

# Nội Dung

Author: Nguyễn Văn Tuyên  
Email: [nguyenvantuyen6789@gmail.com](mailto:nguyenvantuyen6789@gmail.com)  
Phone: 0888.335.333  
Fb: [facebook.com/nguyenvantuyen6789](https://facebook.com/nguyenvantuyen6789)

1. Create Date

2. Sort Integer, String, Date

3. Sort Object By Property

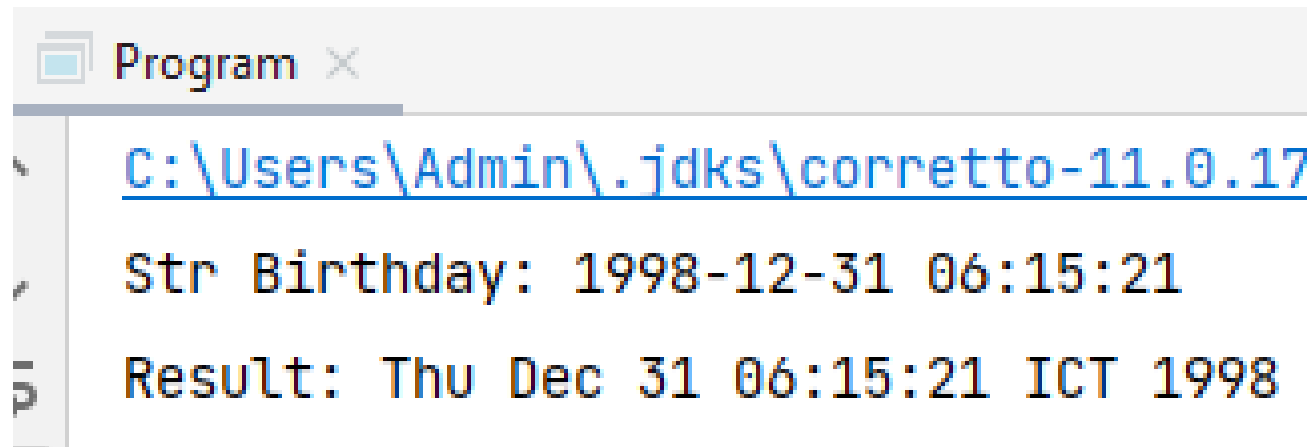
# 1. Create Date

## Create Date

```
Program.java x
77 ▶ public static void main(String[] args) {
78     String strBirthday = "1998-12-31 06:15:21";
79     Date date1 = null;
80     try {
81         date1 = new SimpleDateFormat( pattern: "yyyy-MM-dd hh:mm:ss").parse(strBirthday);
82     } catch (ParseException e) {
83         e.printStackTrace();
84     }
85
86     System.out.println("Str Birthday: " + strBirthday);
87     System.out.println("Result: " + date1);
88 }
```

# 1. Create Date

Kết Quả



```
C:\Users\Admin\.jdk\corretto-11.0.17
Str Birthday: 1998-12-31 06:15:21
Result: Thu Dec 31 06:15:21 ICT 1998
```

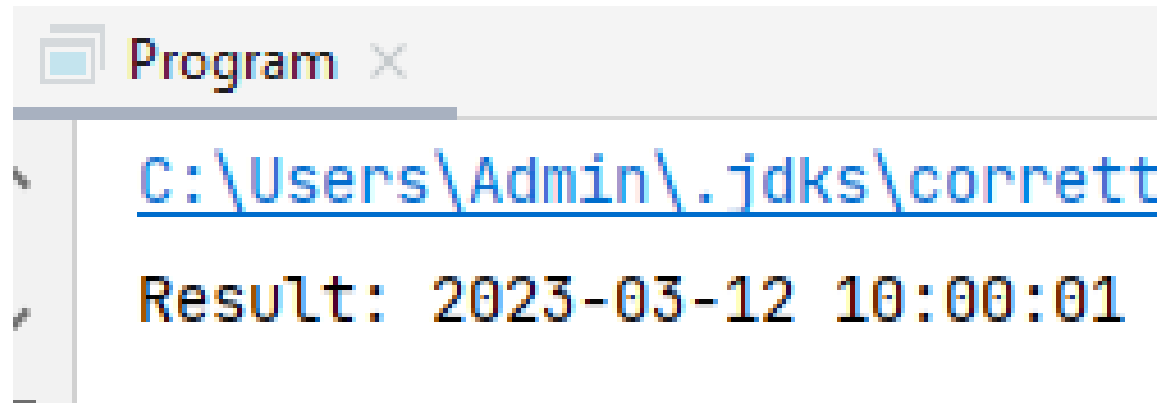
# 1. Create Date

## Create Current Date

```
Program.java x
92 ▶ public static void main(String[] args) {
93     Date dNow = new Date();
94     SimpleDateFormat format = new SimpleDateFormat ( pattern: "yyyy-MM-dd hh:mm:ss");
95
96     System.out.println("Result: " + format.format(dNow));
97 }
```

# 1. Create Date

Result



```
Program x
C:\Users\Admin\.jdk\corrett
Result: 2023-03-12 10:00:01
```

The image shows a screenshot of a command prompt window. The title bar at the top reads "Program" followed by a close button icon. The command prompt displays a file path `C:\Users\Admin\.jdk\corrett` on the first line, which is underlined. On the second line, it displays the output `Result: 2023-03-12 10:00:01`. The text is in a monospaced font, and the background is white.

## 2. Sort Integer, String, Date

Sort: Integer, String, Date

Integer

```
Program.java x
54 public static void main(String[] args) {
55     List<Integer> list = Arrays.asList(5, 2, 7, 1, 3);
56     System.out.println("Input: " + list);
57
58     Collections.sort(list);
59     System.out.println("Output: " + list);
60 }
```

```
Program x
C:\Users\Admin\.jdk\cor
Input: [5, 2, 7, 1, 3]
Output: [1, 2, 3, 5, 7]
```

## 2. Sort Integer, String, Date

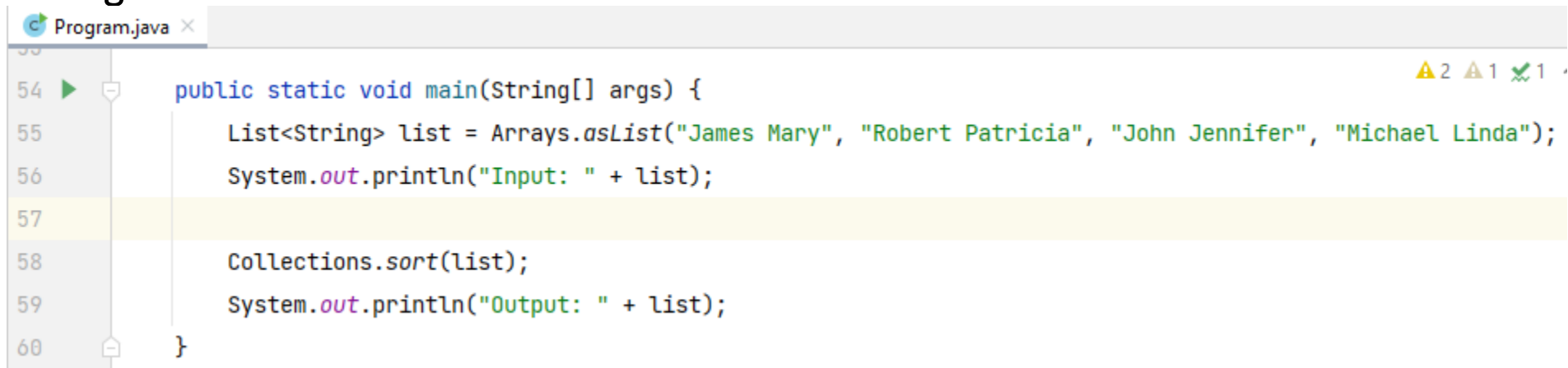
### Sort Decending

```
Comparator<Integer> reverse = Collections.reverseOrder();  
Collections.sort(list, reverse);
```

## 2. Sort Integer, String, Date

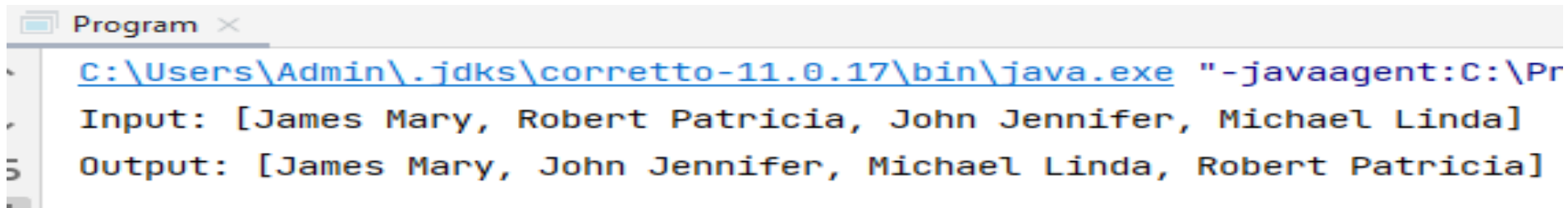
Sort: Integer, String, Date

String



```
Program.java x
54 public static void main(String[] args) {
55     List<String> list = Arrays.asList("James Mary", "Robert Patricia", "John Jennifer", "Michael Linda");
56     System.out.println("Input: " + list);
57
58     Collections.sort(list);
59     System.out.println("Output: " + list);
60 }
```

The screenshot shows a Java IDE with a file named 'Program.java'. The code defines a 'main' method that creates a list of four strings: "James Mary", "Robert Patricia", "John Jennifer", and "Michael Linda". It prints the input list, sorts it using 'Collections.sort()', and prints the output list. The output list is sorted alphabetically: "James Mary", "John Jennifer", "Michael Linda", and "Robert Patricia".



```
Program x
C:\Users\Admin\.jdk\corretto-11.0.17\bin\java.exe "-javaagent:C:\Pr
Input: [James Mary, Robert Patricia, John Jennifer, Michael Linda]
Output: [James Mary, John Jennifer, Michael Linda, Robert Patricia]
```

The screenshot shows a command prompt window titled 'Program'. It displays the command to run the Java program: 'C:\Users\Admin\.jdk\corretto-11.0.17\bin\java.exe "-javaagent:C:\Pr'. The output of the program is shown below the command: 'Input: [James Mary, Robert Patricia, John Jennifer, Michael Linda]' and 'Output: [James Mary, John Jennifer, Michael Linda, Robert Patricia]'. The output list is sorted alphabetically.



## 2. Sort Integer, String, Date

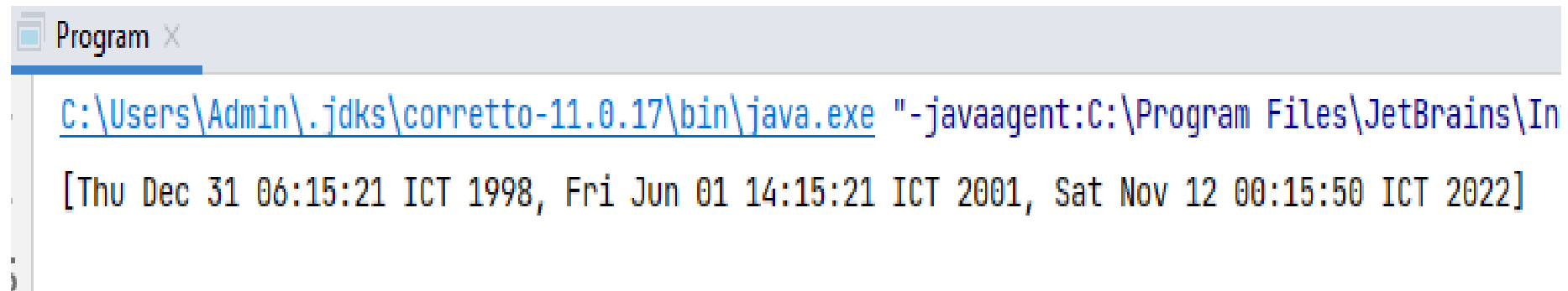
### Sort: Integer, String, Date

#### Date

```
Program.java x
105 public static void main(String[] args) {
106     String strDate1 = "1998-12-31 06:15:21";
107     String strDate2 = "2022-11-12 00:15:50";
108     String strDate3 = "2001-06-01 14:15:21";
109     Date date1 = null;
110     Date date2 = null;
111     Date date3 = null;
112     try {
113         date1 = new SimpleDateFormat(pattern: "yyyy-MM-dd hh:mm:ss").parse(strDate1);
114         date2 = new SimpleDateFormat(pattern: "yyyy-MM-dd hh:mm:ss").parse(strDate2);
115         date3 = new SimpleDateFormat(pattern: "yyyy-MM-dd hh:mm:ss").parse(strDate3);
116     } catch (ParseException e) {
117         e.printStackTrace();
118     }
119
120     List<Date> listDate = Arrays.asList(date1, date2, date3);
121     Collections.sort(listDate);
122
123     System.out.println(listDate);
124 }
```

## 2. Sort Integer, String, Date

Kết Quả



```
Program x
C:\Users\Admin\.jdk\corretto-11.0.17\bin\java.exe "-javaagent:C:\Program Files\JetBrains\In
[Thu Dec 31 06:15:21 ICT 1998, Fri Jun 01 14:15:21 ICT 2001, Sat Nov 12 00:15:50 ICT 2022]
;
```

# 3. Sort Object By Property

## Sort By fullName

```
List<Person> people = new ArrayList<>();

Person person = new Person( id: 1, username: "t1", password: "123", fullName: "Oliver", age: 20);
people.add(person);

person = new Person( id: 2, username: "t2", password: "123", fullName: "Jake", age: 16);
people.add(person);

person = new Person( id: 3, username: "t3", password: "123", fullName: "Noah", age: 21);
people.add(person);

Collections.sort(people, (obj1, obj2) -> obj1.getFullName().compareTo(obj2.getFullName()));

System.out.println(people);
```

### 3. Sort Object By Property

#### Kết Quả

```
[  
  Person{id=2, username='t2', password='123', fullName='Jake', age=16},  
  Person{id=3, username='t3', password='123', fullName='Noah', age=21},  
  Person{id=1, username='t1', password='123', fullName='Oliver', age=20}  
]
```

### 3. Sort Object By Property

#### Decending

`obj1.getFullName().compareTo(obj2.getFullName()) ->`

`obj2.getFullName().compareTo(obj1.getFullName())`

### 3. Sort Object By Property

Vd: Hãy sắp xếp theo age tăng, giảm

### 3. Sort Object By Property

Có Thể Dùng Cách 2, Decending đổi obj1 với obj2

```
Collections.sort(people, new Comparator<Person>() {  
    @Override  
    public int compare(Person obj1, Person obj2) {  
        return obj1.getFullName().compareTo(obj2.getFullName());  
    }  
});
```