
     *
     * @param args
     */
    public static void main(String[] args) {
        System.out.print("Nhập số phần tử của mảng: ");
        int n = scanner.nextInt();
        // khởi tạo arr
        int [] arr = new int [n];
        System.out.print("Nhập các phần tử của mảng: \n");
        for (int i = 0; i < n; i++) {
            System.out.printf("a[%d] = ", i);
            arr[i] = scanner.nextInt();
        }
        System.out.print("Nhập phần tử k = ");
        int k = scanner.nextInt();
        // sắp xếp dãy số theo thứ tự tăng dần
        sortASC(arr);
        System.out.print("Sắp xếp mảng tăng dần: ");
        show(arr);
        System.out.printf("\nChèn phần tử %d vào mảng.", k);
        arr = insert(arr, k);
        System.out.print("\nMảng sau khi chèn: ");
        show(arr);
    }

    /**
     * sắp xếp mảng số nguyên theo thứ tự tăng dần
     *
     * @param arr: mảng các số nguyên
     */
```

```
    public static int [] insert(int [] arr, int k) {
        int arrIndex = arr.length - 1;
        int tempIndex = arr.length;
        int [] tempArr = new int [tempIndex + 1];
        boolean inserted = false;

        for (int i = tempIndex; i >= 0; i--) {
            if (arrIndex > -1 && arr[arrIndex] > k) {
                tempArr[i] = arr[arrIndex--];
            } else {
                if (!inserted) {
                    tempArr[i] = k;
                    inserted = true;
                } else {
                    tempArr[i] = arr[arrIndex--];
                }
            }
        }
        return tempArr;
    }

    /**
     * in các phần tử của mảng ra màn hình
     *
     * @param arr: mảng các số nguyên
     */
    public static void show(int [] arr) {
        for (int i = 0; i < arr.length; i++) {
            System.out.print(arr[i] + " ");
        }
    }
}
```

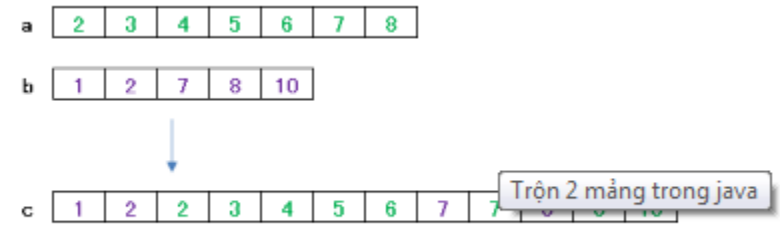
Kết quả:

```
Nhập số phần tử của mảng: 5
Nhập các phần tử của mảng:
a[0] = 2
a[1] = 3
a[2] = 4
a[3] = 5
a[4] = 6
Nhập phần tử k = 1
Sắp xếp mảng tăng dần: 2 3 4 5 6
Chèn phần tử 1 vào mảng.
Mảng sau khi chèn: 1 2 3 4 5 6
```

Bài 7: Viết chương trình Java nhập 2 mảng số thực a0, a1, a2, …, an và b0, b1, b2, …, bm. Giả sử 2 mảng này đã được sắp xếp tăng dần. Hãy tận dụng tính sắp xếp của 2 dãy và tạo dãy c0, c1, c2, …, cn+m là hợp của 2 dãy trên sao cho ci cũng có thứ tự tăng dần.

Lời giải

Trộn 2 mảng trong java.



File: BaiTap24.java

```
import java.util.Scanner;

/**
 * Chương trình mảng số thực a0, a1, a2, …, am-1 và b0, b1, b2, …, bn-1.
 * Giả sử 2 mảng này đã được sắp xếp tăng dần.
 * Hãy tận dụng tính sắp xếp của 2 dãy và tạo dãy c0, c1, c2, …, cm+n-1
 * là hợp của 2 dãy trên sao cho ci cũng có thứ tự tăng dần
 *
 * @author viettuts.vn
 */
public class BaiTap26 {
    public static Scanner scanner = new Scanner(System.in);

    /**
     * main
     *
     * @param args
     */
    public static void main(String[] args) {
        float[] a = null;
        float[] b = null;
        float[] c = null;

        System.out.println("---Nhập mảng a---");
        a = input(a);
        System.out.println("---Nhập mảng b---");
        b = input(b);

        // trộn mảng a và b thành c
        c = merge(a, b);

        // in mảng c ra màn hình
        show(c);
    }

    /**
     * nhập mảng
     *
     * @param arr: mảng số nguyên
     */
    public static float[] input(float[] arr) {
        System.out.print("Nhập số phần tử của mảng: ");
        int n = scanner.nextInt();
        // khởi tạo arr
        arr = new float[n];
        System.out.print("Nhập các phần tử của mảng: \n");
        for (int i = 0; i < n; i++) {
            System.out.printf("a[%d] = ", i);
            arr[i] = scanner.nextInt();
        }
        return arr;
    }
```

```java
    /**
     * sắp xếp mảng số nguyên theo thứ tự tăng dần
     *
     * @param arr: mảng các số nguyên
     * @param n: số phần tử của mảng
     */
    public static void sortASC(float [] arr) {
        float temp = arr[0];
        for (int i = 0 ; i < arr.length - 1; i++) {
            for (int j = i + 1; j < arr.length; j++) {
                if (arr[i] > arr[j]) {
                    temp = arr[j];
                    arr[j] = arr[i];
                    arr[i] = temp;
                }
            }
        }
    }

    /**
     * trộn mảng a và b vào mảng c
     * sao cho c cũng duy trì thứ tự tăng dần.
     *
     * @param a: mảng a
     * @param b: mảng b
     * @return mảng c
     */
    public static float[] merge(float[] a, float[] b) {
        int aIndex = a.length - 1;
        int bIndex = b.length - 1;
        int cIndex = a.length + b.length - 1;
        float[] c = new float[cIndex + 1];

        // sắp xếp các mảng theo thứ tự tăng dần
        sortASC(a);
        sortASC(b);

        // trộn mảng a và b thành c
        for (int i = cIndex; i > -1; i--) {
            if (aIndex > -1 && bIndex > -1) {
                if (a[aIndex] > b[bIndex]) {
                    c[i] = a[aIndex--];
                } else {
                    c[i] = b[bIndex--];
                }
            } else if (bIndex == -1) {
                c[i] = a[aIndex--];
            } else if (aIndex == -1) {
                c[i] = b[bIndex--];
            }
        }
        return c;
    }

    /**
     * in các phần tử của mảng ra màn hình
     *
     * @param arr: mảng các số nguyên
     * @param n: số phần tử của mảng
     */
    public static void show(float[] arr) {
        for (int i = 0; i < arr.length; i++) {
            System.out.print(arr[i] + " ");
        }
    }
}
```

Kết quả:

```
---Nhập mảng a---
Nhập số phần tử của mảng: 7
Nhập các phần tử của mảng:
a[0] = 2
a[1] = 3
a[2] = 4
a[3] = 5
a[4] = 6
a[5] = 7
a[6] = 8
---Nhập mảng b---
Nhập số phần tử của mảng: 5
Nhập các phần tử của mảng:
a[0] = 1
a[1] = 2
a[2] = 7
a[3] = 9
a[4] = 10
1.0 2.0 2.0 3.0 4.0 5.0 6.0 7.0 7.0 8.0 9.0 10.0
```

4. Bài tập về các thuật toán sắp xếp trong Java

Bài 1: Viết chương trình Java sắp xếp một dãy số theo thứ tự tăng dần bằng thuật toán nổi bọt (Bubble Sort).

Lời giải

Sắp xếp nổi bọt (Bubble Sort) là một giải thuật sắp xếp đơn giản. Giải thuật sắp xếp này được tiến hành dựa trên việc so sánh cặp phần tử liền kề nhau và tráo đổi thứ tự nếu chúng không theo thứ tự.

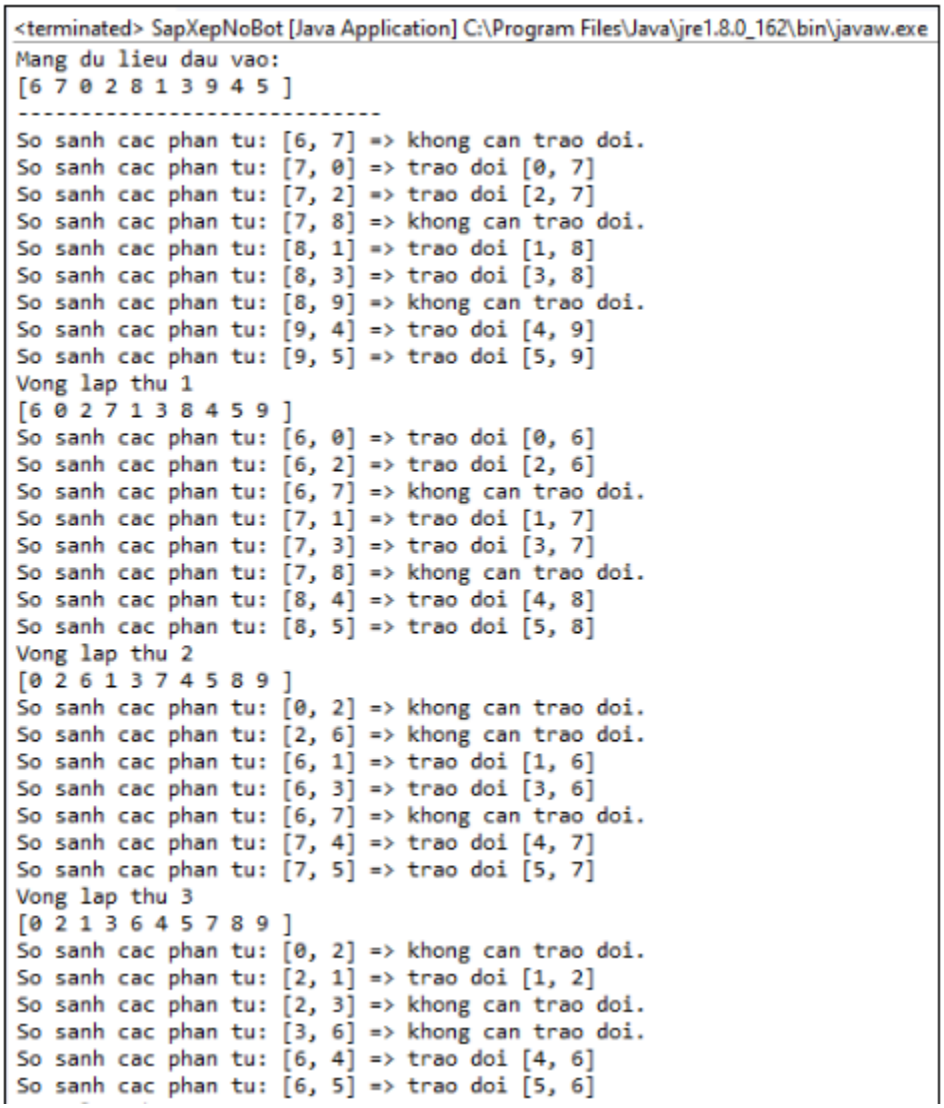
Giải thuật này không thích hợp sử dụng với các tập dữ liệu lớn khi mà độ phức tạp trường hợp xấu nhất và trường hợp trung bình là O(n2) với n là số phần tử.

Giải thuật sắp xếp nổi bọt là giải thuật chậm nhất trong số các giải thuật sắp xếp cơ bản. Giải thuật này còn chậm hơn giải thuật đổi chỗ trực tiếp mặc dù số lần so sánh bằng nhau, nhưng do đổi chỗ hai phần tử kề nhau nên số lần đổi chỗ nhiều hơn.

Dưới đây là chương trình Java để giải bài sắp xếp nổi bọt (Bubble Sort) trong Java:

```
    public void bubbleSort(int arr[]) {
        int temp;
        int i, j;

        boolean swapped = false;

        // lap qua tat ca cac so
        for (i = 0; i < arr.length - 1; i++) {
            swapped = false;

            // vong lap thu hai
            for (j = 0; j < arr.length - 1 - i; j++) {
                System.out.print("So sanh cac phan tu: [" + arr[j] + ", " + arr[j + 1] + "]");

                // kiem xa xem so ke tiep co nho hon so hien tai hay khong
                // trao doi cac so.
                // (Muc dich: lam noi bot (bubble) so lon nhat)
                if (arr[j] > arr[j + 1]) {
                    temp = arr[j];
                    arr[j] = arr[j + 1];
                    arr[j + 1] = temp;

                    swapped = true;
                    System.out.println(" => trao doi [" + arr[j] + ", " + arr[j + 1] + "]");
                } else {
                    System.out.println(" => khong can trao doi.");
                }
            }

            // neu khong can trao doi nua, tuc la
            // mang da duoc sap xep va thoat khoi vong lap.
            if (!swapped) {
                break;
            }

            System.out.println("Vong lap thu " + (i + 1));
            display(arr);

        }
    }

    public void display(int arr[]) {
        int i;
        System.out.print("[");

        // Duyet qua tat ca phan tu
        for (i = 0; i < arr.length; i++) {
            System.out.print(arr[i] + " ");
        }

        System.out.print("]\n");
    }

    public static void main(String[] args) {
        // khoi tao mang arr
        int arr[] = { 6, 7, 0, 2, 8, 1, 3, 9, 4, 5 };

        SapXepNoBot sapXepNoBot = new SapXepNoBot();
        System.out.println("Mang du lieu dau vao: ");
        sapXepNoBot.display(arr);
        System.out.println("----------------------------");
        sapXepNoBot.bubbleSort(arr);
        System.out.println("----------------------------");
        System.out.println("\nMang sau khi da sap xep: ");
        sapXepNoBot.display(arr);
    }
}
```

Chạy chương trình Java trên cho kết quả như sau:



```
<terminated> SapXepNoBot [Java Application] C:\Program Files\Java\jre1.8.0_162\bin\javaw.exe
Mang du lieu dau vao:
[6 7 0 2 8 1 3 9 4 5 ]
----------------------------
So sanh cac phan tu: [6, 7] => khong can trao doi.
So sanh cac phan tu: [7, 0] => trao doi [0, 7]
So sanh cac phan tu: [7, 2] => trao doi [2, 7]
So sanh cac phan tu: [7, 8] => khong can trao doi.
So sanh cac phan tu: [8, 1] => trao doi [1, 8]
So sanh cac phan tu: [8, 3] => trao doi [3, 8]
So sanh cac phan tu: [8, 9] => khong can trao doi.
So sanh cac phan tu: [9, 4] => trao doi [4, 9]
So sanh cac phan tu: [9, 5] => trao doi [5, 9]
Vong lap thu 1
[6 0 2 7 1 3 8 4 5 9 ]
So sanh cac phan tu: [6, 0] => trao doi [0, 6]
So sanh cac phan tu: [6, 2] => trao doi [2, 6]
So sanh cac phan tu: [6, 7] => khong can trao doi.
So sanh cac phan tu: [7, 1] => trao doi [1, 7]
So sanh cac phan tu: [7, 3] => trao doi [3, 7]
So sanh cac phan tu: [7, 8] => khong can trao doi.
So sanh cac phan tu: [8, 4] => trao doi [4, 8]
So sanh cac phan tu: [8, 5] => trao doi [5, 8]
Vong lap thu 2
[0 2 6 1 3 7 4 5 8 9 ]
So sanh cac phan tu: [0, 2] => khong can trao doi.
So sanh cac phan tu: [2, 6] => khong can trao doi.
So sanh cac phan tu: [6, 1] => trao doi [1, 6]
So sanh cac phan tu: [6, 3] => trao doi [3, 6]
So sanh cac phan tu: [6, 7] => khong can trao doi.
So sanh cac phan tu: [7, 4] => trao doi [4, 7]
So sanh cac phan tu: [7, 5] => trao doi [5, 7]
Vong lap thu 3
[0 2 1 3 6 4 5 7 8 9 ]
So sanh cac phan tu: [0, 2] => khong can trao doi.
So sanh cac phan tu: [2, 1] => trao doi [1, 2]
So sanh cac phan tu: [2, 3] => khong can trao doi.
So sanh cac phan tu: [3, 6] => khong can trao doi.
So sanh cac phan tu: [6, 4] => trao doi [4, 6]
So sanh cac phan tu: [6, 5] => trao doi [5, 6]
```

```
Vong lap thu 4
[0 1 2 3 4 5 6 7 8 9 ]
So sanh cac phan tu: [0, 1] => khong can trao doi.
So sanh cac phan tu: [1, 2] => khong can trao doi.
So sanh cac phan tu: [2, 3] => khong can trao doi.
So sanh cac phan tu: [3, 4] => khong can trao doi.
So sanh cac phan tu: [4, 5] => khong can trao doi.
----------------------------

Mang sau khi da sap xep:
[0 1 2 3 4 5 6 7 8 9 ]
```

Bài 2: Viết chương trình Java sắp xếp một dãy số theo thứ tự tăng dần bằng thuật toán chọn (Selection Sort).

Lời giải

Giải thuật sắp xếp chọn (Selection Sort) là một giải thuật đơn giản. Giải thuật sắp xếp này là một giải thuật dựa trên việc so sánh **in-place**, trong đó danh sách được chia thành hai phần, phần được sắp xếp (sorted list) ở bên trái và phần chưa được sắp xếp (unsorted list) ở bên phải. Ban đầu, phần được sắp xếp là trống và phần chưa được sắp xếp là toàn bộ danh sách ban đầu.

Phần tử nhỏ nhất được lựa chọn từ mảng chưa được sắp xếp và được tráo đổi với phần bên trái nhất và phần tử đó trở thành phần tử của mảng được sắp xếp. Tiến trình này tiếp tục cho tới khi toàn bộ từng phần tử trong mảng chưa được sắp xếp đều được di chuyển sang mảng đã được sắp xếp.

Dưới đây là chương trình Java để giải bài sắp xếp chọn (Selection Sort) trong Java:

```
public class SapXepChon {

    public void selectionSort(int arr[]) {
        int indexMin, i, j;

        // lap qua ta ca cac so
        for (i = 0; i < arr.length - 1; i++) {
            // thiet lap phan tu hien tai la min
            indexMin = i;

            // kiem tra phan tu hien tai co phai la nho nhat khong
            for (j = i + 1; j < arr.length; j++) {
                if (arr[j] < arr[indexMin]) {
                    indexMin = j;
                }
            }

            if (indexMin != i) {
                System.out.println(" ==> Trao doi phan tu: [" + arr[i]
                        + ", " + arr[indexMin] + "]");
                // Trao doi cac so
                int temp = arr[indexMin];
                arr[indexMin] = arr[i];
                arr[i] = temp;
            }

            System.out.println("Vong lap thu " + (i + 1));
            display(arr);
        }
    }
```

```
    public void display(int arr[]) {
        int i;
        System.out.print("[");

        // Duyet qua tat ca phan tu
        for (i = 0; i < arr.length; i++) {
            System.out.print(arr[i] + " ");
        }

        System.out.print("]\n");
    }

    public static void main(String[] args) {
        // khoi tao mang arr
        int arr[] = { 6, 7, 0, 2, 8, 1, 3, 9, 4, 5 };

        SapXepChon sapXepChon = new SapXepChon();
        System.out.println("Mang du lieu dau vao: ");
        sapXepChon.display(arr);
        System.out.println("------------------------------");
        sapXepChon.selectionSort(arr);
        System.out.println("------------------------------");
        System.out.println("\nMang sau khi da sap xep: ");
        sapXepChon.display(arr);
    }
}
```

Chạy chương trình Java trên cho kết quả như sau:

```
<terminated> SapXepChon [Java Application] C:\Program Files\Java\jre1.8.0_162\bin\javaw.exe
Mang du lieu dau vao:
[6 7 0 2 8 1 3 9 4 5 ]
----------------------------
 ==> Trao doi phan tu: [6, 0]
Vong lap thu 1
[0 7 6 2 8 1 3 9 4 5 ]
 ==> Trao doi phan tu: [7, 1]
Vong lap thu 2
[0 1 6 2 8 7 3 9 4 5 ]
 ==> Trao doi phan tu: [6, 2]
Vong lap thu 3
[0 1 2 6 8 7 3 9 4 5 ]
 ==> Trao doi phan tu: [6, 3]
Vong lap thu 4
[0 1 2 3 8 7 6 9 4 5 ]
 ==> Trao doi phan tu: [8, 4]
Vong lap thu 5
[0 1 2 3 4 7 6 9 8 5 ]
 ==> Trao doi phan tu: [7, 5]
Vong lap thu 6
[0 1 2 3 4 5 6 9 8 7 ]
Vong lap thu 7
[0 1 2 3 4 5 6 9 8 7 ]
 ==> Trao doi phan tu: [9, 7]
Vong lap thu 8
[0 1 2 3 4 5 6 7 8 9 ]
Vong lap thu 9
[0 1 2 3 4 5 6 7 8 9 ]
----------------------------

Mang sau khi da sap xep:
[0 1 2 3 4 5 6 7 8 9 ]
```

Bài 3: Viết chương trình Java sắp xếp một dãy số theo thứ tự tăng dần bằng thuật toán chèn (Insertion Sort).

Lời giải

Sắp xếp chèn là một giải thuật sắp xếp dựa trên so sánh in-place. Ở đây, một danh sách con luôn luôn được duy trì dưới dạng đã qua sắp xếp. Sắp xếp chèn là chèn thêm một phần tử vào danh sách con đã qua sắp xếp. Phần tử được chèn vào vị trí thích hợp sao cho vẫn đảm bảo rằng danh sách con đó vẫn sắp theo thứ tự.

Với cấu trúc dữ liệu mảng, chúng ta tưởng tượng là: mảng gồm hai phần: một danh sách con đã được sắp xếp và phần khác là các phần tử không có thứ tự. Giải thuật sắp xếp chèn sẽ thực hiện việc tìm kiếm liên tiếp qua mảng đó, và các phần tử không có thứ tự sẽ được di chuyển và được chèn vào vị trí thích hợp trong danh sách con (của cùng mảng đó).

Giải thuật này không thích hợp sử dụng với các tập dữ liệu lớn khi độ phức tạp trường hợp xấu nhất và trường hợp trung bình là O(n2) với n là số phần tử.

Dưới đây là chương trình Java để giải bài sắp xếp chèn (Insertion Sort) trong Java:

```java
public class SapXepChen {

    public void insertionSort(int arr[]) {
        int valueToInsert;
        int holePosition;
        int i;

        // lap qua tat ca cac so
        for (i = 1; i < arr.length; i++) {

            // chon mot gia tri de chen
            valueToInsert = arr[i];

            // lua chon vi tri de chen
            holePosition = i;

            // kiem tra xem so lien truoc co lon hon gia tri duoc chen khong
            while (holePosition > 0 && arr[holePosition - 1] > valueToInsert) {
                arr[holePosition] = arr[holePosition - 1];
                holePosition--;
                System.out.println("Di chuyen phan tu: " + arr[holePosition]);
            }

            if (holePosition != i) {
                System.out.println(" Chen phan tu: " + valueToInsert
                        + ", tai vi tri: " + holePosition);
                // chen phan tu tai vi tri chen
                arr[holePosition] = valueToInsert;
            }

            System.out.println("Vong lap thu " + i);
            display(arr);
        }
    }

    public void display(int arr[]) {
        int i;
        System.out.print("[");

        // Duyet qua tat ca phan tu
        for (i = 0; i < arr.length; i++) {
            System.out.print(arr[i] + " ");
        }

        System.out.print("]\n");
    }

    public static void main(String[] args) {
        // khoi tao mang arr
        int arr[] = { 6, 7, 0, 2, 8, 1, 3, 9, 4, 5 };

        SapXepChen sapXepChen = new SapXepChen();
        System.out.println("Mang du lieu dau vao: ");
        sapXepChen.display(arr);
        System.out.println("----------------------------");
        sapXepChen.insertionSort(arr);
        System.out.println("----------------------------");
        System.out.println("\nMang sau khi da sap xep: ");
        sapXepChen.display(arr);
    }
}
```

Bài 3: Viết chương trình Java sắp xếp một dãy số theo thứ tự tăng dần bằng thuật toán nhanh (Quick Sort).

Lời giải

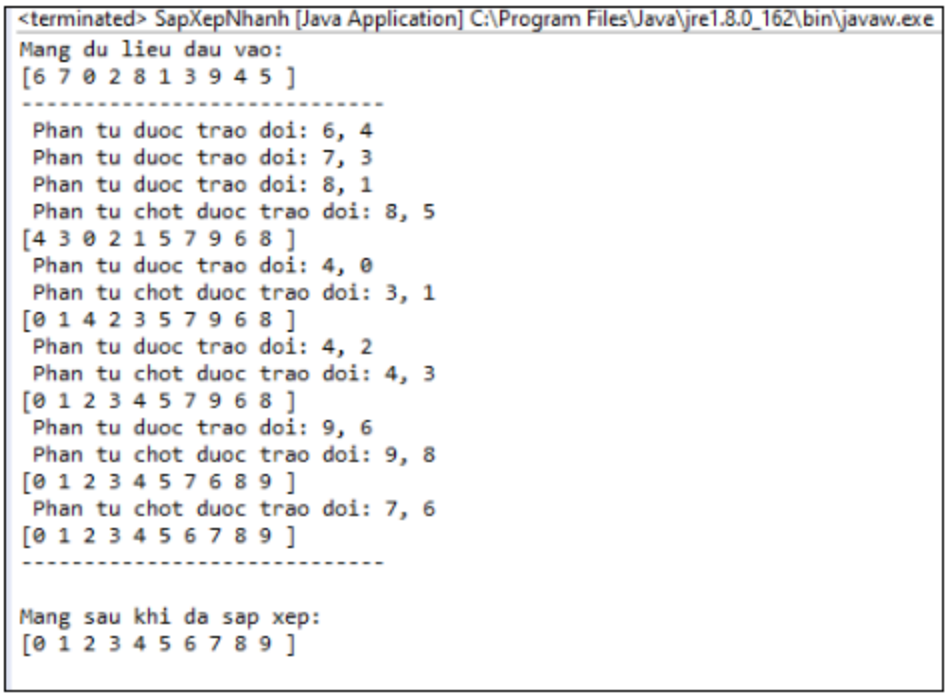
Giải thuật sắp xếp nhanh (Quick Sort) là một giải thuật hiệu quả cao và dựa trên việc chia mảng dữa liệu thành các mảng nhỏ hơn. Giải thuật sắp xếp nhanh chia mảng thành hai phần bằng cách so sánh từng phần tử của mảng với một phần tử được chọn gọi là **phần tử chốt (Pivot)**: một mảng bao gồm các phần tử nhỏ hơn hoặc bằng phần tử chốt và mảng còn lại bao gồm các phần tử lớn hơn hoặc bằng phần tử chốt.

Dưới đây là chương trình Java để giải bài sắp xếp nhanh (Quick Sort) trong Java:

```java
public class SapXepNhanh {

    // ham de trao doi gia tri
    public void swap(int arr[], int num1, int num2) {
        int temp = arr[num1];
        arr[num1] = arr[num2];
        arr[num2] = temp;
    }

    // ham de chia mang thanh 2 phan, su dung phan tu chot (pivot)
    public int partition(int arr[], int left, int right, int pivot) {
        int leftPointer = left - 1;
        int rightPointer = right;

        while (true) {

            while (arr[++leftPointer] < pivot) {
                // khong lam gi
            }

            while (rightPointer > 0 && arr[--rightPointer] > pivot) {
                // khong lam gi
            }

            if (leftPointer >= rightPointer) {
                break;
            } else {
                System.out.println(" Phan tu duoc trao doi: " + arr[leftPointer]
                        + ", " + arr[rightPointer]);
                swap(arr, leftPointer, rightPointer);
            }

        }
```

```
        System.out.println(" Phan tu chot duoc trao doi: " + arr[leftPointer]
                + ", " + arr[right]);
        swap(arr, leftPointer, right);
        display(arr);
        return leftPointer;
    }

    // ham sap xep
    public void quickSort(int arr[], int left, int right) {
        if (right - left <= 0) {
            return;
        } else {
            int pivot = arr[right];
            int partitionPoint = partition(arr, left, right, pivot);
            quickSort(arr, left, partitionPoint - 1);
            quickSort(arr, partitionPoint + 1, right);
        }
    }

    public void display(int arr[]) {
        int i;
        System.out.print("[");

        // Duyet qua tat ca phan tu
        for (i = 0; i < arr.length; i++) {
            System.out.print(arr[i] + " ");
        }

        System.out.print("]\n");
    }

    public static void main(String[] args) {
        // khoi tao mang arr
        int arr[] = { 6, 7, 0, 2, 8, 1, 3, 9, 4, 5 };

        SapXepNhanh sapXepNhanh = new SapXepNhanh();
        System.out.println("Mang du lieu dau vao: ");
        sapXepNhanh.display(arr);
        System.out.println("------------------------------");
        sapXepNhanh.quickSort(arr, 0, arr.length - 1);
        System.out.println("------------------------------");
        System.out.println("\nMang sau khi da sap xep: ");
        sapXepNhanh.display(arr);
    }
}
```

Javahạy chương trình Java trên cho kết quả như sau:



Bài 3: Viết chương trình Java sắp xếp một dãy số theo thứ tự tăng dần bằng thuật toán trộn (Merge Sort)

Lời giải

Sắp xếp trộn (Merge Sort) là một giải thuật sắp xếp dựa trên giải thuật **Chia để trị (Divide and Javaonquer)**. Với độ phức tạp thời gian trường hợp xấu nhất là O(n log n) thì đây là một trong các giải thuật đáng được quan tâm nhất.

Đầu tiên, giải thuật sắp xếp trộn chia mảng thành hai nửa và sau đó kết hợp chúng lại với nhau thành một mảng đã được sắp xếp.

Dưới đây là chương trình Java để giải bài sắp xếp trộn (Merge Sort) trong Java:

```java
public class SapXepTron {

    public void merging(int arr[], int temp[], int low, int mid, int high) {
        int l1, l2, i;

        l1 = low;
        l2 = mid + 1;
        for (i = low; l1 <= mid && l2 <= high; i++) {
            if (arr[l1] <= arr[l2]) {
                temp[i] = arr[l1++];
            } else {
                temp[i] = arr[l2++];
            }
        }

        while (l1 <= mid) {
            temp[i++] = arr[l1++];
        }
        while (l2 <= high) {
            temp[i++] = arr[l2++];
        }
        for (i = low; i <= high; i++) {
            arr[i] = temp[i];
        }
    }

    public void sort(int arr[], int temp[], int low, int high) {
        int mid;

        if (low < high) {
            mid = (low + high) / 2;
            sort(arr, temp, low, mid);
            sort(arr, temp, mid + 1, high);
            merging(arr, temp, low, mid, high);
            // hien thi mang
            display(arr);
        } else {
            return;
        }
    }
```

```
    public void display(int arr[]) {
        int i;
        System.out.print("[");

        // Duyet qua tat ca phan tu
        for (i = 0; i < arr.length; i++) {
            System.out.print(arr[i] + " ");
        }

        System.out.print("]\n");
    }

    public static void main(String[] args) {
        // khoi tao mang arr
        int arr[] = { 6, 7, 0, 2, 8, 1, 3, 9, 4, 5 };
        int temp[] = new int[10];

        SapXepTron sapXeptron = new SapXepTron();
        System.out.println("Mang du lieu dau vao: ");
        sapXeptron.display(arr);
        System.out.println("-----------------------------");
        sapXeptron.sort(arr, temp, 0, arr.length - 1);
        System.out.println("-----------------------------");
        System.out.println("\nMang sau khi da sap xep: ");
        sapXeptron.display(arr);
    }
}
```

Java chạy chương trình Java trên cho kết quả như sau:

```
<terminated> SapXepTron [Java Application] C:\Program Files\Java\jre1.8.0_162\bin\javaw.exe
Mang du lieu dau vao:
[6 7 0 2 8 1 3 9 4 5 ]
-----------------------------
[6 7 0 2 8 1 3 9 4 5 ]
[0 6 7 2 8 1 3 9 4 5 ]
[0 6 7 2 8 1 3 9 4 5 ]
[0 2 6 7 8 1 3 9 4 5 ]
[0 2 6 7 8 1 3 9 4 5 ]
[0 2 6 7 8 1 3 9 4 5 ]
[0 2 6 7 8 1 3 9 4 5 ]
[0 2 6 7 8 1 3 4 5 9 ]
[0 1 2 3 4 5 6 7 8 9 ]
-----------------------------

Mang sau khi da sap xep:
[0 1 2 3 4 5 6 7 8 9 ]
```