

BÀI TẬP THỰC HÀNH JAVA CĂN BẢN

Nội dung chính

- **Bài tập Java**
- [1. Bài tập java cơ bản](#)
- [2. Bài tập chuỗi trong Java](#)
- [3. Bài tập mảng trong Java](#)
- [4. Bài tập về các thuật toán sắp xếp trong Java](#)
- [5. Bài tập lập trình hướng đối tượng trong java](#)
- [Bài tập quản lý sinh viên trong Java - console](#)

Bài tập Java

Bài này cung cấp cho bạn danh sách các dạng bài tập khác nhau để bạn thực hành khi học java.

Note: Trước khi xem lời giải thì các bạn hãy tự làm trước nhé và hãy common hóa (phân tách thành phương thức riêng) những gì có thể sử dụng lại được.

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:

```

1  package vn.vietttuts.baitap;
2
3  import java.util.ArrayList;
4  import java.util.List;
5
6  public class Bai01 {
7      public static void main(String[] args) {
8          List<Integer> list = new ArrayList<Integer>();
9          for (int i = 10; i < 201; i++) {
10             if ((i % 7 == 0) && (i % 5 != 0)) {
11                 list.add(i);
12             }
13         }
14         // hiển thị list ra màn hình
15         showList(list);
16     }
17
18     public static void showList(List<Integer> list) {
19         if (list != null && !list.isEmpty()) {
20             int size = list.size();
21             for (int i = 0; i < size - 1; i++) {
22                 System.out.print(list.get(i) + ", ");
23             }
24             System.out.println(list.get(size - 1));
25         }
26     }
27 }

```

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

```

1  package vn.vietttuts.baitap;
2
3  import java.util.Scanner;

```

```

4
5     public class GiaiThuaDemo2 {
6         private static Scanner scanner = new Scanner(System.in);
7         /**
8          * main
9          *
10         * @author plpsoft.vn
11         * @param args
12         */
13         public static void main(String[] args) {
14             System.out.print("Nhập số nguyên dương n = ");
15             int n = scanner.nextInt();
16             System.out.println("Giai thừa của " + n + " là: " + tinhGiaiThua(n));
17         }
18
19         /**
20          * tính giai thừa
21          *
22          * @author plpsoft.vn
23          * @param n: số nguyên dương
24          * @return giai thừa của số n
25          */
26         public static long tinhGiaiThua(int n) {
27             if (n > 0) {
28                 return n * tinhGiaiThua(n - 1);
29             } else {
30                 return 1;
31             }
32         }
33     }

```

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:

```

1     package vn.viettuts.baitap;
2
3     import java.util.HashMap;

```

```

4    import java.util.Map;
5    import java.util.Scanner;
6
7    public class Bai03 {
8        private static Scanner scanner = new Scanner(System.in);
9
10       public static void main(String[] args) {
11           System.out.print("Nhập số nguyên dương n = ");
12           int n = scanner.nextInt();
13
14           Map<Integer, Integer> map = new HashMap<Integer, Integer>();
15           for (int i = 1; i < n + 1; i++) {
16               map.put(i, i * i);
17           }
18           System.out.println(map);
19       }
20   }

```

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: $ax^2 + bx + c = 0$.

Code mẫu:

```

1    package vn.vietttuts.baitap;
2
3    import java.util.Scanner;
4
5    /**
6     * Giải phương trình bậc 2
7     *
8     * @author plpsoft.vn
9     */
10   public class PhuongTrinhBac2 {
11       private static Scanner scanner = new Scanner(System.in);
12       /**
13        * main
14        *
15        * @param args
16        */
17       public static void main(String[] args) {
18           System.out.print("Nhập hệ số bậc 2, a = ");
19           float a = scanner.nextFloat();
20           System.out.print("Nhập hệ số bậc 1, b = ");
21           float b = scanner.nextFloat();
22           System.out.print("Nhập hằng số tự do, c = ");

```

```

23         float c = scanner.nextFloat();
24         giaiPTBac2(a, b, c);
25     }
26
27     /**
28     * Giải phương trình bậc 2:  $ax^2 + bx + c = 0$ 
29     *
30     * @param a: hệ số bậc 2
31     * @param b: hệ số bậc 1
32     * @param c: số hạng tự do
33     */
34     public static void giaiPTBac2(float a, float b, float c) {
35         // kiểm tra các hệ số
36         if (a == 0) {
37             if (b == 0) {
38                 System.out.println("Phương trình vô nghiệm!");
39             } else {
40                 System.out.println("Phương trình có một nghiệm: "
41                     + "x = " + (-c / b));
42             }
43             return;
44         }
45         // tính delta
46         float delta = b*b - 4*a*c;
47         float x1;
48         float x2;
49         // tính nghiệm
50         if (delta > 0) {
51             x1 = (float) ((-b + Math.sqrt(delta)) / (2*a));
52             x2 = (float) ((-b - Math.sqrt(delta)) / (2*a));
53             System.out.println("Phương trình có 2 nghiệm là: "
54                 + "x1 = " + x1 + " và x2 = " + x2);
55         } else if (delta == 0) {
56             x1 = (-b / (2 * a));
57             System.out.println("Phương trình có nghiệm kép: "
58                 + "x1 = x2 = " + x1);
59         } else {
60             System.out.println("Phương trình vô nghiệm!");
61         }
62     }
63 }

```

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 ($1 \leq B \leq 32$) bất kỳ. Giả sử hệ cơ số cần chuyển là $2 \leq B \leq 16$. 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:

```

1  package vn.viettuts.baitap;
2
3  import java.util.Scanner;
4
5  public class ConvertNumber {
6      public static final char CHAR_55 = 55;
7      private static Scanner scanner = new Scanner(System.in);
8
9      /**
10       * main
11       *
12       * @author plpsoft.vn
13       * @param args
14       */
15     public static void main(String[] args) {
16         System.out.print("Nhập số nguyên dương n = ");
17         int n = scanner.nextInt();
18         System.out.println("Số " + n + " trong hệ cơ số 2 = "
19             + ConvertNumber.convertNumber(n, 2));
20         System.out.println("Số " + n + " trong hệ cơ số 16 = "
21             + ConvertNumber.convertNumber(n, 16));
22     }
23
24     /**
25      * chuyển đổi số nguyên n sang hệ cơ số b
26      *
27      * @author plpsoft.vn
28      * @param n: số nguyên
29      * @param b: hệ cơ số
30      * @return hệ cơ số b
31      */
32     public static String convertNumber(int n, int b) {
33         if (n < 0 || b < 2 || b > 16) {
34             return "";
35         }

```

```

36
37     StringBuilder sb = new StringBuilder();
38     int m;
39     int remainder = n;
40
41     while (remainder > 0) {
42         if (b > 10) {
43             m = remainder % b;
44             if (m >= 10) {
45                 sb.append((char) (CHAR_55 + m));
46             } else {
47                 sb.append(m);
48             }
49         } else {
50             sb.append(remainder % b);
51         }
52         remainder = remainder / b;
53     }
54     return sb.reverse().toString();
55 }
56 }

```

Kết quả:

```

Nhập số nguyên dương n = 15
Số 15 trong hệ cơ số 2 = 1111
Số 15 trong hệ cơ số 16 = F

```

Bài 06:

Dãy số Fibonacci được định nghĩa như sau: $F_0 = 0$, $F_1 = 1$, $F_2 = 1$, $F_n = F_{n-1} + F_{n-2}$ với $n \geq 2$. Ví dụ: 0, 1, 1, 2, 3, 5, 8, ... Hãy viết chương trình tìm n số Fibonacci đầu tiên.

Code mẫu:

```

1     package vn.viettuts.baitap;
2
3     import java.util.Scanner;
4
5     /**
6      * Tính dãy số Fibonacci bằng phương pháp đệ quy
7      *
8      * @author plpsoft.vn
9      */
10    public class FibonacciExample2 {
11        private static Scanner scanner = new Scanner(System.in);
12        /**
13         * main
14         *
15         * @param args

```

```

15      */
16      public static void main(String[] args) {
17          System.out.print("Nhập số nguyên dương n = ");
18          int n = scanner.nextInt();
19          System.out.println(n + " số đầu tiên của dãy số fibonacci: ");
20          for (int i = 0; i < n; i++) {
21              System.out.print(fibonacci(i) + " ");
22          }
23      }
24
25      /**
26       * Tính số fibonacci thứ n
27       *
28       * @param n: chỉ số của số fibonacci tính từ 0
29       *          vd: F0 = 0, F1 = 1, F2 = 1, F3 = 2
30       * @return số fibonacci thứ n
31       */
32      public static int fibonacci(int n) {
33          if (n < 0) {
34              return -1;
35          } else if (n == 0 || n == 1) {
36              return n;
37          } else {
38              return fibonacci(n - 1) + fibonacci(n - 2);
39          }
40      }
41  }
42

```

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:

```

1      package vn.viettuts.baitap;
2
3      import java.util.Scanner;
4

```



```

5    public class USCLL_BSCNN_1 {
6        private static Scanner scanner = new Scanner(System.in);
7        /**
8         * main
9         *
10        * @param args
11        */
12        public static void main(String[] args) {
13            System.out.print("Nhập số nguyên dương a = ");
14            int a = scanner.nextInt();
15            System.out.print("Nhập số nguyên dương b = ");
16            int b = scanner.nextInt();
17            // tính USCLN của a và b
18            System.out.println("USCLN của " + a + " và " + b
19                               + " là: " + USCLN(a, b));
20            // tính BSCNN của a và b
21            System.out.println("BSCNN của " + a + " và " + b
22                               + " là: " + BSCNN(a, b));
23        }
24
25        /**
26         * Tìm ước số chung lớn nhất (USCLN)
27         *
28         * @param a: số nguyên dương
29         * @param b: số nguyên dương
30         * @return USCLN của a và b
31         */
32        public static int USCLN(int a, int b) {
33            if (b == 0) return a;
34            return USCLN(b, a % b);
35        }
36
37        /**
38         * Tìm bội số chung nhỏ nhất (BSCNN)
39         *
40         * @param a: số nguyên dương
41         * @param b: số nguyên dương
42         * @return BSCNN của a và b
43         */
44        public static int BSCNN(int a, int b) {
45            return (a * b) / USCLN(a, b);
46        }
47    }

```

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

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:

```

1  package vn.viettuts.baitap;
2
3  import java.util.Scanner;
4
5  /**
6   * Chương trình liệt kê tất cả các số nguyên tố nhỏ hơn n.
7   *
8   * @author plpsoft.vn
9   */
10 public class BaiTap08 {
11     private static Scanner scanner = new Scanner(System.in);
12
13     /**
14      * main
15      *
16      * @param args
17      */
18     public static void main(String[] args) {
19         System.out.print("Nhập n = ");
20         int n = scanner.nextInt();
21         System.out.printf("Tất cả các số nguyên tố nhỏ hơn %d là:
22 \n", n);
23         if (n >= 2) {
24             System.out.print(2);
25         }
26         for (int i = 3; i < n; i+=2) {
27             if (isPrimeNumber(i)) {
28                 System.out.print(" " + i);
29             }
30         }
31
32         /**
33          * check so nguyen to
34          *
35          * @author plpsoft.vn
36          * @param n: so nguyen duong
37          * @return true la so nguyen so,
38          *         false khong la so nguyen to
39          */
40         public static boolean isPrimeNumber(int n) {
41             // so nguyen n < 2 khong phai la so nguyen to
42             if (n < 2) {
43                 return false;
44             }
45             // check so nguyen to khi n >= 2
46             int squareRoot = (int) Math.sqrt(n);
47             for (int i = 2; i <= squareRoot; i++) {

```

```

48         if (n % i == 0) {
49             return false;
50         }
51     }
52     return true;
53 }
54 }

```

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:

```

1    package vn.viettuts.baitap;
2
3    import java.util.Scanner;
4
5    /**
6     * Chương trình liệt kê n số nguyên tố đầu tiên.
7     *
8     * @author plpsoft.vn
9     */
10   public class BaiTap09 {
11       private static Scanner scanner = new Scanner(System.in);
12
13       /**
14        * main
15        *
16        * @param args
17        */
18       public static void main(String[] args) {
19           System.out.print("Nhập n = ");
20           int n = scanner.nextInt();
21           System.out.printf("%d số nguyên tố đầu tiên là: \n", n);
22           int dem = 0; // đếm số số nguyên tố
23           int i = 2;   // tìm số nguyên tố bắt đầu từ số 2
24           while (dem < n) {
25               if (isPrimeNumber(i)) {
26                   System.out.print(i + " ");
27                   dem++;
28               }
29               i++;
30           }

```

```

31     }
32
33     /**
34     * check so nguyen to
35     *
36     * @author plpsoft.vn
37     * @param n: so nguyen duong
38     * @return true la so nguyen so,
39     *         false khong la so nguyen to
40     */
41     public static boolean isPrimeNumber(int n) {
42         // so nguyen n < 2 khong phai la so nguyen to
43         if (n < 2) {
44             return false;
45         }
46         // check so nguyen to khi n >= 2
47         int squareRoot = (int) Math.sqrt(n);
48         for (int i = 2; i <= squareRoot; i++) {
49             if (n % i == 0) {
50                 return false;
51             }
52         }
53         return true;
54     }
55 }

```

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:

```

1     package vn.viettuts.baitap;
2
3     /**
4     * Chương trình liệt kê tất cả số nguyên tố có 5 chữ số.
5     *
6     * @author plpsoft.vn
7     */
8     public class BaiTap10 {
9
10        /**
11        * main
12        *
13        * @param args
14        */

```

```

15     public static void main(String[] args) {
16         int count = 0;
17         System.out.println("Liệt kê tất cả số nguyên tố có 5 chữ số:");
18         for (int i = 10001; i < 99999; i+=2) {
19             if (isPrimeNumber(i)) {
20                 System.out.println(i);
21                 count++;
22             }
23         }
24         System.out.println("Tổng các số nguyên tố có 5 chữ số là: " + count);
25     }
26
27     /**
28     * check so nguyen to
29     *
30     * @author plpsoft.vn
31     * @param n: so nguyen duong
32     * @return true la so nguyen so,
33     *         false khong la so nguyen to
34     */
35     public static boolean isPrimeNumber(int n) {
36         // so nguyen n < 2 khong phai la so nguyen to
37         if (n < 2) {
38             return false;
39         }
40         // check so nguyen to khi n >= 2
41         int squareRoot = (int) Math.sqrt(n);
42         for (int i = 2; i <= squareRoot; i++) {
43             if (n % i == 0) {
44                 return false;
45             }
46         }
47         return true;
48     }
49 }

```

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: $100 = 2 \times 2 \times 5 \times 5$.

Code mẫu:

```

1    package vn.viettuts.baitap;
2
3    import java.util.ArrayList;
4    import java.util.List;
5    import java.util.Scanner;
6
7    /**
8     * Chương trình phân tích số nguyên n thành các thừa số nguyên tố.
9     * Ví dụ: 12 = 2 x 2 x 3.
10   *
11   * @author plpsoft.vn
12   */
13   public class BaiTap11 {
14       private static Scanner scanner = new Scanner(System.in);
15
16       /**
17        * main
18        *
19        * @param args
20        */
21       public static void main(String[] args) {
22           System.out.print("Nhập số nguyên dương n = ");
23           int n = scanner.nextInt();
24           // phân tích số nguyên dương n
25           List<Integer> listNumbers = phanTichSoNguyen(n);
26           // in kết quả ra màn hình
27           System.out.printf("Kết quả: %d = ", n);
28           int size = listNumbers.size();
29           for (int i = 0; i < size - 1; i++) {
30               System.out.print(listNumbers.get(i) + " x ");
31           }
32           System.out.print(listNumbers.get(size - 1));
33       }
34
35       /**
36        * Phân tích số nguyên thành tích các thừa số nguyên tố
37        *
38        * @param positiveInt
39        * @return
40        */
41       public static List<Integer> phanTichSoNguyen(int n) {
42           int i = 2;
43           List<Integer> listNumbers = new ArrayList<Integer>();
44           // phân tích
45           while (n > 1) {
46               if (n % i == 0) {
47                   n = n / i;
48                   listNumbers.add(i);
49               } else {
50                   i++;
51               }
52           }
53           // nếu listNumbers trống thì add n vào listNumbers
54           if (listNumbers.isEmpty()) {
55               listNumbers.add(n);
56           }
57           return listNumbers;
58       }
59   }

```

```
57     }
58 }
59
```

Kết quả:

Nhập số nguyên dương $n = 100$

Kết quả: $100 = 2 \times 2 \times 5 \times 5$

Bài 12:

Viết chương trình tính tổng của các chữ số của một 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:

```
1  package vn.viettuts.baitap;
2
3  import java.util.Scanner;
4
5  /**
6   * Chương trình tính tổng của các chữ số của một số nguyên dương n.
7   * Tổng của các chữ số của 6677 là  $6 + 6 + 7 + 7 = 26$ .
8   *
9   * @author plpsoft.vn
10  */
11  public class BaiTap12 {
12      private static Scanner scanner = new Scanner(System.in);
13      public static int DEC_10 = 10;
14
15      /**
16       * main
17       *
18       * @param args
19       */
20      public static void main(String[] args) {
21          System.out.print("Nhập số nguyên dương n = ");
22          int n = scanner.nextInt();
23          System.out.printf("Tổng của các chữ số "
24                          + "của %d là: %d", n, totalDigitsOfNumber(n));
25      }
26
27      /**
28       * Tính tổng của các chữ số của một số nguyên dương
29       *
30       * @param n: số nguyên dương
31       * @return
32       */
33      public static int totalDigitsOfNumber(int n) {
34          int total = 0;
```

```

35         do {
36             total = total + n % DEC_10;
37             n = n / DEC_10;
38         } while (n > 0);
39         return total;
40     }
41 }

```

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:

```

1     package vn.viettuts.baitap;
2
3     import java.util.Scanner;
4
5     /**
6      * Chương trình liệt kê tất cả các số thuận nghịch có 6 chữ số.
7      *
8      * @author plpsoft.vn
9      */
10    public class BaiTap13 {
11        private static Scanner scanner = new Scanner(System.in);
12        /**
13         * main
14         *
15         * @param args
16         */
17        public static void main(String[] args) {
18            System.out.print("Nhập số nguyên dương n = ");
19            int n = scanner.nextInt();
20            System.out.println(n + " là số thuận nghịch: " + isThuanNghich(n));
21            System.out.print("Nhập số nguyên dương m = ");
22            int m = scanner.nextInt();
23            System.out.println(m + " là số thuận nghịch: " + isThuanNghich(m));
24        }
25        /**
26         * Kiểm tra số thuận nghịch
27         *
28         * @param n: số nguyên dương
29         * @return true là số thuận nghịch
30         *         false không là số thuận nghịch
31         */
32        public static boolean isThuanNghich(int n) {
33            // chuyển đổi số n thành một chuỗi String
34            String numberStr = String.valueOf(n);

```



```

35         // kiểm tra tính thuận nghịch
36         int size = numberStr.length();
37         for (int i = 0; i < (size/2); i++) {
38             if (numberStr.charAt(i) != numberStr.charAt(size - i - 1)) {
39                 return false;
40             }
41         }
42         return true;
43     }
44 }
45

```

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:

```

1     package vn.viettuts.baitap;
2
3     import java.util.Scanner;
4
5     /**
6      * Chương trình liệt kê các số Fibonacci nhỏ hơn n là số nguyên tố.
7      * Với n được nhập từ bàn phím.
8      *
9      * @author plpsoft.vn
10    */
11    public class BaiTap14 {
12        private static Scanner scanner = new Scanner(System.in);
13        /**
14         * main
15         *
16         * @param args
17         */
18        public static void main(String[] args) {
19            System.out.print("Nhập số tự nhiên n = ");
20            int n = scanner.nextInt();
21            System.out.printf("Các số fibonacci nhỏ hơn %d và "
22                + "là số nguyên tố: ", n);
23            int i = 0;
24            while (fibonacci(i) < 100) {
25                int fi = fibonacci(i);
26                if (isPrimeNumber(fi)) {
27                    System.out.print(fi + " ");
28                }

```

```

29         i++;
30     }
31 }
32
33 /**
34  * Tính số fibonacci thứ n
35  *
36  * @param n: chỉ số của số fibonacci tính từ 0
37  *          vd: F0 = 0, F1 = 1, F2 = 1, F3 = 2
38  * @return số fibonacci thứ n
39  */
40 public static int fibonacci(int n) {
41     if (n < 0) {
42         return -1;
43     } else if (n == 0 || n == 1) {
44         return n;
45     } else {
46         return fibonacci(n - 1) + fibonacci(n - 2);
47     }
48 }
49
50 /**
51  * check số nguyên tố
52  *
53  * @author plpsoft.vn
54  * @param n: số nguyên dương
55  * @return true là số nguyên tố,
56  *         false không là số nguyên tố
57  */
58 public static boolean isPrimeNumber(int n) {
59     // số nguyên n < 2 không phải là số nguyên tố
60     if (n < 2) {
61         return false;
62     }
63     // check số nguyên tố khi n >= 2
64     int squareRoot = (int) Math.sqrt(n);
65     for (int i = 2; i <= squareRoot; i++) {
66         if (n % i == 0) {
67             return false;
68         }
69     }
70     return true;
71 }
72 }

```

Kết quả:

Nhập số tự nhiên n = 100

Các số fibonacci nhỏ hơn 100 và là số nguyên tố: 2 3 5 13 89

Các bài tập khác:

1. Viết chương trình nhập số nguyên dương n và thực hiện các chức năng sau:
 - a) Tính tổng các chữ số của n.
 - b) Phân tích n thành tích các thừa số nguyên tố.

- c) Liệt kê các ước số của n .
 - d) Liệt kê các ước số là nguyên tố của n .
2. Viết chương trình liệt kê các số nguyên có từ 5 đến 7 chữ số thỏa mãn:
- a) Là số nguyên tố.
 - b) Là số thuận nghịch.
 - c) Mỗi chữ số đều là số nguyên tố.
 - d) Tổng các chữ số là số nguyên tố.
3. Viết chương trình liệt kê các số nguyên có 7 chữ số thỏa mãn:
- a) Là số nguyên tố.
 - b) Là số thuận nghịch.
 - c) Mỗi chữ số đều là số nguyên tố.
 - d) Tổng các chữ số là số thuận nghịch.
-

2. Bài tập chuỗi trong Java

Danh sách bài tập:

16. Nhập một sô ký tự. Đếm số từ của sô 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: [Đếm số từ trong một chuỗi.](#)

17. Nhập một sô ký tự. Liệt kê số lần xuất hiện của các từ của sô đó.

Lời giải: [Liệt kê số lần xuất hiện của các từ trong một chuỗi.](#)

18. Nhập 2 sô ký tự s1 và s2. Kiểm tra xem sô s1 có chứa s2 không?

Lời giải: [Chuỗi chứa chuỗi trong java.](#)

3. Bài tập mảng trong Java

Các bài tập trong phần này thao tác với mảng một chiều và 2 chiều trong java, bạn có thể tham khảo bài học [mảng \(Array\) trong java](#)

Danh sách bài tập:

19. Nhập một mảng số nguyên $a_0, a_1, a_2, \dots, a_{n-1}$. In ra màn hình các phần tử xuất hiện trong mảng đúng 1 lần.

Lời giải: [Liệt kê các phần tử xuất hiện trong mảng đúng 1 lần](#)

20. Nhập một mảng số nguyên $a_0, a_1, a_2, \dots, a_{n-1}$. In ra màn hình các phần tử xuất hiện trong mảng đúng 2 lần.

Lời giải: [Liệt kê các phần tử xuất hiện trong mảng đúng 2 lần](#)

21. Nhập một mảng số nguyên $a_0, a_1, a_2, \dots, a_{n-1}$. In ra màn hình số lần xuất hiện của các phần tử.

Lời giải: [Liệt kê số lần xuất hiện của các phần tử trong một mảng](#)

22. Nhập một mảng số nguyên $a_0, a_1, a_2, \dots, a_{n-1}$. Hãy sắp xếp mảng theo thứ tự tăng dần.

Lời giải: [Sắp xếp mảng theo thứ tự tăng dần](#)

23. Nhập một mảng số nguyên $a_0, a_1, a_2, \dots, a_{n-1}$. Hãy sắp xếp mảng theo thứ tự giảm dần.

Lời giải: [Sắp xếp mảng theo thứ tự giảm dần](#)

24. Nhập một mảng số nguyên $a_0, a_1, a_2, \dots, a_{n-1}$. Hãy sắp xếp mảng theo thứ tự tăng dần, sau đó chèn phần tử x vào mà vẫn đảm bảo mảng là tăng dần.
Lời giải: [Chèn phần tử vào mảng trong java](#)
25. Nhập một mảng số thực $a_0, a_1, a_2, \dots, a_{n-1}$. Không dùng thêm mảng số thực nào khác (có thể dùng thêm mảng số nguyên), hãy in ra màn hình mảng trên theo thứ tự tăng dần.
26. Nhập 2 mảng số thực $a_0, a_1, a_2, \dots, a_{m-1}$ và $b_0, b_1, b_2, \dots, b_{n-1}$. Giả sử 2 mảng này đã được sắp xếp tăng dần. Hãy tận dụng tính sắp xếp của 2 dãy và tạo dãy $c_0, c_1, c_2, \dots, c_{m+n-1}$ là hợp của 2 dãy trên sao cho c_i cũng có thứ tự tăng dần.
Lời giải: [Trộn 2 mảng trong java](#)
27. Viết chương trình nhập vào mảng A có n phần tử, các phần tử là số nguyên lớn hơn 0 và nhỏ hơn 100. Thực hiện các chức năng sau:
 - a) Tìm phần tử lớn thứ nhất và lớn thứ 2 trong mảng với các chỉ số của chúng (chỉ số đầu tiên tìm được).
 - b) Sắp xếp mảng theo thứ tự tăng dần.
 - c) Nhập số nguyên x và chèn x vào mảng A sao cho vẫn đảm bảo tính tăng dần cho mảng A.
28. Viết chương trình nhập vào ma trận A có n dòng, m cột, các phần tử là số nguyên lớn hơn 0 và nhỏ hơn 100. Thực hiện các chức năng sau:
 - a) Tìm phần tử lớn thứ nhất với chỉ số của nó (chỉ số đầu tiên tìm được).
 - b) Tìm và in ra các phần tử là số nguyên tố của ma trận (các phần tử không nguyên tố thì thay bằng số 0).
 - c) Sắp xếp tất cả các cột của ma trận theo thứ tự tăng dần và in kết quả ra màn hình.
 - d) Tìm cột trong ma trận có nhiều số nguyên tố nhất.

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

Bạn có thể xem các giải thuật sắp xếp trong phần cấu trúc dữ liệu và giải thuật: [Giải thuật sắp xếp](#)

Dưới đây là tổng hợp một số bài tập thuật toán sắp xếp trong Java:

- [Sắp xếp nổi bọt \(Bubble Sort\) trong Java](#)
- [Sắp xếp chon \(Selection Sort\) trong Java](#)
- [Sắp xếp chèn \(Insertion Sort\) trong Java](#)

- [Sắp xếp nhanh \(Quick Sort\) trong Java](#)
- [Sắp xếp trộn \(Merge Sort\) trong Java](#)
- [Sắp xếp Shell Sort trong Java](#)

5. Bài tập lập trình hướng đối tượng

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

- Lớp và đối tượng trong java.
- Access modifier trong java
- Các tính chất của lập trình hướng đối tượng (OOP).
- Các khái niệm Java OOPs.
- Collection trong java.
- Xử lý ngoại lệ trong java.

Bài tập quản lý sinh viên trong Java - console

Đề bài: Viết chương trình quản lý sinh viên. Mỗi đối tượng sinh viên có các thuộc tính sau: id, name, age, address và gpa (điểm trung bình). Yêu cầu: tạo ra một menu với các chức năng sau:

```

/*****/
1. Add student.
2. Edit student by id.
3. Delete student by id.
4. Sort student by gpa.
5. Sort student by name.
6. Show student.
0. Exit.
/*****/
    
```

Lời giải: [Bài tập quản lý sinh viên trong java - giao diện dòng lệnh](#)

Lời Giải Bài Tập Java Cơ Bản 1

Giải phương trình bậc 2 trong java

Đề bài

Viết chương trình **giải phương trình bậc 2** trong java. Phương trình bậc 2 có dạng:

$$ax^2 + bx + c = 0$$

Lời giải

Kiến thức sử dụng trong bài này, **java.util.Scanner** được sử dụng để đọc dữ liệu nhập vào từ bàn phím và [từ khóa static trong java](#). Bạn cũng nên tìm hiểu về [package trong java](#).

Bài này được viết trên eclipse, bạn có thể tham khảo bài [tạo chương trình java đầu tiên trên eclipse](#).

File: BaiTap1.java

```

1    package vn.viettuts.baitap;
2
3    import java.util.Scanner;
4
5    /**
6     * Giải phương trình bậc 2
7     *
8     * @author plpsoft.vn
9     */
10   public class BaiTap1 {
11       private static Scanner scanner = new Scanner(System.in);
12       /**
13        * main
14        *
15        * @param args
16        */
17       public static void main(String[] args) {
18           System.out.print("Nhập hệ số bậc 2, a = ");
19           float a = BaiTap1.scanner.nextFloat();
20           System.out.print("Nhập hệ số bậc 1, b = ");
21           float b = BaiTap1.scanner.nextFloat();
22           System.out.print("Nhập hằng số tự do, c = ");
23           float c = scanner.nextFloat();

```

```

24         BaiTap1.giaiPTBac2(a, b, c);
25     }
26
27     /**
28     * Giải phương trình bậc 2:  $ax^2 + bx + c = 0$ 
29     *
30     * @param a: hệ số bậc 2
31     * @param b: hệ số bậc 1
32     * @param c: số hạng tự do
33     */
34     public static void giaiPTBac2(float a, float b, float c) {
35         // kiểm tra các hệ số
36         if (a == 0) {
37             if (b == 0) {
38                 System.out.println("Phương trình vô nghiệm!");
39             } else {
40                 System.out.println("Phương trình có một nghiệm: "
41                     + "x = " + (-c / b));
42             }
43             return;
44         }
45         // tính delta
46         float delta = b*b - 4*a*c;
47         float x1;
48         float x2;
49         // tính nghiệm
50         if (delta > 0) {
51             x1 = (float) ((-b + Math.sqrt(delta)) / (2*a));
52             x2 = (float) ((-b - Math.sqrt(delta)) / (2*a));
53             System.out.println("Phương trình có 2 nghiệm là: "
54                 + "x1 = " + x1 + " và x2 = " + x2);
55         } else if (delta == 0) {
56             x1 = (-b / (2 * a));
57             System.out.println("Phương trình có nghiệm kép: "
58                 + "x1 = x2 = " + x1);
59         } else {
60             System.out.println("Phương trình vô nghiệm!");
61         }
62     }
63 }

```

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

Trong ví dụ trên, phương thức **Math.sqrt(double a)** được sử dụng để tính căn bậc 2 của a.

Tính giai thừa trong java

Đề bài

Viết chương trình tính giai thừa của một số nguyên.

Định nghĩa : giai thừa của 1 số là tích các số liên tiếp từ 1 đến số đó. Trường hợp đặc biệt, giai thừa của 0 và 1 là 1.

Ví dụ: giai thừa của 5 là $1*2*3*4*5 = 120$

Lời giải

Có 2 cách để viết chương trình tính giai thừa trong java:

- Tính giai thừa không sử dụng đệ quy
- Tính giai thừa có sử dụng đệ quy

Tính giai thừa không sử dụng đệ quy

Ví dụ chương trình tính giai thừa trong java không sử dụng phương pháp đệ quy:

```

1      public class GiaiThuaDemo1 {
2
3          /**
4           * main
5           *
6           * @author plpsoft.vn
7           * @param args
8           */
9      public static void main(String[] args) {
10         int a = 5;
11         int b = 0;
12         int c = 10;
13
14         System.out.println("Giai thừa của " + a + " là: "
15             + GiaiThuaDemo1.tinhGiaiThua(a));
16         System.out.println("Giai thừa của " + b + " là: "
17             + GiaiThuaDemo1.tinhGiaiThua(b));
18         System.out.println("Giai thừa của " + c + " là: "
19             + GiaiThuaDemo1.tinhGiaiThua(c));
20     }
21
22     /**
23      * tinh giai thua
24      *
25      * @author plpsoft.vn
    
```

```

26      * @param n: so nguyen duong
27      * @return giai thua cua so n
28      */
29      public static long tinhGiaithua(int n) {
30          long giai_thua = 1;
31          if (n == 0 || n == 1) {
32              return giai_thua;
33          } else {
34              for (int i = 2; i <= n; i++) {
35                  giai_thua *= i;
36              }
37              return giai_thua;
38          }
39      }
40  }

```

Kết quả:

Giai thừa của 5 là: 120

Giai thừa của 0 là: 1

Giai thừa của 10 là: 3628800

Chuyển đổi hệ cơ số trong java

Bài toán

1. Chuyển đổi số nguyên N từ hệ cơ số 10 sang hệ cơ số B bất kỳ.
2. Chuyển đổi hệ cơ số B sang hệ cơ số 10 bất kỳ.

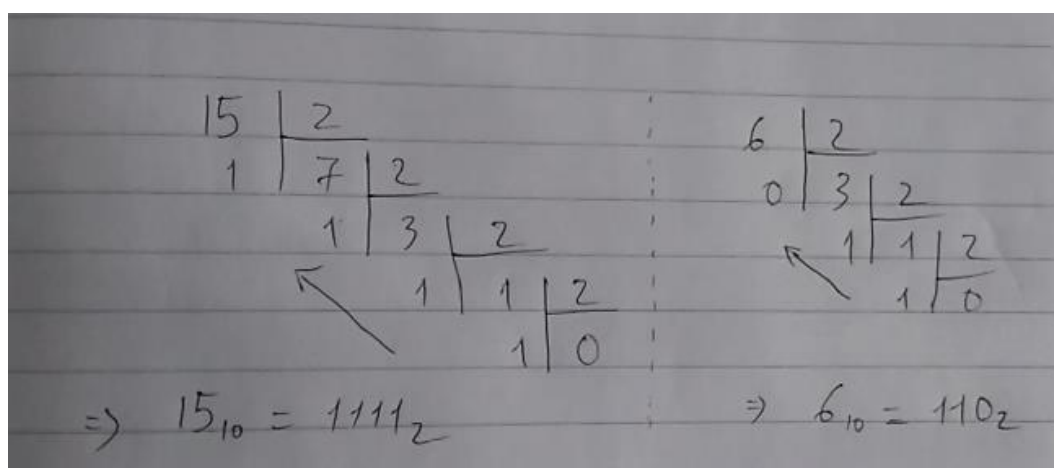
Cách chuyển đổi hệ cơ số

- Chuyển đổi hệ cơ số 10 sang hệ cơ số B
- Chuyển đổi hệ cơ số B sang hệ cơ số 10

Chuyển đổi hệ cơ số 10 sang hệ cơ số B

Cách chuyển đổi số nguyên N từ hệ cơ số 10 sang hệ cơ số 2, trong ảnh dưới đây là ví dụ chuyển số 15 và số 6 sang hệ cơ số 2:

1. Gán $m = 15$.
2. Put phần dư m chia cho 2 ($m \% 2$) vào stack.
3. Gán $m = m / 2$.
4. Nếu $m > 0$ quay lại bước 2.
5. Đảo ngược lại stack ta được số cần tính.



Các hệ số khác chuyển đổi tương tự.

Ví dụ chuyển đổi hệ cơ số 10 sang hệ cơ số B

Giả sử hệ cơ số cần chuyển là $2 \leq B \leq 16$. Số đại diện cho hệ cơ số $B > 10$ là A = 10, B = 11, C = 12, D = 13, E = 14, F = 15.

Chúng ta tạo ra một chương trình như sau để chuyển đổi hệ cơ số trong java:

```

1  public class ConvertNumber {
2      public static final char CHAR_55 = 55;
3
4      /**
5       * main
6       *
7       * @author plpsoft.vn
8       * @param args
9       */
10     public static void main(String[] args) {
11         int n = 15;
12         System.out.println("Số " + n + " trong hệ cơ số 2 = "
13             + ConvertNumber.convertNumber(n, 2));
14         System.out.println("Số " + n + " trong hệ cơ số 16 = "
15             + ConvertNumber.convertNumber(n, 16));
16     }
17
18     /**
19     * chuyển đổi số nguyên n sang hệ cơ số b
20     *
21     * @author plpsoft.vn
22     * @param n: số nguyên
23     * @param b: hệ cơ số
24     * @return hệ cơ số b
25     */
26     public static String convertNumber(int n, int b) {
27         if (n < 0 || b < 2 || b > 16) {
28             return "";
29         }
30
31         StringBuilder sb = new StringBuilder();
32         int m;
33         int remainder = n;
34
35         while (remainder > 0) {
36             if (b > 10) {
37                 m = remainder % b;
38                 if (m >= 10) {
39                     sb.append((char) (CHAR_55 + m));
40                 } else {
41                     sb.append(m);
42                 }
43             } else {
44                 sb.append(remainder % b);
45             }
46             remainder = remainder / b;
47         }
48     }
49 }

```

```

45         return sb.reverse().toString();
46     }
47 }
48

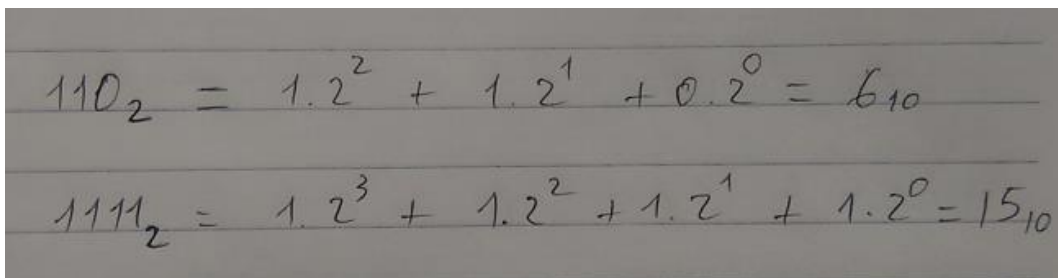
```

Kết quả:

So 15 trong he co so 2 = 1111
 So 15 trong he co so 16 = F

Chuyển đổi hệ cơ số B sang hệ cơ số 10

Trong ảnh dưới đây là ví dụ về chuyển đổi hệ cơ số 2 sang hệ cơ số 10.



$$110_2 = 1 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 = 6_{10}$$

$$1111_2 = 1 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 = 15_{10}$$

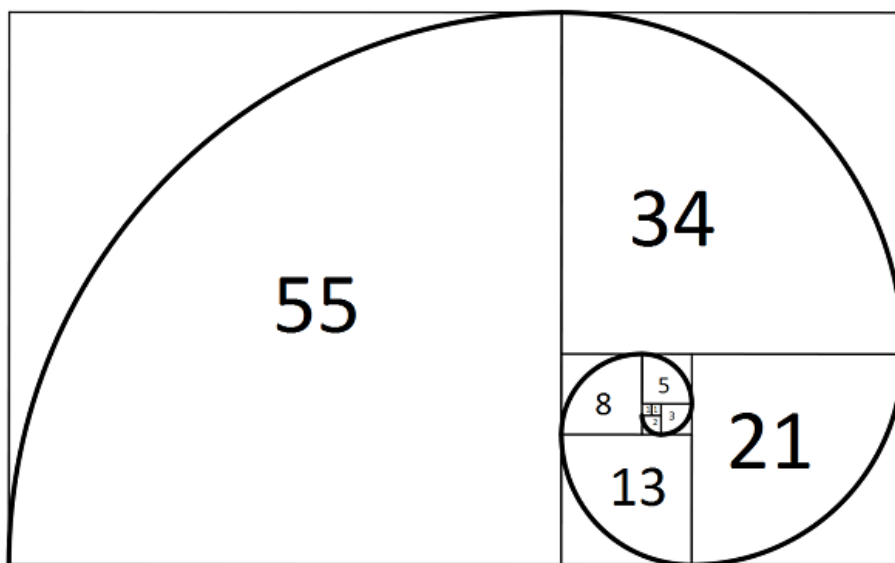
Các hệ cơ số khác tương tự.

Dãy số Fibonacci trong java

Đề bài

Viết chương trình tìm n số Fibonacci đầu tiên trong java. Số nguyên dương n được nhập từ bàn phím.

Quy luật của **dãy số Fibonacci**: số tiếp theo bằng tổng của 2 số trước, 2 số đầu tiên của dãy số là 0, 1. Ví dụ: 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, ...



Lời giải

Có 2 cách để viết chương trình dãy số Fibonacci trong java

- Tính dãy số Fibonacci trong java không dùng phương pháp đệ quy
- Tính dãy số Fibonacci trong java sử dụng phương pháp đệ quy

Tính dãy số Fibonacci không dùng phương pháp đệ quy

Ví dụ chương trình tính dãy số Fibonacci không sử dụng phương pháp đệ quy:

File: FibonacciExample1.java

```
1 package vn.viettuts.baitap;
```

```

2
3  /**
4   * Tính số fibonacci KHÔNG dùng phương pháp đệ quy
5   *
6   * @author plpsoft.vn
7   */
8  public class FibonacciExample1 {
9
10     /**
11     * main
12     *
13     * @param args
14     */
15     public static void main(String[] args) {
16         System.out.println("10 số đầu tiên của dãy số fibonacci: ");
17         for (int i = 0; i < 10; i++) {
18             System.out.print(fibonacci(i) + " ");
19         }
20     }
21
22     /**
23     * Tính số fibonacci thứ n
24     *
25     * @param n: chỉ số của số fibonacci tính từ 0
26     *           vd: F0 = 0, F1 = 1, F2 = 1, F3 = 2
27     * @return số fibonacci thứ n
28     */
29     public static int fibonacci(int n) {
30         int f0 = 0;
31         int f1 = 1;
32         int fn = 1;
33
34         if (n < 0) {
35             return -1;
36         } else if (n == 0 || n == 1) {
37             return n;
38         } else {
39             for (int i = 2; i < n; i++) {
40                 f0 = f1;
41                 f1 = fn;
42                 fn = f0 + f1;
43             }
44             return fn;
45         }
46     }
47 }

```

Kết quả:

```

10 số đầu tiên của dãy số fibonacci:
0 1 1 2 3 5 8 13 21 34

```

Tính dãy số Fibonacci sử dụng phương pháp đệ quy

Ví dụ chương trình tính dãy số Fibonacci sử dụng phương pháp đệ quy:

File: FibonacciExample2.java

```

1      package vn.vietttuts.baitap;
2
3      /**
4       * Tính dãy số Fibonacci bằng phương pháp đệ quy
5       *
6       * @author plpsoft.vn
7       */
8      public class FibonacciExample2 {
9
10         /**
11          * main
12          *
13          * @param args
14          */
15         public static void main(String[] args) {
16             System.out.println("10 số đầu tiên của dãy số fibonacci: ");
17             for (int i = 0; i < 10; i++) {
18                 System.out.print(fibonacci(i) + " ");
19             }
20         }
21
22         /**
23          * Tính số fibonacci thứ n
24          *
25          * @param n: chỉ số của số fibonacci tính từ 0
26          *          vd: F0 = 0, F1 = 1, F2 = 1, F3 = 2
27          * @return số fibonacci thứ n
28          */
29         public static int fibonacci(int n) {
30             if (n < 0) {
31                 return -1;
32             } else if (n == 0 || n == 1) {
33                 return n;
34             } else {
35                 return fibonacci(n - 1) + fibonacci(n - 2);
36             }
37         }
38     }

```

Kết quả:

```

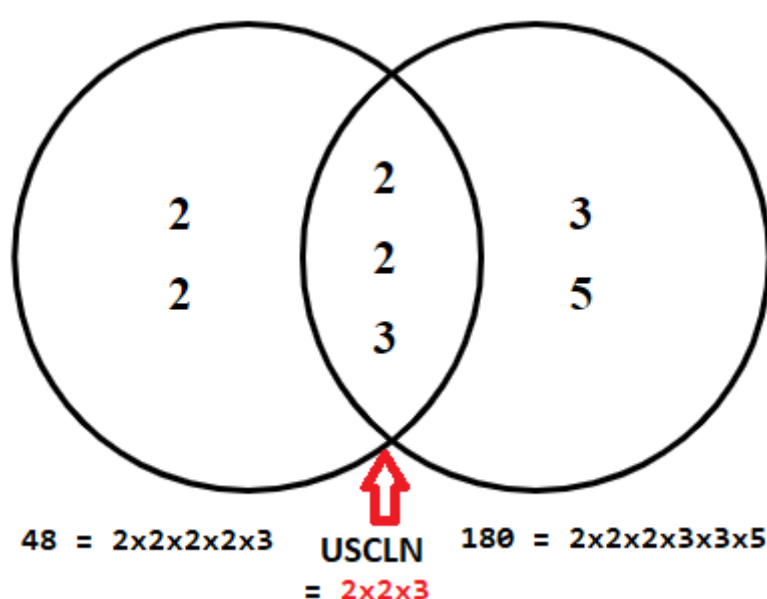
10 số đầu tiên của dãy số fibonacci:
0 1 1 2 3 5 8 13 21 34

```


Bài tập Java - Tìm ước số chung lớn nhất và bội số chung nhỏ nhất của 2 số

Bài tập Java - Tìm ước số chung lớn nhất và bội số chung nhỏ nhất của 2 số

Đề bài: Viết chương trình tìm ước số chung lớn nhất (**USCLN**) và bội số chung nhỏ nhất (**BSCNN**) của 2 số nguyên dương a và b nhập từ bàn phím.



Định nghĩa

USCLN của 2 số nguyên dương a và b là một số k lớn nhất, sao cho a và b đều chia hết cho k.

BSCNN của 2 số nguyên dương a và b là một số h nhỏ nhất, sao cho h chia hết cho cả a và b.

Lời giải

Một phương pháp đơn giản để tìm USCLN của a và b là duyệt từ số nhỏ hơn trong 2 số a và b cho đến 1, khi gặp số nào đó mà cả a và b đều chia hết cho nó thì đó chính là USCLN của a và b. Tuy vậy phương pháp này chưa phải là hiệu quả nhất.

Vào thế kỷ 3 TCN, nhà toán học Euclid (phiên âm tiếng Việt là **O'-clit**) đã phát minh ra một giải thuật tìm USCLN của hai số nguyên dương rất hiệu quả được gọi là giải thuật Euclid. Cụ thể về ý tưởng của bài toán, giả sử a lớn hơn b , khi đó việc tính USCLN của a và b sẽ được đưa về bài toán tính USCLN của $a \bmod b$ và b vì $\text{USCLN}(a, b) = \text{USCLN}(a \bmod b, b)$.

Khi đã tìm được USCLN thì việc tìm BSCNN của hai số nguyên dương a và b khá đơn giản. Khi đó $\text{BSCNN}(a, b) = (a * b) / \text{USCLN}(a, b)$.

1. Tìm USCLN và BSCNN của 2 số nguyên dương trong Java bằng giải thuật Euclid

Ví dụ dưới đây sử dụng **giải thuật Euclid** để giải quyết bài toán tìm ước số chung lớn nhất (**USCLN**) và bội số chung nhỏ nhất (**BSCNN**) của hai số nguyên dương a và b .

File: BaiTap5.java

```

1  package vn.viettuts.baitap;
2
3  import java.util.Scanner;
4
5  /**
6   * Chương trình tìm ước số chung lớn nhất (USCLN)
7   * và bội số chung nhỏ nhất (BSCNN) của 2 số a và b
8   *
9   * @author plpsoft.vn
10  */
11  public class BaiTap5 {
12      private static Scanner scanner = new Scanner(System.in);
13      /**
14       * main
15       *
16       * @param args
17       */
18      public static void main(String[] args) {
19          System.out.print("Nhập số nguyên dương a = ");
20          int a = scanner.nextInt();
21          System.out.print("Nhập số nguyên dương b = ");
22          int b = scanner.nextInt();
23          // tính USCLN của a và b
24          System.out.println("USCLN của " + a + " và " + b
25              + " là: " + USCLN(a, b));
26          // tính BSCNN của a và b
27          System.out.println("BSCNN của " + a + " và " + b
28              + " là: " + BSCNN(a, b));
29      }
30
31      /**
32       * Tìm ước số chung lớn nhất (USCLN)
33       *
34       * @param a: số nguyên dương
35       * @param b: số nguyên dương
36       * @return USCLN của a và b
37       */

```

```

38     public static int USCLN(int a, int b) {
39         if (b == 0) return a;
40         return USCLN(b, a % b);
41     }
42
43     /**
44      * Tìm bội số chung nhỏ nhất (BSCNN)
45      *
46      * @param a: số nguyên dương
47      * @param b: số nguyên dương
48      * @return BSCNN của a và b
49      */
50     public static int BSCNN(int a, int b) {
51         return (a * b) / USCLN(a, b);
52     }
53 }

```

Kết quả:

```

Nhập số nguyên dương a = 15
Nhập số nguyên dương b = 40
USCLN của 15 và 40 là: 5
BSCNN của 15 và 40 là: 120

```

2. Tìm USCLN và BSCNN của 2 số nguyên dương trong Java bằng cách khác

```

1     package vn.viettuts.baitap;
2
3     import java.util.Scanner;
4
5     /**
6      * Chương trình tìm ước số chung lớn nhất (USCLN)
7      * và bội số chung nhỏ nhất (BSCNN) của 2 số a và b
8      *
9      * @author plpsoft.vn
10    */
11    public class BaiTap5_2 {
12        private static Scanner scanner = new Scanner(System.in);
13        /**
14         * main
15         *
16         * @param args
17         */
18        public static void main(String[] args) {
19            System.out.print("Nhập số nguyên dương a = ");
20            int a = scanner.nextInt();
21            System.out.print("Nhập số nguyên dương b = ");
22            int b = scanner.nextInt();
23            // tính USCLN của a và b
24            System.out.println("USCLN của " + a + " và " + b
25                               + " là: " + USCLN(a, b));
26            // tính BSCNN của a và b
27            System.out.println("BSCNN của " + a + " và " + b
28                               + " là: " + BSCNN(a, b));

```

```

29     }
30
31     /**
32     * Tìm ước số chung lớn nhất (USCLN)
33     *
34     * @param a: số nguyên dương
35     * @param b: số nguyên dương
36     * @return USCLN của a và b
37     */
38     public static int USCLN(int a, int b) {
39         int temp1 = a;
40         int temp2 = b;
41         while (temp1 != temp2) {
42             if (temp1 > temp2) {
43                 temp1 -= temp2;
44             } else {
45                 temp2 -= temp1;
46             }
47         }
48         int uscln = temp1;
49         return uscln;
50     }
51
52     /**
53     * Tìm bội số chung nhỏ nhất (BSCNN)
54     *
55     * @param a: số nguyên dương
56     * @param b: số nguyên dương
57     * @return BSCNN của a và b
58     */
59     public static int BSCNN(int a, int b) {
60         return (a * b) / USCLN(a, b);
61     }
62 }

```

Kết quả:

```

Nhập số nguyên dương a = 15
Nhập số nguyên dương b = 20
Ước số chung lớn nhất của 15 và 20 là: 5
Bội số chung nhỏ nhất của 15 và 20 là: 60

```

Giải thích hoạt động của chương trình trên: Trong chương trình này, tôi nhập vào hai số 15 và 20 thì trình biên dịch sẽ thực thi hàm uscln() với các bước như sau:

1. Gán giá trị biến temp1 = 15 và temp2 = 20.
2. Kiểm tra điều kiện bên trong while: Vì 15 != 20 nên sẽ thực thi lệnh bên trong while. Lúc này 15 < 20 nên lệnh bên trong else sẽ được thực thi và lúc này biến temp2 = 5.
3. Quay lại bước 3, kiểm tra điều kiện bên trong while: Vì 15 != 5 nên sẽ thực thi lệnh bên trong while. Lúc này 15 > 5 nên lệnh bên trong if sẽ được thực thi và lúc này biến temp1 = 10.

4. Quay lại bước 3, kiểm tra điều kiện bên trong while: Vì $10 \neq 5$ nên sẽ thực thi lệnh bên trong while. Lúc này $10 > 5$ nên lệnh bên trong if sẽ được thực thi và lúc này biến `temp1 = 5`.
5. Quay lại bước 3, kiểm tra điều kiện bên trong while: Vì $5 == 5$ nên sẽ không thực thi lệnh bên trong while. Vòng lặp kết thúc, trả về giá trị `uscln = 5`.

Bài tập Java - Liệt kê tất cả các số nguyên tố nhỏ hơn n

Bài tập Java - Liệt kê tất cả các số nguyên tố nhỏ hơn n

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

Định nghĩa

Số nguyên tố là số lớn hơn 1 và chỉ chia hết cho 1 và chính nó. Ví dụ: 2, 3, 5, 7, 11, 13, 17, ... là những số nguyên tố. **Chú ý:** Số 0 và 1 không phải là số nguyên tố. Chỉ có số 2 là số nguyên tố chẵn, tất cả các số chẵn khác không phải là số nguyên tố vì chúng chia hết cho 2.

Ví dụ số nguyên tố

Danh sách số nguyên tố nhỏ hơn 100:

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

Lời giải

File: BaiTap6.java

```

1  package vn.vietttuts.baitap;
2
3  import java.util.Scanner;
4
5  /**
6   * Chương trình liệt kê tất cả các số nguyên tố nhỏ hơn n.
7   *
8   * @author plpsoft.vn
9   */
10 public class BaiTap6 {
11     private static Scanner scanner = new Scanner(System.in);
12
13     /**
14      * main
15      *
16      * @param args
17      */
18     public static void main(String[] args) {
19         System.out.print("Nhập n = ");
20         int n = scanner.nextInt();
21         System.out.printf("Tất cả các số nguyên tố nhỏ hơn %d là: \n", n);
22         if (n >= 2) {
23             System.out.print(2);
24         }
25         for (int i = 3; i < n; i+=2) {
26             if (isPrimeNumber(i)) {
27                 System.out.print(" " + i);
28             }
29         }
30     }
31
32     /**
33      * check so nguyen to
34      *
35      * @author plpsoft.vn
36      * @param n: so nguyen duong
37      * @return true la so nguyen so,
38      *         false khong la so nguyen to
39      */
40     public static boolean isPrimeNumber(int n) {
41         // so nguyen n < 2 khong phai la so nguyen to
42         if (n < 2) {
43             return false;
44         }
45         // check so nguyen to khi n >= 2
46         int squareRoot = (int) Math.sqrt(n);
47         for (int i = 2; i <= squareRoot; i++) {
48             if (n % i == 0) {
49                 return false;
50             }
51         }
52         return true;
53     }
54 }

```

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 tập Java - Liệt kê n số nguyên tố đầu tiên trong Java

Bài tập Java - Liệt kê n số nguyên tố đầu tiên trong Java

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

Định nghĩa

Số nguyên tố là số lớn hơn 1 và chỉ chia hết cho 1 và chính nó. Ví dụ: 2, 3, 5, 7, 11, 13, 17, ... là những số nguyên tố. **Chú ý:** Số 0 và 1 không phải là số nguyên tố. Chỉ có số 2 là số nguyên tố chẵn, tất cả các số chẵn khác không phải là số nguyên tố vì chúng chia hết cho 2.

Ví dụ số nguyên tố

Danh sách số nguyên tố nhỏ hơn 100:

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

Lời giải

File: BaiTap7.java

```
1 package vn.viettuts.baitap;
2
```

```

3     import java.util.Scanner;
4
5     /**
6      * Chương trình liệt kê n số nguyên tố đầu tiên.
7      *
8      * @author plpsoft.vn
9      */
10    public class BaiTap7 {
11        private static Scanner scanner = new Scanner(System.in);
12
13        /**
14         * main
15         *
16         * @param args
17         */
18        public static void main(String[] args) {
19            System.out.print("Nhập n = ");
20            int n = scanner.nextInt();
21            System.out.printf("%d số nguyên tố đầu tiên là: \n", n);
22            int dem = 0; // đếm số số nguyên tố
23            int i = 2;   // tìm số nguyên tố bắt đầu từ số 2
24            while (dem < n) {
25                if (isPrimeNumber(i)) {
26                    System.out.print(i + " ");
27                    dem++;
28                }
29                i++;
30            }
31        }
32
33        /**
34         * check so nguyen to
35         *
36         * @author plpsoft.vn
37         * @param n: so nguyen duong
38         * @return true la so nguyen so,
39         *         false khong la so nguyen to
40         */
41        public static boolean isPrimeNumber(int n) {
42            // so nguyen n < 2 khong phai la so nguyen to
43            if (n < 2) {
44                return false;
45            }
46            // check so nguyen to khi n >= 2
47            int squareRoot = (int) Math.sqrt(n);
48            for (int i = 2; i <= squareRoot; i++) {
49                if (n % i == 0) {
50                    return false;
51                }
52            }
53            return true;
54        }
55    }

```

Kết quả:

```

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

```

Bài tập Java - Liệt kê tất cả số nguyên tố có 5 chữ số trong Java

Bài tập Java - Liệt kê tất cả số nguyên tố có 5 chữ số trong Java

Đề bài: Viết chương trình liệt kê tất cả số nguyên tố có 5 chữ số trong Python.

Định nghĩa

Số nguyên tố là số lớn hơn 1 và chỉ chia hết cho 1 và chính nó. Ví dụ: 2, 3, 5, 7, 11, 13, 17, ... là những số nguyên tố. **Chú ý:** Số 0 và 1 không phải là số nguyên tố. Chỉ có số 2 là số nguyên tố chẵn, tất cả các số chẵn khác không phải là số nguyên tố vì chúng chia hết cho 2.

Ví dụ số nguyên tố

Danh sách số nguyên tố nhỏ hơn 100:

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

Lời giải

```
1 package vn.viettuts.baitap;
```

```

2
3  /**
4   * Chương trình liệt kê tất cả số nguyên tố có 5 chữ số.
5   *
6   * @author plpsoft.vn
7   */
8  public class BaiTap8 {
9
10     /**
11     * main
12     *
13     * @param args
14     */
15     public static void main(String[] args) {
16         int count = 0;
17         System.out.println("Liệt kê tất cả số nguyên tố có 5 chữ số:");
18         for (int i = 10001; i < 99999; i+=2) {
19             if (isPrimeNumber(i)) {
20                 System.out.println(i);
21                 count++;
22             }
23         }
24         System.out.println("Tổng các số nguyên tố có 5 chữ số là: " + count);
25     }
26
27     /**
28     * check so nguyen to
29     *
30     * @author plpsoft.vn
31     * @param n: so nguyen duong
32     * @return true la so nguyen so,
33     *         false khong la so nguyen to
34     */
35     public static boolean isPrimeNumber(int n) {
36         // so nguyen n < 2 khong phai la so nguyen to
37         if (n < 2) {
38             return false;
39         }
40         // check so nguyen to khi n >= 2
41         int squareRoot = (int) Math.sqrt(n);
42         for (int i = 2; i <= squareRoot; i++) {
43             if (n % i == 0) {
44                 return false;
45             }
46         }
47         return true;
48     }
49 }

```

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 tập Java - Phân tích số nguyên n thành tích các số nguyên tố

Bài tập Java - Phân tích số nguyên n thành tích các số nguyên tố

Đề bài: Viết chương trình phân tích số nguyên n thành các thừa số nguyên tố trong Java. Ví dụ: $12 = 2 \times 2 \times 3$. Số nguyên dương n được nhập từ bàn phím.

$$100 = 2 \times 2 \times 5 \times 5$$

Lời giải

Sau đây là chương trình java phân tích số nguyên n thành tích các số nguyên tố:

File: BaiTap9.java

```

1    package vn.viettuts.baitap;
2
3    import java.util.ArrayList;
4    import java.util.List;
5    import java.util.Scanner;
6
7    /**
8     * Chương trình phân tích số nguyên n thành các thừa số nguyên tố.
9     * Ví dụ:  $12 = 2 \times 2 \times 3$ .
10    *
11    * @author plpsoft.vn
12    */
13    public class BaiTap9 {
14        private static Scanner scanner = new Scanner(System.in);
15
16        /**
17         * main
18         *
19         * @param args
20         */
21        public static void main(String[] args) {
22            System.out.print("Nhập số nguyên dương n = ");

```

```

23         int n = scanner.nextInt();
24         // phân tích số nguyên dương n
25         List<Integer> listNumbers = phanTichSoNguyen(n);
26         // in kết quả ra màn hình
27         System.out.printf("Kết quả: %d = ", n);
28         int size = listNumbers.size();
29         for (int i = 0; i < size - 1; i++) {
30             System.out.print(listNumbers.get(i) + " x ");
31         }
32         System.out.print(listNumbers.get(size - 1));
33     }
34
35     /**
36      * Phân tích số nguyên thành tích các thừa số nguyên tố
37      *
38      * @param positiveInt
39      * @return
40      */
41     public static List<Integer> phanTichSoNguyen(int n) {
42         int i = 2;
43         List<Integer> listNumbers = new ArrayList<Integer>();
44         // phân tích
45         while (n > 1) {
46             if (n % i == 0) {
47                 n = n / i;
48                 listNumbers.add(i);
49             } else {
50                 i++;
51             }
52         }
53         // nếu listNumbers trống thì add n vào listNumbers
54         if (listNumbers.isEmpty()) {
55             listNumbers.add(n);
56         }
57         return listNumbers;
58     }
59 }

```

Kết quả:

Nhập số nguyên dương n = 100

Kết quả: 100 = 2 x 2 x 5 x 5

Trong ví dụ trên, chúng tôi sử dụng **List interface** để lưu trữ các phần tử là các thừa số nguyên tố.

Bài tập Java - Tính tổng của các chữ số của một số nguyên n trong Java

Bài tập Java - Tính tổng của các chữ số của một số nguyên n trong Java

Đề bài: viết chương trình tính tổng các chữ số của một số nguyên n trong Java. Số nguyên dương n được nhập từ bàn phím. Ví dụ: với 1234, tổng các chữ số là $1 + 2 + 3 + 4 = 10$.

1234, 1+2+3+4=10

Lời giải

Dưới đây là chương trình tính tổng các chữ số của một số nguyên n trong Java.

File: BaiTap10.java

```

1    package vn.viettuts.baitap;
2
3    import java.util.Scanner;
4
5    /**
6     * Chương trình tính tổng của các chữ số của một số nguyên dương n.
7     * Tổng của các chữ số của 6677 là 6 + 6 + 7 + 7 = 26.
8     *
9     * @author plpsoft.vn
10    */
11    public class BaiTap10 {
12        private static Scanner scanner = new Scanner(System.in);
13        public static int DEC_10 = 10;
14
15        /**
16         * main
17         *
18         * @param args
19         */
20        public static void main(String[] args) {
21            System.out.print("Nhập số nguyên dương n = ");
22            int n = scanner.nextInt();
23            System.out.printf("Tổng của các chữ số "
24                            + "của %d là: %d", n, totalDigitsOfNumber(n));
25        }
26
27        /**
28         * Tính tổng của các chữ số của một số nguyên dương

```

```

29      *
30      * @param n: số nguyên dương
31      * @return
32      */
33      public static int totalDigitsOfNumber(int n) {
34          int total = 0;
35          do {
36              total = total + n % DEC_10;
37              n = n / DEC_10;
38          } while (n > 0);
39          return total;
40      }
41  }

```

Kết quả:

Nhập số nguyên dương n = 6677

Tổng của các chữ số của 6677 là: 26

Bài tập Java - Tìm số thuận nghịch trong java

Bài tập Java - Tìm số thuận nghịch trong java

Đề bài: Viết chương trình tìm các số thuận nghịch có sáu chữ số từ 100000 đến 999999. Một số được gọi là **số thuận nghịch** nếu nó bằng số đảo ngược, tức là ta đọc từ trái sang phải hay từ phải sang trái số đó ta vẫn nhận được một số giống nhau. Ví dụ 123321 là một số thuận nghịch.

123321 -> True
123451 -> False

Lời giải

Cách 1: Duyệt các số từ 100000 đến 999999, rồi phân tách số hiện tại thành các chữ số đơn lẻ để kiểm tra tính thuận nghịch.

File: BaiTap11.java

```

1    package vn.vietttuts.baitap;
2
3    import java.util.ArrayList;
4    import java.util.List;
5
6    /**
7     * Chương trình liệt kê tất cả các số thuận nghịch có 6 chữ số.
8     *
9     * @author plpsoft.vn
10    */
11    public class BaiTap11 {
12        public static int DEC_10 = 10;
13
14        /**
15         * main
16         *
17         * @param args
18         */
19        public static void main(String[] args) {
20            int count = 0;
21            // in ra màn hình các số thuận nghịch có 6 chữ số
22            for (int i = 100000; i < 1000000; i++) {
23                if (isThuanNghich(i)) {
24                    System.out.println(i);
25                    count++;
26                }
27            }
28        }
29    }

```

```

26         }
27         System.out.println("Tổng các số thuận nghịch có 6 chữ số: "
28             + count);
29     }
30
31
32     /**
33      * Kiểm tra số thuận nghịch
34      *
35      * @param n: số nguyên dương
36      * @return true là số thuận nghịch
37      *         false không là số thuận nghịch
38      */
39     public static boolean isThuanNghich(int n) {
40         List<Integer> listNumbers = new ArrayList<>();
41         // phân tích số n thành các chữ số và lưu vào listNumbers
42         do {
43             listNumbers.add(n % DEC_10);
44             n = n / DEC_10;
45         } while (n > 0);
46         // kiểm tra tính thuận nghịch
47         int size = listNumbers.size();
48         for (int i = 0; i < (size/2); i++) {
49             if (listNumbers.get(i) != listNumbers.get(size - i - 1)) {
50                 return false;
51             }
52         }
53         return true;
54     }
55 }
56

```

Kết quả:

```

100001
101101
102201
...
997799
998899
999999
Tổng các số thuận nghịch có 6 chữ số: 900

```

Cách 2: sử dụng thuật toán + [StringBuilder trong java](#)

```

1     /**
2      * Chương trình liệt kê tất cả các số thuận nghịch có 6 chữ số.
3      *
4      * @author plpsoft.vn
5      */
6     package vn.viettuts.baitap;
7
8     public class BaiTap11_2 {
9         public static final int[] DEC_ARR_1_9 = {1, 2, 3, 4, 5, 6, 7, 8, 9};

```

```

10     public static final int[] DEC_ARR_10 = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9};
11
12     /**
13      * main
14      *
15      * @param args
16      */
17     public static void main(String[] args) {
18         int count = 0;
19         StringBuilder sb;
20         // in ra màn hình các số thuận nghịch có 6 chữ số
21         for (int a : DEC_ARR_10) {
22             for (int b : DEC_ARR_10) {
23                 for (int c : DEC_ARR_10) {
24                     sb = new StringBuilder();
25                     sb.append(a).append(b).append(c).append(c).append(b).append(a);
26                     System.out.println(sb.toString());
27                     count++;
28                 }
29             }
30         }
31         System.out.println("Tổng các số thuận nghịch có 6 chữ số: "
32                             + count);
33     }
34 }

```

Kết quả:

```

100001
101101
102201
...
997799
998899
999999
Tổng các số thuận nghịch có 6 chữ số: 900

```

Cách 3: Chuyển đổi n thành chuỗi String rồi kiểm tra tính thuận nghịch.

```

1     package vn.viettuts.baitap;
2
3     /**
4      * Chương trình liệt kê tất cả các số thuận nghịch có 6 chữ số.
5      *
6      * @author plpsoft.vn
7      */
8     public class BaiTap11_3 {
9
10        /**
11         * main
12         *
13         * @param args
14         */
15        public static void main(String[] args) {

```

```

16         int count = 0;
17         // in ra màn hình các số thuận nghịch có 6 chữ số
18         for (int i = 100000; i < 1000000; i++) {
19             if (isThuanNghich(i)) {
20                 System.out.println(i);
21                 count++;
22             }
23         }
24         System.out.println("Tổng các số thuận nghịch có 6 chữ số: "
25             + count);
26     }
27
28
29     /**
30     * Kiểm tra số thuận nghịch
31     *
32     * @param n: số nguyên dương
33     * @return true là số thuận nghịch
34     *         false không là số thuận nghịch
35     */
36     public static boolean isThuanNghich(int n) {
37         // chuyển đổi số n thành một chuỗi String
38         String numberStr = String.valueOf(n);
39         // kiểm tra tính thuận nghịch
40         int size = numberStr.length();
41         for (int i = 0; i < (size/2); i++) {
42             if (numberStr.charAt(i) != numberStr.charAt(size - i - 1)) {
43                 return false;
44             }
45         }
46         return true;
47     }
48 }

```

Kết quả:

```

100001
101101
102201
...
997799
998899
999999
Tổng các số thuận nghịch có 6 chữ số: 900

```

Bài tập Java - Liệt kê số Fibonacci nhỏ hơn n và là số nguyên tố

Đề bài

Đề bài: 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.

Lời giải

Sau đây là chương trình java liệt kê các số Fibonacci nhỏ hơn n là số nguyên tố trong Java.

File: BaiTap12.java

```

1  package vn.viettuts.baitap;
2
3  import java.util.Scanner;
4
5  /**
6   * Chương trình liệt kê các số Fibonacci nhỏ hơn n là số nguyên tố.
7   * Với n được nhập từ bàn phím.
8   *
9   * @author plpsoft.vn
10  */
11  public class BaiTap12 {
12      private static Scanner scanner = new Scanner(System.in);
13      /**
14       * main
15       *
16       * @param args
17       */
18      public static void main(String[] args) {
19          System.out.print("Nhập số tự nhiên n = ");
20          int n = scanner.nextInt();
21          System.out.printf("Các số fibonacci nhỏ hơn %d và "
22              + "là số nguyên tố: ", n);
23          int i = 0;
24          while (fibonacci(i) < 100) {
25              int fi = fibonacci(i);
26              if (isPrimeNumber(fi)) {
27                  System.out.print(fi + " ");
28              }
29              i++;
30          }
31      }
32      /**

```

```

33      * Tính số fibonacci thứ n
34      *
35      * @param n: chỉ số của số fibonacci tính từ 0
36      *          vd: F0 = 0, F1 = 1, F2 = 1, F3 = 2
37      * @return số fibonacci thứ n
38      */
39      public static int fibonacci(int n) {
40          if (n < 0) {
41              return -1;
42          } else if (n == 0 || n == 1) {
43              return n;
44          } else {
45              return fibonacci(n - 1) + fibonacci(n - 2);
46          }
47      }
48
49      /**
50       * check số nguyên tố
51       *
52       * @author plpsoft.vn
53       * @param n: số nguyên dương
54       * @return true là số nguyên tố,
55       *         false không là số nguyên tố
56       */
57      public static boolean isPrimeNumber(int n) {
58          // số nguyên n < 2 không phải là số nguyên tố
59          if (n < 2) {
60              return false;
61          }
62          // check số nguyên tố khi n >= 2
63          int squareRoot = (int) Math.sqrt(n);
64          for (int i = 2; i <= squareRoot; i++) {
65              if (n % i == 0) {
66                  return false;
67              }
68          }
69          return true;
70      }
71  }
72

```

Kết quả:

Nhập số tự nhiên n = 100

Các số fibonacci nhỏ hơn 100 và là số nguyên tố: 2 3 5 13 89

Bài tập Java - Viết chương trình java tính hàm SLOPE trong excel

1. Mô tả và công thức tính hàm SLOPE trong Microsoft Excel

Mô tả

Hàm SLOPE trả về độ dốc của đường hồi quy tuyến tính thông qua **các điểm dữ liệu (x, y)** trong known_y's và known_x's. Độ dốc là khoảng cách dọc chia cho khoảng cách ngang giữa bất kỳ hai điểm nào trên đường đó, là tỉ lệ thay đổi dọc theo đường hồi quy.

Cú pháp

1 SLOPE (known_y's, known_x's)

Trong đó:

- **known_y's**: Bắt buộc. Mảng hoặc Range có kiểu dữ liệu dạng **số** hoặc **date**.
- **known_x's**: Bắt buộc. Mảng hoặc Range có kiểu dữ liệu dạng **số** hoặc **date**.

Công thức tính

$$b = \frac{\sum (x - \text{averageX})(y - \text{averageY})}{\sum (x - \text{averageX})^2}$$

Trong đó:

- **x**: là các phần tử của known_x's.
- **averageX**: là giá trị trung bình của known_x's.
- **y**: là các phần tử của known_y's.
- **averageY**: là giá trị trung bình của known_y's.

2. Cách dùng hàm SLOPE trong Microsoft Excel

Ví dụ:

	A	B	C	D	E	F
1						
2	12/19/2017	4		2	4	
3	12/18/2017	1		1	1	
4	1/2/2018	2		16	2	
5						
6		-0.714285714			-0.714285714	
7						
8						

Trong ví dụ trên:

Công thức	Mô tả	Kết quả
B6 = SLOPE(A2:A4, B2:B4)	Độ dốc của đường hồi quy tuyến tính qua các điểm dữ liệu tại A2:A4 và B2:B4.	-0.714285714
E6 = SLOPE(D2:D4, E2:E4)	Độ dốc của đường hồi quy tuyến tính qua các điểm dữ liệu tại D2:D4 và E2:E4.	-0.714285714

Bạn để ý thấy rằng **Range(A2:A4)** và **Range(D2:D4)** tương đương với nhau. Vì B6 và E6 trả về giá trị giống nhau.

Khi giá trị của known_x's hoặc known_y's là **kiểu date** thì nó sẽ được quy đổi sang kiểu số trước khi tính slope. **Với quy ước giá trị date nhỏ nhất bằng 1.0 tính theo đơn vị ngày.**

3. Viết chương trình java tính hàm SLOPE trong excel

Đề bài: Viết chương trình java tính hàm SLOPE trong excel. Với known_x's là một mảng các số và known_y's là một mảng các giá trị có kiểu date.

Tạo hàm lớp common chứa các phương thức tính slope và phương thức chuyển đổi dạng list date thành dạng list các số:

```

1 package vn.viettuts;
2
3 import java.text.ParseException;
4 import java.text.SimpleDateFormat;
5 import java.util.ArrayList;
6 import java.util.Date;
7 import java.util.List;
8
9 public class Common {
10     public static final long MILLI_SECOND_PER_DAY = 60 * 60 * 24 * 1000;
11     public static final String DATE_FOMART = "MM/dd/yyyy";

```



```

11
12     /**
13      * tính slope theo công thức  $slope = \frac{\sum (x - averageX)(y - averageY)}{\sum (x -$ 
14      *  $averageX)^2}$ 
15      * trong đó: averageX là giá trị trung bình của listX
16      *           averageY là giá trị trung bình của listY
17      *
18      * @param listY: danh sách số thực
19      * @param listX: danh sách số thực
20      * @return slope của listX và listY
21      * @throws ParseException
22      */
23     public static float slope(List<Float> listY, List<Float> listX)
24         throws ParseException {
25         float averageX;
26         float averageY;
27         float totalX = 0;
28         float totalY = 0;
29         float totalAX_AY = 0;
30         float totalAX_AX = 0;
31         int i;
32
33         // tính tổng cột x và y
34         for (i = 0; i < listX.size(); i++) {
35             totalX = totalX + listX.get(i);
36             totalY = totalY + listY.get(i);
37         }
38         // tính giá trị trung bình của listX và listY
39         averageX = totalX / listX.size();
40         averageY = totalY / listX.size();
41
42         for (i = 0; i < listX.size(); i++) {
43             // tính tổng  $(x - averageX)(y - averageY)$ 
44             totalAX_AY = totalAX_AY
45                 + ((listX.get(i) - averageX) * (listY.get(i) -
46                 averageY));
47             // tính tổng  $(x - averageX)^2$ 
48             totalAX_AX = totalAX_AX
49                 + ((listX.get(i) - averageX) * (listX.get(i) -
50                 averageX));
51         }
52         // return slope của dãy số x và dãy số y
53         return totalAX_AY / totalAX_AX;
54     }
55
56     /**
57      * chuyển đổi danh sách date dạng chuỗi thành danh sách số thực
58      * với quy ước min date (day) = 1.0
59      *
60      * @param listStrDates: danh sách date dạng chuỗi
61      * @return List<Float>: danh sách số thực
62      * @throws ParseException
63      */
64     public static List<Float> parseDateToFloats(List<String>
65     listStrDates)
66         throws ParseException {
67         List<Date> listDates = new ArrayList<>();
68         List<Float> listY = new ArrayList<>();
69
70         // phân tích string thành date
71         listDates = parseStringToDates(listStrDates);
72     }

```

```

68         Date minDate = listDates.get(0);
69
70         // tìm min date
71         for (Date date : listDates) {
72             if (minDate.getTime() > date.getTime()) {
73                 minDate = date;
74             }
75         }
76
77         // chuyển đổi date thành số thực với quy ước min date (day) =
1.0
78         for (Date date : listDates) {
79             listY.add((float) (
80                 (date.getTime() - minDate.getTime()) /
81                 MILLI_SECOND_PER_DAY)
82                 + 1.0F);
83         }
84
85         return listY;
86     }
87
88     /**
89     * phân tích list string thành list date
90     *
91     * @param listStrDates: list date dạng string
92     * @return List<Date>
93     * @throws ParseException
94     */
95     public static List<Date> parseStringToDates(List<String>
listStrDates)
96     throws ParseException {
97         List<Date> listDates = new ArrayList<>();
98         SimpleDateFormat dateFormat = new SimpleDateFormat(DATE_FOMART);
99
100        for (String strDate : listStrDates) {
101            listDates.add(dateFormat.parse(strDate.trim()));
102        }
103
104        return listDates;
105    }
106 }

```

Tạo lớp Test.java

```

1    package vn.viettuts;
2
3    import java.text.ParseException;
4    import java.util.ArrayList;
5    import java.util.List;
6
7    public class Test {
8
9        public static void main(String[] args) {

```

```

9         float slope;
10        List<String> listStrDates = new ArrayList<>();
11        List<Float> listY = new ArrayList<>();
12        List<Float> listX = new ArrayList<>();
13
14        try {
15            // add gia tri cho listStrDates
16            listStrDates.add("12/19/2017");
17            listStrDates.add("12/18/2017");
18            listStrDates.add("1/2/2018");
19            // chuyen doi listStrDates sang dang so listY
20            listY = Common.parseDateToFloats(listStrDates);
21            // define listX
22            listX.add(4F);
23            listX.add(1F);
24            listX.add(2F);
25            // tinh slope cua listY va listX
26            slope = Common.slope(listY, listX);
27            System.out.println(slope);
28        } catch (ParseException e) {
29            e.printStackTrace();
30        }
31    }

```

Kết quả:

```
-0.7142857
```

Bài tập Java - Đếm số từ trong một chuỗi

Bài tập Java - Đếm số từ trong một chuỗi

Đề bài: Viết chương trình đếm số từ trong một chuỗi trong Java. Mỗi từ cách nhau bởi một khoảng trắng (tab, space, ...) 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

```

1    package vn.viettuts.baitap;
2
3    /**
4     * Chương trình đếm số từ của một chuỗi trong java
5     *
6     * @author plpsoft.vn
7     */
8    public class StringExample1 {
9        public static final char SPACE = ' ';
10       public static final char TAB = '\t';
11       public static final char BREAK_LINE = '\n';
12
13       /**
14        * main
15        *
16        * @param args
17        */
18       public static void main(String[] args) {
19           String str = "hoc java    co ban den nang cao"
20               + "        \n test";
21           System.out.print("Số từ của chuỗi đã cho là: ")

```

```

21         + countWords(str));
22     }
23
24     /**
25     * Đếm số từ của một chuỗi,
26     * giả sử các từ được ngăn cách nhau bởi một hoặc nhiều
27     * dấu 'space', tab '\t' và xuống dòng '\n'
28     *
29     * @param input - chuỗi ký tự
30     * @return số từ của chuỗi ký tự input
31     */
32     public static int countWords(String input) {
33         if (input == null) {
34             return -1;
35         }
36         int count = 0;
37         int size = input.length();
38         boolean notCounted = true;
39         for (int i = 0; i < size; i++) {
40             if (input.charAt(i) != SPACE && input.charAt(i) != TAB
41                 && input.charAt(i) != BREAK_LINE) {
42                 if (notCounted) {
43                     count++;
44                     notCounted = false;
45                 }
46             } else {
47                 notCounted = true;
48             }
49         }
50         return count;
51     }
52 }
53

```

Kết quả:

Số từ của chuỗi đã cho là: 8

Bài tập Java - Liệt kê số lần xuất hiện của các từ trong một chuỗi

Bài tập Java - Liệt kê số lần xuất hiện của các từ trong một chuỗi

Đề bài: 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

```
1 package vn.viettuts.baitap;
2
3 import java.util.Map;
4 import java.util.TreeMap;
5
6 /**
7  * Chương trình liệt kê số lần xuất hiện của các từ của một chuỗi
8  * trong java
9  *
10 * @author plpsoft.vn
11 */
12 public class StringExample2 {
13     public static final char SPACE = ' ';
14     public static final char TAB = '\t';
15     public static final char BREAK_LINE = '\n';
16
17     /**
18      * main
19      *
20      * @param args
21      */
22     public static void main(String[] args) {
23         String str = "hoc java co ban den nang cao"
24             + "\n hoc c++ co ban den nang cao.";
25         System.out.println("-----");
26         System.out.println(str);
27         System.out.println("-----");
28         // liệt kê số lần xuất hiện của các từ trong chuỗi trên
29         System.out.println("Liệt kê số lần xuất hiện của các từ: ");
30         Map<String, Integer> wordMap = countWords(str);
31         for (String key : wordMap.keySet()) {
```

```

31         System.out.print(key + " " + wordMap.get(key) + "\n");
32     }
33 }
34
35 /**
36  * Đếm số từ của một chuỗi,
37  * giả sử các từ được ngăn cách nhau bởi một hoặc nhiều
38  * dấu 'space', tab '\t' và xuống dòng '\n'
39  *
40  * @param input - chuỗi ký tự
41  * @return số từ của chuỗi ký tự input
42  */
43 public static Map<String, Integer> countWords(String input) {
44     // khởi tạo wordMap
45     Map<String, Integer> wordMap = new TreeMap<String, Integer>();
46     if (input == null) {
47         return wordMap;
48     }
49     int size = input.length();
50     StringBuilder sb = new StringBuilder();
51     for (int i = 0; i < size; i++) {
52         if (input.charAt(i) != SPACE && input.charAt(i) != TAB
53             && input.charAt(i) != BREAK_LINE) {
54             // build một từ
55             sb.append(input.charAt(i));
56         } else {
57             // thêm từ vào wordMap
58             addWord(wordMap, sb);
59             sb = new StringBuilder();
60         }
61     }
62     // thêm từ cuối cùng tìm được vào wordMap
63     addWord(wordMap, sb);
64     return wordMap;
65 }
66
67 /**
68  * Thêm từ vào wordMap
69  *
70  * @param wordMap: map chứa các từ và số lần xuất hiện
71  * @param sb: từ cần thêm vào wordMap
72  */
73 public static void addWord(Map<String, Integer> wordMap, StringBuilder sb) {
74     String word = sb.toString();
75     if (word.length() == 0) {
76         return;
77     }
78     if (wordMap.containsKey(word)) {
79         int count = wordMap.get(word) + 1;
80         wordMap.put(word, count);
81     } else {
82         wordMap.put(word, 1);
83     }
84 }
85
86

```

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.

Một bài toán có trăm nghìn cách giải. Hãy giải bài toán đó theo cách của bạn!

Bài tập java - Kiểm tra chuỗi s1 có chứa chuỗi s2 hay không?

Bài tập java - Kiểm tra chuỗi s1 có chứa chuỗi s2 hay không?

Đề bài: 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

```
1 package vn.viettuts.baitap;
2
3 public class StringExample3 {
4     public static void main(String[] args) {
5         String str1 = "hoc java co ban den nang cao.";
6         String str2 = "java co ban";
7         System.out.println(str1.contains(str2));
8     }
9 }
```

Kết quả:

```
true
```

Bài tập Java - Liệt kê các phần tử xuất hiện trong mảng đúng 1 lần

Bài tập Java - Liệt kê các phần tử xuất hiện trong mảng đúng 1 lần

Đề bài: Viết chương trình Java nhập một mảng số nguyên $a_0, a_1, a_2, \dots, a_{n-1}$. 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

```
1 package vn.viettuts.baitap.array;
2
3 import java.util.Map;
4 import java.util.Scanner;
5 import java.util.TreeMap;
6
7 /**
8  * Chương trình liệt kê số lần xuất hiện các phần tử trong một mảng
9  * nhập từ bàn phím trong java.
10  *
11  * @author plpsoft.vn
12  */
13 public class BaiTap19 {
14     public static Scanner scanner = new Scanner(System.in);
15
16     /**
17      * main
18      *
19      * @param args
20      */
21     public static void main(String[] args) {
22         System.out.print("Nhập số phần tử của mảng: ");
23         int n = scanner.nextInt();
24         // khởi tạo arr
25         int[] arr = new int[n];
26         System.out.println("Nhập các phần tử của mảng: ");
27         for (int i = 0; i < n; i++) {
28             System.out.printf("a[%d] = ", i);
29             arr[i] = scanner.nextInt();
30         }
31         // tìm số lần xuất hiện của các phần tử
```

```

31         Map<Integer, Integer> map = new TreeMap<Integer, Integer>();
32         for (inti = 0; i < n; i++) {
33             addElement(map, arr[i]);
34         }
35         System.out.print("Các phần tử xuất hiện 1 lần: ");
36         for (Integer key : map.keySet()) {
37             if (map.get(key) == 1) {
38                 System.out.print(key + " ");
39             }
40         }
41     }
42
43     /**
44     * Thêm từ vào map
45     *
46     * @param wordMap: map chứa các từ và số lần xuất hiện
47     * @param sb: từ cần thêm vào wordMap
48     */
49     public static void addElement(Map<Integer, Integer> map, int element) {
50         if (map.containsKey(element)) {
51             int count = map.get(element) + 1;
52             map.put(element, count);
53         } else {
54             map.put(element, 1);
55         }
56     }
57 }
58

```

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 tập Java - Liệt kê các phần tử xuất hiện trong mảng đúng 2 lần

Bài tập Java - Liệt kê các phần tử xuất hiện trong mảng đúng 2 lần

Đề bài: Nhập một mảng số nguyên $a_0, a_1, a_2, \dots, a_{n-1}$. 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

```
1 package vn.viettuts.baitap.array;
2
3 import java.util.Map;
4 import java.util.Scanner;
5 import java.util.TreeMap;
6
7 /**
8  * Chương trình liệt kê số lần xuất hiện các phần tử trong một mảng
9  * nhập từ bàn phím trong java.
10  *
11  * @author plpsoft.vn
12  */
13 public class BaiTap20 {
14     public static Scanner scanner = new Scanner(System.in);
15
16     /**
17      * main
18      *
19      * @param args
20      */
21     public static void main(String[] args) {
22         System.out.print("Nhập số phần tử của mảng: ");
23         int n = scanner.nextInt();
24         // khởi tạo arr
25         int[] arr = new int[n];
26         System.out.println("Nhập các phần tử của mảng: ");
27         for (int i = 0; i < n; i++) {
28             System.out.printf("a[%d] = ", i);
29             arr[i] = scanner.nextInt();
30         }
31         // tìm số lần xuất hiện của các phần tử
```

```

31         Map<Integer, Integer> map = new TreeMap<Integer, Integer>();
32         for (inti = 0; i < n; i++) {
33             addElement(map, arr[i]);
34         }
35         System.out.print("Các phần tử xuất hiện 1 lần: ");
36         for (Integer key : map.keySet()) {
37             if (map.get(key) == 2) {
38                 System.out.print(key + " ");
39             }
40         }
41     }
42
43     /**
44     * Thêm từ vào map
45     *
46     * @param wordMap: map chứa các từ và số lần xuất hiện
47     * @param sb: từ cần thêm vào wordMap
48     */
49     public static void addElement(Map<Integer, Integer> map, int element) {
50         if (map.containsKey(element)) {
51             int count = map.get(element) + 1;
52             map.put(element, count);
53         } else {
54             map.put(element, 1);
55         }
56     }
57 }
58

```

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 tập Java - Liệt kê số lần xuất hiện của các phần tử trong một mảng

Java - Liệt kê số lần xuất hiện của các phần tử trong một mảng

Đề bài: Viết chương trình nhập một mảng số nguyên $a_0, a_1, a_2, \dots, a_{n-1}$. 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

```
1 package vn.viettuts.baitap.array;
2
3 import java.util.Map;
4 import java.util.Scanner;
5 import java.util.TreeMap;
6
7 /**
8  * Chương trình liệt kê số lần xuất hiện các phần tử trong một mảng
9  * nhập từ bàn phím trong java.
10  *
11  * @author plpsoft.vn
12  */
13 public class BaiTap21 {
14     public static Scanner scanner = new Scanner(System.in);
15
16     /**
17      * main
18      *
19      * @param args
20      */
21     public static void main(String[] args) {
22         System.out.print("Nhập số phần tử của mảng: ");
23         int n = scanner.nextInt();
24         // khởi tạo arr
25         int[] arr = new int[n];
26         System.out.print("Nhập các phần tử của mảng: \n");
27         for (int i = 0; i < n; i++) {
28             System.out.printf("a[%d] = ", i);
29             arr[i] = scanner.nextInt();
30         }
31         // tìm số lần xuất hiện của các phần tử
```

```

31         Map<Integer, Integer> map = new TreeMap<Integer, Integer>();
32         for (inti = 0; i < n; i++) {
33             addElement(map, arr[i]);
34         }
35         System.out.print("Các phần tử xuất hiện 2 lần: \n");
36         for (Integer key : map.keySet()) {
37             System.out.printf("%d xuất hiện %d lần.\n", key, map.get(key));
38         }
39     }
40
41     /**
42      * Thêm từ vào map
43      *
44      * @param wordMap: map chứa các từ và số lần xuất hiện
45      * @param sb: từ cần thêm vào wordMap
46      */
47     public static void addElement(Map<Integer, Integer> map, int element) {
48         if (map.containsKey(element)) {
49             int count = map.get(element) + 1;
50             map.put(element, count);
51         } else {
52             map.put(element, 1);
53         }
54     }
55 }
56

```

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 tập Java - Sắp xếp mảng theo thứ tự tăng dần

Bài tập Java - Sắp xếp mảng theo thứ tự tăng dần

Đề bài: 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

```

1    package vn.viettuts.baitap.array;
2
3    import java.util.Scanner;
4
5    /**
6     * Chương trình sắp xếp mảng số nguyên theo thứ tự tăng dần.
7     *
8     * @author plpsoft.vn
9     */
10   public class BaiTap24 {
11       public static Scanner scanner = new Scanner(System.in);
12
13       /**
14        * main
15        *
16        * @param args
17        */
18       public static void main(String[] args) {
19           System.out.print("Nhập số phần tử của mảng: ");
20           int n = scanner.nextInt();
21           // khởi tạo arr
22           int [] arr = new int [n];
23           System.out.print("Nhập các phần tử của mảng: \n");
24           for (int i = 0; i < n; i++) {
25               System.out.printf("a[%d] = ", i);
26               arr[i] = scanner.nextInt();
27           }
28           // sắp xếp dãy số theo thứ tự tăng dần
29           sortASC(arr);
30           System.out.println("Dãy số được sắp xếp tăng dần: ");
31           show(arr);
32       }
33   }

```

```

34      * sắp xếp mảng số nguyên theo thứ tự tăng dần
35      *
36      * @param arr: mảng các số nguyên
37      * @param n: số phần tử của mảng
38      */
39      public static void sortASC(int [] arr) {
40          int temp = arr[0];
41          for (int i = 0; i < arr.length - 1; i++) {
42              for (int j = i + 1; j < arr.length; j++) {
43                  if (arr[i] > arr[j]) {
44                      temp = arr[j];
45                      arr[j] = arr[i];
46                      arr[i] = temp;
47                  }
48              }
49          }
50      }
51
52      /**
53       * in các phần tử của mảng ra màn hình
54       *
55       * @param arr: mảng các số nguyên
56       * @param n: số phần tử của mảng
57       */
58      public static void show(int [] arr) {
59          for (int i = 0; i < arr.length; i++) {
60              System.out.print(arr[i] + " ");
61          }
62      }
63  }
64

```

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 tập Java - Sắp xếp mảng theo thứ tự giảm dần

Bài tập Java - Sắp xếp mảng theo thứ tự giảm dần

Đề bài: Viết chương trình Java nhập một mảng số nguyên $a_0, a_1, a_2, \dots, a_{n-1}$. 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

```

1      package vn.viettuts.baitap.array;
2
3      import java.util.Scanner;
4
5      /**
6       * Chương trình sắp xếp mảng số nguyên theo thứ tự giảm dần.
7       *
8       * @author plpsoft.vn
9       */
10     public class BaiTap23 {
11         public static Scanner scanner = new Scanner(System.in);
12
13         /**
14          * main
15          *
16          * @param args
17          */
18         public static void main(String[] args) {
19             System.out.print("Nhập số phần tử của mảng: ");
20             int n = scanner.nextInt();
21             // khởi tạo arr
22             int[] arr = new int[n];
23             System.out.print("Nhập các phần tử của mảng: \n");
24             for (int i = 0; i < n; i++) {
25                 System.out.printf("a[%d] = ", i);
26                 arr[i] = scanner.nextInt();
27             }
28             // sắp xếp dãy số theo thứ tự giảm dần
29             sortDESC(arr);
30             System.out.println("Dãy số được sắp xếp giảm dần: ");
31             show(arr);
32         }
33     }

```

```

34      * sắp xếp mảng số nguyên theo thứ tự giảm dần
35      *
36      * @param arr: mảng các số nguyên
37      * @param n: số phần tử của mảng
38      */
39      public static void sortDESC(int [] arr) {
40          int temp = arr[0];
41          for (int i = 0; i < arr.length - 1; i++) {
42              for (int j = i + 1; j < arr.length; j++) {
43                  if (arr[i] < arr[j]) {
44                      temp = arr[j];
45                      arr[j] = arr[i];
46                      arr[i] = temp;
47                  }
48              }
49          }
50      }
51
52      /**
53       * in các phần tử của mảng ra màn hình
54       *
55       * @param arr: mảng các số nguyên
56       * @param n: số phần tử của mảng
57       */
58      public static void show(int [] arr) {
59          for (int i = 0; i < arr.length; i++) {
60              System.out.print(arr[i] + " ");
61          }
62      }
63  }
64

```

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

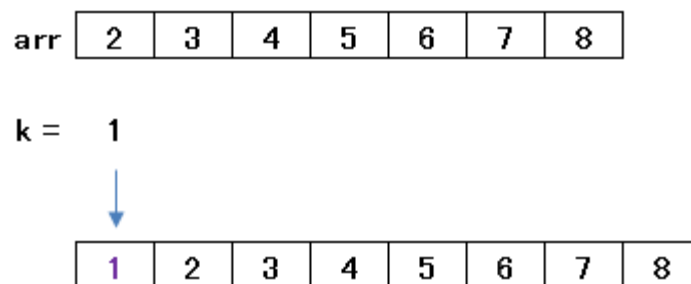
Bài tập Java - Chèn phần tử vào mảng trong java

Bài tập Java - Chèn phần tử vào mảng trong Java

Đề bài: Viết chương trình Java nhập một mảng số nguyên $a_0, a_1, a_2, \dots, a_{n-1}$. 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

```

1  package vn.viettuts.baitap.array;
2
3  import java.util.Scanner;
4
5  /**
6   * Chương trình sắp xếp mảng theo thứ tự tăng dần,
7   * sau đó chèn phần tử k vào mà vẫn đảm bảo mảng là tăng dần.
8   *
9   * @author plpsoft.vn
10  */
11 public class BaiTap24 {
12     public static Scanner scanner = new Scanner(System.in);
13
14     /**
15      * main
16      *
17      * @param args
18      */
19     public static void main(String[] args) {
20         System.out.print("Nhập số phần tử của mảng: ");
21         int n = scanner.nextInt();
22         // khởi tạo arr
    
```

```

22         int [] arr = new int [n];
23         System.out.print("Nhập các phần tử của mảng: \n");
24         for (inti = 0; i < n; i++) {
25             System.out.printf("a[%d] = ", i);
26             arr[i] = scanner.nextInt();
27         }
28         System.out.print("Nhập phần tử k = ");
29         int k = scanner.nextInt();
30         // sắp xếp dãy số theo thứ tự tăng dần
31         sortASC(arr);
32         System.out.print("Sắp xếp mảng tăng dần: ");
33         show(arr);
34         System.out.printf("\nChèn phần tử %d vào mảng.", k);
35         arr = insert(arr, k);
36         System.out.print("\nMảng sau khi chèn: ");
37         show(arr);
38     }
39
40     /**
41     * sắp xếp mảng số nguyên theo thứ tự tăng dần
42     *
43     * @param arr: mảng các số nguyên
44     */
45     public static void sortASC(int [] arr) {
46         int temp = arr[0];
47         for (inti = 0; i < arr.length - 1; i++) {
48             for (intj = i + 1; j < arr.length; j++) {
49                 if (arr[i] > arr[j]) {
50                     temp = arr[j];
51                     arr[j] = arr[i];
52                     arr[i] = temp;
53                 }
54             }
55         }
56     }
57
58     /**
59     * chèn phần tử vào mảng số nguyên tăng dần
60     * sau khi chèn mảng vẫn duy trì thứ tự tăng dần
61     *
62     * @param arr: mảng số nguyên tăng dần
63     * @param k: phần tử chèn vào mảng arr
64     */
65     public static int [] insert(int [] arr, int k) {
66         int arrIndex = arr.length - 1;
67         int tempIndex = arr.length;
68         int [] tempArr = new int [tempIndex + 1];
69         boolean inserted = false;
70
71         for (inti = tempIndex; i >= 0; i--) {
72             if (arrIndex > -1 && arr[arrIndex] > k) {
73                 tempArr[i] = arr[arrIndex--];
74             } else {
75                 if (!inserted) {
76                     tempArr[i] = k;
77                     inserted = true;
78                 } else {
79                     tempArr[i] = arr[arrIndex--];
80                 }
81             }
82         }

```

```

82         }
83         return tempArr;
84     }
85
86     /**
87      * in các phần tử của mảng ra màn hình
88      *
89      * @param arr: mảng các số nguyên
90      */
91     public static void show(int [] arr) {
92         for (int i = 0; i < arr.length; i++) {
93             System.out.print(arr[i] + " ");
94         }
95     }
96 }
97
98

```

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

```

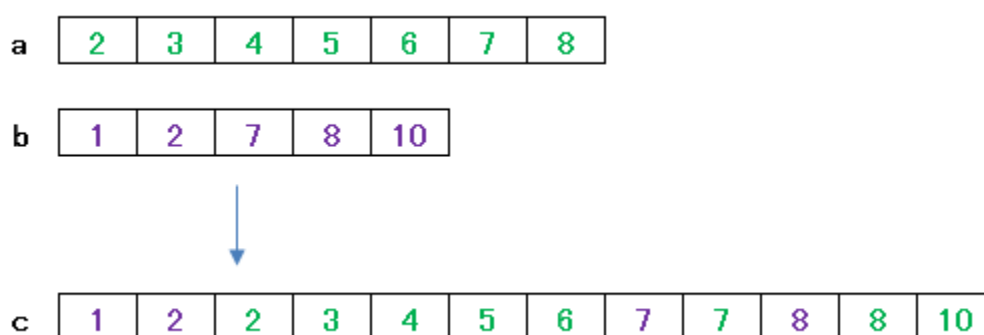
Bài tập Java - Trộn 2 mảng trong Java

Bài tập Java - Trộn 2 mảng trong java

Đề bài: Viết chương trình Java nhập 2 mảng số thực $a_0, a_1, a_2, \dots, a_n$ và $b_0, b_1, b_2, \dots, b_m$. Giả sử 2 mảng này đã được sắp xếp tăng dần. Hãy tận dụng tính sắp xếp của 2 dãy và tạo dãy $c_0, c_1, c_2, \dots, c_{n+m}$ là hợp của 2 dãy trên sao cho c_i cũng có thứ tự tăng dần.

Lời giải

Trộn 2 mảng trong java.



File: BaiTap24.java

```

1      package vn.viettuts.baitap.array;
2
3      import java.util.Scanner;
4
5      /**
6       * Chương trình mảng số thực a0, a1, a2, ..., am-1 và b0, b1, b2, ..., bn-1.
7       * Giả sử 2 mảng này đã được sắp xếp tăng dần.
8       * Hãy tận dụng tính sắp xếp của 2 dãy và tạo dãy c0, c1, c2, ..., cm+n-1
9       * là hợp của 2 dãy trên sao cho ci cũng có thứ tự tăng dần
10      *
11      * @author plpsoft.vn
12      */
13     public class BaiTap26 {
14         public static Scanner scanner = new Scanner(System.in);
15
16         /**
17          * main
18          *
19          * @param args
20          */
21         public static void main(String[] args) {

```



```

22         float[] a = null;
23         float[] b = null;
24         float[] c = null;
25
26         System.out.println("---Nhập mảng a---");
27         a = input(a);
28         System.out.println("---Nhập mảng b---");
29         b = input(b);
30
31         // trộn mảng a và b thành c
32         c = merge(a, b);
33
34         // in mảng c ra màn hình
35         show(c);
36     }
37
38     /**
39     * nhập mảng
40     *
41     * @param arr: mảng số nguyên
42     */
43     public static float[] input(float[] arr) {
44         System.out.print("Nhập số phần tử của mảng: ");
45         int n = scanner.nextInt();
46         // khởi tạo arr
47         arr = new float[n];
48         System.out.print("Nhập các phần tử của mảng: \n");
49         for (int i = 0; i < n; i++) {
50             System.out.printf("a[%d] = ", i);
51             arr[i] = scanner.nextInt();
52         }
53         return arr;
54     }
55
56     /**
57     * sắp xếp mảng số nguyên theo thứ tự tăng dần
58     *
59     * @param arr: mảng các số nguyên
60     * @param n: số phần tử của mảng
61     */
62     public static void sortASC(float [] arr) {
63         float temp = arr[0];
64         for (int i = 0; i < arr.length - 1; i++) {
65             for (int j = i + 1; j < arr.length; j++) {
66                 if (arr[i] > arr[j]) {
67                     temp = arr[j];
68                     arr[j] = arr[i];
69                     arr[i] = temp;
70                 }
71             }
72         }
73     }
74
75     /**
76     * trộn mảng a và b vào mảng c
77     * sao cho c cũng duy trì thứ tự tăng dần.
78     *
79     * @param a: mảng a
80     * @param b: mảng b
81     * @return mảng c
82     */

```

```

82         public static float[] merge(float[] a, float[] b) {
83             int aIndex = a.length - 1;
84             int bIndex = b.length - 1;
85             int cIndex = a.length + b.length - 1;
86             float[] c = new float[cIndex + 1];
87
88             // sắp xếp các mảng theo thứ tự tăng dần
89             sortASC(a);
90             sortASC(b);
91
92             // trộn mảng a và b thành c
93             for (int i = cIndex; i > -1; i--) {
94                 if (aIndex > -1 && bIndex > -1) {
95                     if (a[aIndex] > b[bIndex]) {
96                         c[i] = a[aIndex--];
97                     } else {
98                         c[i] = b[bIndex--];
99                     }
100                } else if (bIndex == -1) {
101                    c[i] = a[aIndex--];
102                } else if (aIndex == -1) {
103                    c[i] = b[bIndex--];
104                }
105            }
106            return c;
107        }
108
109        /**
110         * in các phần tử của mảng ra màn hình
111         *
112         * @param arr: mảng các số nguyên
113         * @param n: số phần tử của mảng
114         */
115        public static void show(float[] arr) {
116            for (int i = 0; i < arr.length; i++) {
117                System.out.print(arr[i] + " ");
118            }
119        }
120    }
121

```

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

5. Bài tập quản lý sinh viên trong Java

Lập trình hướng đối tượng (OOPs)

Nội dung chính

- [Bài tập quản lý sinh viên trong Java](#)
- [Lời giải](#)
- [Cấu trúc của project](#)
- [1. Tạo lớp Student.java](#)
- [2. Tạo lớp StudentDao.java](#)
- [3. Tạo lớp SortStudentByGPA.java](#)
- [4. Tạo lớp SortStudentByName.java](#)
- [5. Tạo lớp StudentManager.java](#)
- [6. Tạo lớp Main.java](#)
- [Run bài tập quản lý sinh viên trong java](#)

Bài tập quản lý sinh viên trong Java

Đề bài: Viết chương trình quản lý sinh viên. Mỗi đối tượng sinh viên có các thuộc tính sau: id, name, age, address và gpa (điểm trung bình). Yêu cầu: tạo ra một menu với các chức năng sau:

```

/*****/
1. Add student.
2. Edit student by id.
3. Delete student by id.
4. Sort student by gpa.
5. Sort student by name.
6. Show student.
0. Exit.
/*****/
    
```

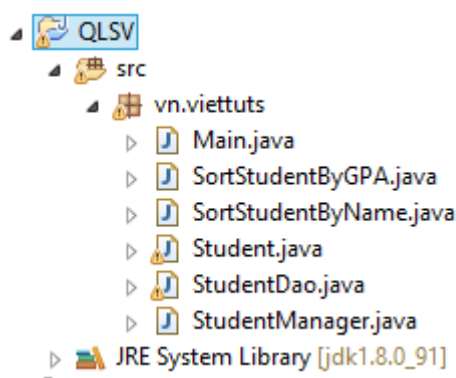
Yêu cầu thêm: list sinh viên được lưu vào file "**student.txt**" hoặc cơ sở dữ liệu.

Lời giải

Các bạn lưu ý: trước khi xem lời giải các bạn hãy tự làm trước nhé. Hãy coi lời giải này như một bài tham khảo.

Cấu trúc của project

Cấu trúc của project được tạo trên **eclipse**:



Trong đó:

- Lớp Student để lưu thông tin cho mỗi sinh viên.
- Lớp StudentDao để đọc và ghi sinh viên vào file.
- Lớp SortStudentByGPA được implements Comparator để sắp xếp sinh viên tăng dần theo điểm trung bình.
- Lớp SortStudentByName được implements Comparator để sắp xếp sinh viên tăng dần theo tên.
- Lớp StudentManager cung cấp các phương thức để quản lý sinh viên như thêm, sửa, xóa, sắp xếp và hiển thị sinh viên.
- Lớp Main chứa phương thức public static void main() để chạy ứng dụng và menu như yêu cầu của bài toán.

1. Tạo lớp Student.java

Lớp này để lưu thông tin cho mỗi sinh viên.

File: Student.java

```
package vn.viettuts;

import java.io.Serializable;

/**
 * Student class
 *
 * @author plpsoft.vn
 */
public class Student implements Serializable {
    private int id;
    private String name;
    private byte age;
    private String address;
    /* điểm trung bình của sinh viên */
    private float gpa;
}
```

```

public Student() {
}

public Student(int id, String name, byte age,
               String address, float gpa) {
    super();
    this.id = id;
    this.name = name;
    this.age = age;
    this.address = address;
    this.gpa = gpa;
}

public int getId() {
    return id;
}

public void setId(int id) {
    this.id = id;
}

public String getName() {
    return name;
}

public void setName(String name) {
    this.name = name;
}

public byte getAge() {
    return age;
}

public void setAge(byte age) {
    this.age = age;
}

public String getAddress() {
    return address;
}

public void setAddress(String address) {
    this.address = address;
}

public float getGpa() {
    return gpa;
}

public void setGpa(float gpa) {
    this.gpa = gpa;
}
}

```

2. Tạo lớp StudentDao.java

Tạo file "**student.txt**" tại thư mục gốc của dự án để lưu danh sách sinh viên.

wsjavaweb > QLSV		Search QLSV	
Name	Date modified	Type	Size
bin	9/29/2017 10:07 AM	File folder	
src	9/28/2017 4:11 PM	File folder	
.classpath	9/28/2017 3:47 PM	CLASSPATH File	1 KB
.project	9/28/2017 3:47 PM	PROJECT File	1 KB
student.txt	9/29/2017 10:56 AM	Text Document	1 KB

Trong trường hợp này chúng ta sử dụng file để lưu trữ và truy xuất các đối tượng sinh viên. Nên lớp **Student** phải được implements [Serializable](#).

Lớp **StudentDao.java** chứa phương thức **read()** để đọc thông tin danh sách sinh viên từ file "student.txt" và phương thức **write()** để ghi thông tin danh sách sinh viên vào file.

Phương thức **read()** sử dụng đối tượng [ObjectInputStream trong java](#) để đọc danh sách sinh viên từ file.

Phương thức **write()** sử dụng đối tượng [ObjectOutputStream trong java](#) để ghi danh sách sinh viên vào file.

File: StudentDao.java

```
package vn.viettuts;

import java.io.File;
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStream;
import java.io.ObjectInputStream;
import java.io.ObjectOutputStream;
import java.io.OutputStream;
import java.util.ArrayList;
import java.util.List;

/**
 * StudentDao class
 *
 * @author plpsoft.vn
 */
public class StudentDao {
    private static final String STUDENT_FILE_NAME = "student.txt";

    /**
     * save list student to file
     *
     * @param studentList: list student to save
     */
}
```

```

    */
    public void write(List<Student> studentList) {
        FileOutputStream fos = null;
        ObjectOutputStream oos = null;
        try {
            fos = new FileOutputStream(new File(STUDENT_FILE_NAME));
            oos = new ObjectOutputStream(fos);
            oos.writeObject(studentList);
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        } finally {
            closeStream(fos);
            closeStream(oos);
        }
    }

    /**
     * read list student from file
     *
     * @return list student
     */
    public List<Student> read() {
        List<Student> studentList = new ArrayList<>();
        FileInputStream fis = null;
        ObjectInputStream ois = null;
        try {
            fis = new FileInputStream(new File(STUDENT_FILE_NAME));
            ois = new ObjectInputStream(fis);
            studentList = (List<Student>) ois.readObject();
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        } catch (ClassNotFoundException e) {
            e.printStackTrace();
        } finally {
            closeStream(fis);
            closeStream(ois);
        }
        return studentList;
    }

    /**
     * close input stream
     *
     * @param is: input stream
     */
    private void closeStream(InputStream is) {
        if (is != null) {
            try {
                is.close();
            } catch (IOException e) {
                e.printStackTrace();
            }
        }
    }

```



```

        }
    }
}

/**
 * close output stream
 *
 * @param os: output stream
 */
private void closeStream(OutputStream os) {
    if (os != null) {
        try {
            os.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
}

```

3. Tạo lớp SortStudentByGPA.java

Lớp SortStudentByGPA implements Comparator được sử dụng để sắp xếp sinh viên tăng dần theo điểm trung bình. Tìm hiểu thêm về cách sử dụng [Comparator trong java](#).

File: SortStudentByGPA.java

```

package vn.viettuts;

import java.util.Comparator;

/**
 * SortStudentByGPA class
 *
 * @author plpsoft.vn
 */
public class SortStudentByGPA implements Comparator<Student> {
    @Override
    public int compare(Student student1, Student student2) {
        if (student1.getGpa() > student2.getGpa()) {
            return 1;
        }
        return -1;
    }
}

```

4. Tạo lớp SortStudentByName.java

Lớp `SortStudentByName` implements `Comparator` được sử dụng để sắp xếp sinh viên tăng dần theo tên.

File: `SortStudentByName.java`

```
package vn.viettuts;

import java.util.Comparator;

/**
 * SortStudentByName class
 *
 * @author plpsoft.vn
 */
public class SortStudentByName implements Comparator<Student> {
    @Override
    public int compare(Student student1, Student student2) {
        return student1.getName().compareTo(student2.getName());
    }
}
```

5. Tạo lớp `StudentManager.java`

Lớp này cung cấp các phương thức để thêm, sửa, xóa, sắp xếp và hiển thị sinh viên.

```
package vn.viettuts;
import java.util.Collections;
import java.util.List;
import java.util.Scanner;
/**
 * StudentManager class
 *
 * @author plpsoft.vn
 */
public class StudentManager {
    public static Scanner scanner = new Scanner(System.in);
    private List<Student> studentList;
    private StudentDao studentDao;
    /**
     * init StudentDao object and
     * read list student when init StudentManager object
     */
    public StudentManager() {
        studentDao = new StudentDao();
        studentList = studentDao.read();
    }

    /**
     * add student to studentList
     *
     * @param student
     */
    public void add() {
        int id = (studentList.size() > 0) ? (studentList.size() + 1):1;
```

```

        System.out.println("student id = " + id);
        String name = inputName();
        byte age = inputAge();
        String address = inputAddress();
        float gpa = inputGpa();
        Student student = new Student(id, name, age, address, gpa);
        studentList.add(student);
        studentDao.write(studentList);
    }
    /**
     * edit student by id
     *
     * @param id
     */
    public void edit(int id) {
        boolean isExisted = false;
        int size = studentList.size();
        for (int i = 0; i < size; i++) {
            if (studentList.get(i).getId() == id) {
                isExisted = true;
                studentList.get(i).setName(inputName());
                studentList.get(i).setAge(inputAge());
                studentList.get(i).setAddress(inputAddress());
                studentList.get(i).setGpa(inputGpa());
                break;
            }
        }
        if (!isExisted) {
            System.out.printf("id = %d not existed.\n", id);
        } else {
            studentDao.write(studentList);
        }
    }
    /**
     * delete student by id
     *
     * @param id: student id
     */
    public void delete(int id) {
        Student student = null;
        int size = studentList.size();
        for (int i = 0; i < size; i++) {
            if (studentList.get(i).getId() == id) {
                student = studentList.get(i);
                break;
            }
        }
        if (student != null) {
            studentList.remove(student);
            studentDao.write(studentList);
        } else {
            System.out.printf("id = %d not existed.\n", id);
        }
    }
    /**
     * sort student by name

```

```

    */
    public void sortStudentByName() {
        Collections.sort(studentList, new SortStudentByName());
    }
    /**
     * sort student by id
     */
    public void sortStudentByGPA() {
        Collections.sort(studentList, new SortStudentByGPA());
    }
    /**
     * show list student to screen
     */
    public void show() {
        for (Student student : studentList) {
            System.out.format("%5d | ", student.getId());
            System.out.format("%20s | ", student.getName());
            System.out.format("%5d | ", student.getAge());
            System.out.format("%20s | ", student.getAddress());
            System.out.format("%10.1f%n", student.getGpa());
        }
    }
    /**
     * input student id
     *
     * @return student id
     */
    public int inputId() {
        System.out.print("Input student id: ");
        while (true) {
            try {
                int id = Integer.parseInt(scanner.nextLine());
                return id;
            } catch (NumberFormatException ex) {
                System.out.print("invalid! Input student id again: ");
            }
        }
    }

    /**
     * input student name
     *
     * @return student name
     */
    private String inputName() {
        System.out.print("Input student name: ");
        return scanner.nextLine();
    }

    /**
     * input student address
     *
     * @return student address
     */
    private String inputAddress() {

```

```

        System.out.print("Input student address: ");
        return scanner.nextLine();
    }

    /**
     * input student age
     *
     * @return student age
     */
    private byte inputAge() {
        System.out.print("Input student age: ");
        while (true) {
            try {
                byte age = Byte.parseByte((scanner.nextLine()));
                if (age < 0 && age > 100) {
                    throw new NumberFormatException();
                }
                return age;
            } catch (NumberFormatException ex) {
                System.out.print("invalid! Input student id again: ");
            }
        }
    }

    /**
     * input student gpa
     *
     * @return gpa
     */
    private float inputGpa() {
        System.out.print("Input student gpa: ");
        while (true) {
            try {
                float gpa = Float.parseFloat((scanner.nextLine()));
                if (gpa < 0.0 && gpa > 10.0) {
                    throw new NumberFormatException();
                }
                return gpa;
            } catch (NumberFormatException ex) {
                System.out.print("invalid! Input student age again: ");
            }
        }
    }

    // getter && setter
    public List<Student> getStudentList() {
        return studentList;
    }

    public void setStudentList(List<Student> studentList) {
        this.studentList = studentList;
    }
}

```

6. Tạo lớp Main.java

Lớp này chứa phương thức main(), định nghĩa menu.

```
package vn.viettuts;
import java.util.Scanner;

/**
 * Main class
 *
 * @author plpsoft.vn
 */
public class Main {
    public static Scanner scanner = new Scanner(System.in);
    public static void main(String[] args) {
        String choose = null;
        boolean exit = false;
        StudentManager studentManager = new StudentManager();
        int studentId;

        // show menu
        showMenu();
        while (true) {
            choose = scanner.nextLine();
            switch (choose) {
                case "1":
                    studentManager.add();
                    break;
                case "2":
                    studentId = studentManager.inputId();
                    studentManager.edit(studentId);
                    break;
                case "3":
                    studentId = studentManager.inputId();
                    studentManager.delete(studentId);
                    break;
                case "4":
                    studentManager.sortStudentByGPA();
                    break;
                case "5":
                    studentManager.sortStudentByName();
                    break;
                case "6":
                    studentManager.show();
                    break;
                case "0":
                    System.out.println("exited!");
                    exit = true;
                    break;
                default:
                    System.out.println("invalid! please choose action
in below menu:");
                    break;
            }
            if (exit) {
```

```

        break;
    }
    // show menu
    showMenu();
}

/**
 * create menu
 */
public static void showMenu() {
    System.out.println("-----menu-----");
    System.out.println("1. Add student.");
    System.out.println("2. Edit student by id.");
    System.out.println("3. Delete student by id.");
    System.out.println("4. Sort student by gpa.");
    System.out.println("5. Sort student by name.");
    System.out.println("6. Show student.");
    System.out.println("0. exit.");
    System.out.println("-----");
    System.out.print("Please choose: ");
}
}

```

7. Run bài tập quản lý sinh viên trong java

Kết quả:

```

-----menu-----
1. Add student.
2. Edit student by id.
3. Delete student by id.
4. Sort student by gpa.
5. Sort student by name.
6. Show student.
0. exit.
-----
Please choose: 1
student id = 1
Input student name: Vu Van Vinh
Input student age: 22
Input student address: Ha Noi
Input student gpa: 8
-----menu-----

```

```

1. Add student.
2. Edit student by id.
3. Delete student by id.
4. Sort student by gpa.
5. Sort student by name.
6. Show student.
0. exit.
-----

Please choose: 1
student id = 2
Input student name: Pham Ba Hao
Input student age: 21
Input student address: Vinh Phuc
Input student gpa: 9
-----menu-----

1. Add student.
2. Edit student by id.
3. Delete student by id.
4. Sort student by gpa.
5. Sort student by name.
6. Show student.
0. exit.
-----

Please choose: 6
      1 | Vu Van Vinh |      22 |              Ha Noi |              8.0
      2 | Pham Ba Hao |      21 |              Vinh Phuc |              9.0
-----menu-----

1. Add student.
2. Edit student by id.
3. Delete student by id.
4. Sort student by gpa.
5. Sort student by name.
6. Show student.
0. exit.
-----

```



```

Please choose: 5
-----menu-----
1. Add student.
2. Edit student by id.
3. Delete student by id.
4. Sort student by gpa.
5. Sort student by name.
6. Show student.
0. exit.
-----

Please choose: 6
      2 |      Pham Ba Hao |      21 |      Vinh Phuc |      9.0
      1 |      Vu Van Vinh |      22 |      Ha Noi |      8.0
-----menu-----
1. Add student.
2. Edit student by id.
3. Delete student by id.
4. Sort student by gpa.
5. Sort student by name.
6. Show student.
0. exit.
-----

Please choose: 0
exited!
    
```

Nếu bạn quan tâm bạn có thể mở rộng bài này bằng cách:

- Dụng giao diện cho bài này bằng Swing, JavaFX hoặc JSP.
- Sử dụng cơ sở dữ liệu để lưu sinh viên.
- Thêm chức năng tìm kiếm sinh viên.

TÀI LIỆU THAM KHẢO

- [1] Giáo trình, bài giảng “Kỹ năng lập trình ứng dụng với Java”, Ths. Võ Văn Phúc, Trường đại học Nam Cần Thơ
- [2] Giáo trình, bài giảng “Lập trình Java căn bản””, Ths. Võ Văn Phúc, Trường đại học Nam Cần Thơ
- [3] Allen B. Downey & Chris Mayfield, Think Java, 2016
- [4] Lập Trình Hướng Đối Tượng Với Java – Đoàn Văn Ban, NXB KHOA HỌC VÀ KỸ THUẬT, 2005
- [5] Java Cơ Bản – Nguyễn Văn Khoa, NXB Hồng Đức, 2007
- [6] Java Core (Tiếng Việt), UDS Ebook, updateSoft.
- [7] Slide Lập Trình Java - Phạm Quang Dũng, ĐH KHTN
- [8] Lập trình hướng đối tượng – ĐH Công nghệ, ĐHQGHN
- [9] Lập Trình Java: <https://viettuts.vn/java>

Tài liệu tham khảo khác:

- [10] Cay S. Horstmann, *Core Java Volume I*, Prentice Hall, 2016.
- [11] Java Tutorial, <https://www.tutorialspoint.com/java/index.htm>
- [12] Java cơ bản (o7planning.org): <https://o7planning.org/vi/10973/java-co-ban>
- [13] Java cơ bản (vncoder.vn) <https://vncoder.vn/java/lap-trinh-java-co-ban>
- [14] Lập trình Java: <https://yellowcodebooks.com/category/lap-trinh-java/>
- [15] Java nâng cao (o7planning.org): <https://o7planning.org/vi/10985/java-nang-cao>

CÁC LINK BÀI TẬP THAM KHẢO

- [16] 247 bài tập:
https://vietjack.com/bai_tap_java/bai_tap_mau_java_va_vi_du_java.jsp
- [17] 200 câu hỏi phỏng vấn Java:
https://vietjack.com/cau_hoi_phong_van_java/index.jsp
- [18] 200 câu hỏi phỏng vấn Java: <https://plpsoft.vn/interview/list-cau-hoi-phong-van-java-core>