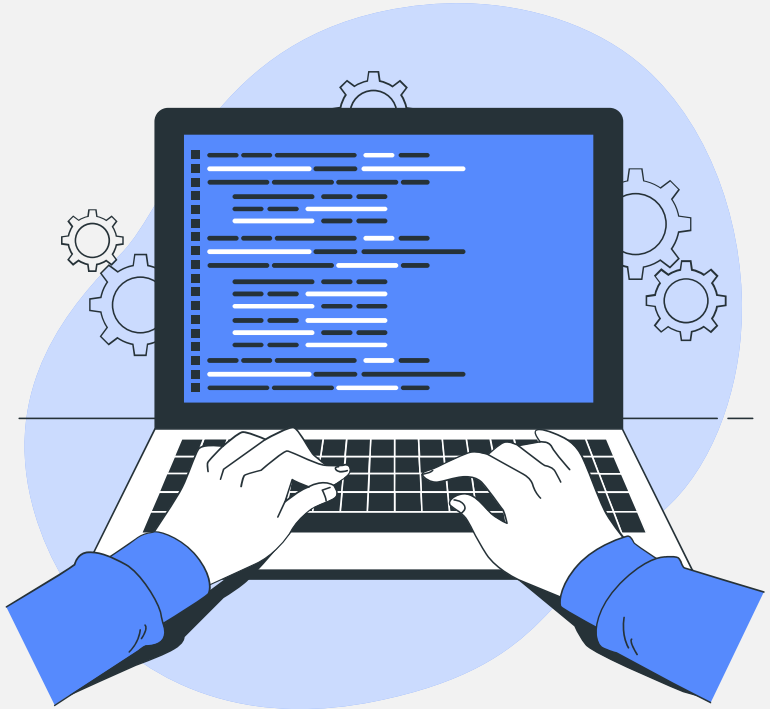
# STRING

# Mục lục:





**String** là lớp để xử lý xâu kí tự trong ngôn ngữ lập trình Java. Các bạn có thể nghĩ **String** như một mảng kí tự nhưng có thể mở rộng, thu hẹp và hỗ trợ rất nhiều hàm xử lý xâu thông dụng.

**Khai báo String:**

```
string name_string;
```

EXAMPLE

```
public class Main {
    public static void main(String[] args) {
        String s = "28tech java";
        System.out.println(s);
    }
}
```

**OUTPUT**

```
28tech java
```

# 1. Nhập xuất string:

Để nhập xâu kí tự ta sử dụng **hàm nextLine(),** hàm này sẽ đọc tới khi gặp kí tự xuống dòng.

EXAMPLE

```
public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner("System.in");
        String s = sc.nextLine();
        System.out.println(s);
    }
}
```

# 1. Nhập xuất string:

Khi dùng nextLine(), bản chất cách hoạt động của nextLine() sẽ dừng nhập tới khi gặp dấu xuống dòng, vì thế hãy đảm bảo trước khi nhập nextLine(), trong bộ đệm bàn phím không còn thừa dấu enter do các hàm như nextInt(), nextLong()... để lại từ câu lệnh nhập trước

**Tình huống xảy ra trôi lệnh:**

EXAMPLE

```
public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner("System.in");
        int n = sc.nextInt();
        String s = sc.nextLine();
        System.out.println(s);
    }
}
```

**OUTPUT**

Rỗng

# 1. Nhập xuất string:

**Cách xử lý:** Hãy nhớ rằng không phải cứ trước nextLine() là bạn cần xóa bộ đệm, bao giờ trước nextLine() mà có câu lệnh như nextInt(), nextLong().. thì mới cần phải xóa bộ đệm. Các bạn xóa đi phím enter trong bộ đệm bằng câu lệnh nextLine().

**Xử lí trôi lệnh:**


```
public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner("System.in");
        int n = sc.nextInt();
        sc.nextLine(); // đọc phím enter thừa
        String s = sc.nextLine();
        System.out.println(s);
    }
}
```

# 2. Các hàm thông dụng:

## Chú ý:

String trong Java một khi đã khai báo bạn không thể thay đổi nó, vì thế các hàm của String đều trả về 1 xâu kí tự mới sau khi áp dụng các hàm.

# 2. Các hàm thông dụng:

## Hàm length():

Trả về số kí tự trong xâu

EXAMPLE

```
public class Main {
    public static void main(String[] args) {
        String s = "28tech java";
        System.out.println(s.length());
    }
}
```

**OUTPUT**

11

# 2. Các hàm thông dụng:

## Hàm charAt():

String tương tự như mảng, chỉ số bắt đầu từ 0,
hàm charAt(index) trả về kí tự ở chỉ số index

EXAMPLE

```
public class Main {
    public static void main(String[] args) {
        String s = "28tech java";
        for(int i = 0; i < s.length(); i++){
            System.out.print(s.charAt(i) + " ");
        }
    }
}
```

**OUTPUT**

```
2 8 t e c h   j a v a
```

# 2. Các hàm thông dụng:

## Hàm toUpperCase():

Trả về xâu ở dạng in hoa, hàm này không thay đổi xâu ban đầu

EXAMPLE

```
public class Main {
    public static void main(String[] args) {
        String s = "28tech java";
        String t = s.toUpperCase();
        System.out.println(s);
        System.out.println(t);
    }
}
```

**OUTPUT**

```
28tech java
28TECH JAVA
```

# 2. Các hàm thông dụng:

## Hàm toLowerCase():

Trả về xâu ở dạng in thường, hàm này không thay đổi xâu ban đầu

EXAMPLE

```
public class Main {
    public static void main(String[] args) {
        String s = "28TECH JAVA";
        String t = s.toLowerCase();
        System.out.println(s);
        System.out.println(t);
    }
}
```

**OUTPUT**

```
28TECH JAVA
28tech java
```

# 2. Các hàm thông dụng:

## Hàm concat():

**Hàm concat()** nối xâu kí tự khác vào cuối xâu hiện tại, bạn có thể sử dụng toán tử + để làm chức năng tương tự.

EXAMPLE

```
public class Main {
    public static void main(String[] args) {
        String s = "java ";
        String t = "28tech.com.vn";
        String st = s.concat(t);
        String ts = t + s;
        System.out.println(st);
        System.out.println(ts);
    }
}
```

**OUTPUT**

```
java 28tech.com.vn
28tech.com.vnjava
```

# 2. Các hàm thông dụng:

## Hàm compareTo() và compareToIgnoreCase():

**Hàm compareTo()** so sánh 2 xâu theo thứ tự từ điển, nếu 2 xâu bằng nhau trả về 0, trả về số âm nếu xâu hiện tại nhỏ hơn xâu cần so sánh, ngược lại trả về số dương.

**Hàm compareToIgnoreCase()** sẽ bỏ qua in hoa in thường khi so sánh.

EXAMPLE

```
public class Main {
    public static void main(String[] args) {
        String s = "java 28tech";
        String t = "java 28TECH";
        System.out.println(s.compareTo(t));
        System.out.println(s.compareToIgnoreCase(t));
    }
}
```

OUTPUT

```
32
0
```

# 2. Các hàm thông dụng:

## Hàm substring():

**Hàm substring()** trả về xâu con

EXAMPLE

```
public class Main {
    public static void main(String[] args) {
        String s = "java 28tech";
        System.out.println(s.substring(5));
        System.out.println(s.subSequence(5, 7));
    }
}
```

OUTPUT

```
28tech
28
```

# 2. Các hàm thông dụng:

## Hàm contains():

Kiểm tra sự tồn tại của xâu con

EXAMPLE

```
public class Main {
    public static void main(String[] args) {
        String s = "java 28tech";
        System.out.println(s.contains("28tech"));
    }
}
```

OUTPUT

true

# 3. Tách các từ trong xâu:

## Cách 1: Sử dụng hàm split và regex

Để tách các từ trong xâu theo dấu cách ta sử dụng hàm split, hàm này trả về 1 mảng chứa các từ trong xâu.

EXAMPLE

```
public class Main {
    public static void main(String[] args) {
        String s = "java    28tech   C++   28tech.com.vn";
        String[] arr = s.split("\\s+");
        for(String x : arr){
            System.out.println(x);
        }
    }
}
```

**OUTPUT**

```
java
28tech
C++
28tech.com.vn
```

# 3. Tách các từ trong xâu:

## Cách 2: Sử dụng lớp StringTokenizer


```
public class Main {
    public static void main(String[] args) {
        String s = "java    28tech   C++   28tech.com.vn";
        StringTokenizer st = new StringTokenizer(s);
        while(st.hasMoreTokens()){
            System.out.println(st.nextToken());
        }
    }
}
```

**OUTPUT**

```
java
28tech
C++
28tech.com.vn
```

# 4. Nhắc lại kiểm tra loại kí tự:

Để kiểm tra loại kí tự hay chuyển đổi in hoa in thường các bạn sử dụng **lớp Character.**

| Hàm | Chức năng |
|---|---|
| isDigit(char c) | Kiểm tra chữ số |
| isLowerCase (char c) | Kiểm tra chữ in thường |
| isUpperCase(char c) | Kiểm tra in hoa |
| isAlphabetic(char c) | Kiểm tra chữ cái |
| char toLower(char c) | Chuyển thành chữ in thường |
| char toUpper(char c) | Chuyển thành chữ in hoa |

# 4. Nhắc lại kiểm tra loại kí tự:

## Hàm chuyển xâu thành in thường:

EXAMPLE

```
public class Main {
    public static void main(String[] args) {
        String s = "java    28tech   C++   28tech.com.vn";
        int digit = 0, lower = 0, upper = 0, special = 0;
        for(char x : s.toCharArray()){
            if(Character.isDigit(x))
                ++digit;
            else if(Character.isLowerCase(x))
                ++lower;
            else if(Character.isUpperCase(x))
                ++upper;
            else ++special;
        System.out.println(digit + " " + lower + " " + upper + " " + special);
    }
}
```

**OUTPUT**

```
4 17 1 14
```

# 5. Chuyển xâu thành số và ngược lại:

Trong Java để chuyển số thành xâu kí tự bạn chỉ cần cộng với 1 xâu rỗng, còn chuyển từ xâu thành số có thể sử dụng các hàm parse của lớp Integer, Long...


```
public class Main {
    public static void main(String[] args) {
        int n = 2804;
        String s = "" + n;
        System.out.println(s);
        System.out.println(Integer.parseInt(s));
    }
}
```

**OUTPUT**

```
2804
2804
```

# 6. Bài toán số lớn:

Khi gặp bài toán mà số lượng chữ số của số đầu bài cho lên tới hàng nghìn chữ số thì các bạn không thể dùng int hay long để lưu. Trong trường hợp này các bạn lưu như một xâu kí tự

## Tính tổng chữ số của 1 số có hàng trăm nghìn chữ số:

EXAMPLE

```
public class Main {
    public static void main(String[] args) {
        String s = "18238128381283812381123123123123123823812381283";
        int sum = 0;
        for(char x : s.toCharArray()){
            sum += x - '0';
        }
        System.out.println(sum);
    }
}
```

**OUTPUT**

160

# 7. Lớp BigInteger:

**Trong Java bạn có thể xử lý số nguyên lớn bằng lớp BigInteger:**

EXAMPLE

```
public class Main {
    public static void main(String[] args) {
        String s1 = "18238128381283812381123123123123";
        String s2 = "192399239192393";
        BigInteger num1 = new BigInteger(s1);
        BigInteger num2 = new BigInteger(s2);
        System.out.println(num1.add(num2));
        System.out.println(num1.subtract(num2));
        System.out.println(num1.multiply(num2));
        System.out.println(num1.divide(num2));
    }
}
```

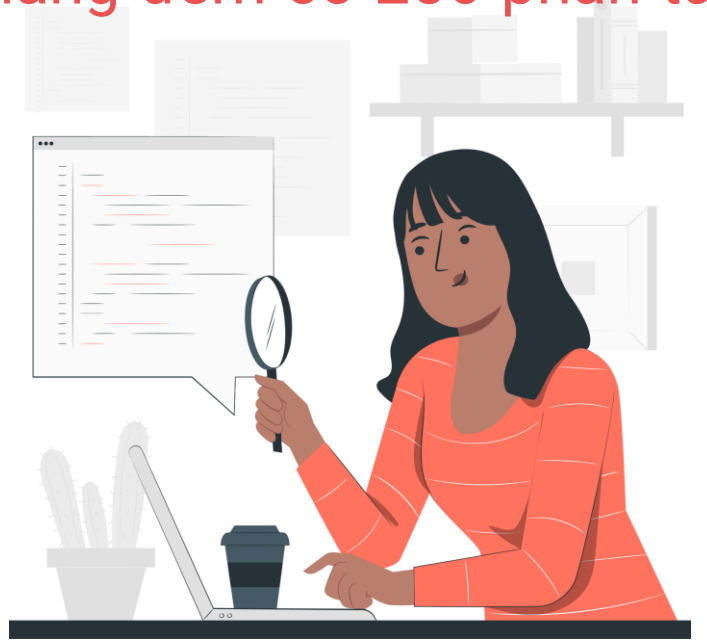
**OUTPUT**

```
18238128381283812573522362315516
18238128381283812188723883930730
3509002024852195578807203369633589530424003339
94793141895152066
```

# 8. Các bài toán liên quan tới tần suất xuất hiện của kí tự trong xâu:

**Bài toán:** Đếm số lần xuất hiện của các kí tự trong xâu sau đó liệt kê theo thứ tự từ điển tăng dần

Để đếm tần suất xuất hiện của các kí tự xuất hiện trong xâu các bạn có thể sử dụng mảng để đếm, vì các kí tự thường gặp đều có mã ASCII từ 0 tới 255 nên sử dụng mảng đếm có 256 phần tử là có thể đếm được kí tự xuất hiện trong xâu.



# 8. Các bài toán liên quan tới tần suất xuất hiện của kí tự trong xâu:

## Cách 1: Sử dụng mảng

EXAMPLE

```
public class Main {
    public static void main(String[] args) {
        String s = "28tech28te";
        int[] cnt = new int[256];
        for(char x : s.toCharArray()){
            cnt[x]++;
        }
        for(int i = 0; i < 256; i++){
            if(cnt[i] != 0){
                System.out.println((char)i + " " + cnt[i]);
            }
        }
    }
}
```

**OUTPUT**

```
2 2
8 2
c 1
e 2
h 1
t 2
```

# 8. Các bài toán liên quan tới tần suất xuất hiện của kí tự trong xâu:

## Cách 2: Sử dụng map

EXAMPLE

```
public class Main {
    public static void main(String[] args) {
        String s = "28tech28te";
        TreeMap<Character, Integer> map = new TreeMap<>();
        for(char x : s.toCharArray()){
            if(map.containsKey(x)){
                map.put(x, map.get(x) + 1);
            }
            else map.put(x, 1);
        }
        Set<Map.Entry<Character, Integer>> entrySet = map.entrySet();
        for(Map.Entry<Character, Integer> entry : entrySet){
            System.out.println(entry.getKey() + " " + entry.getValue());
        }
    }
}
```

**OUTPUT**

```
2 2
8 2
c 1
e 2
h 1
t 2
```

# 9. Lớp StringBuilder:

String trong Java không thể thay đổi một khi nó đã được khai báo, để sử dụng các hàm như chèn, xóa, thêm kí tự ta có thể sử dụng lớp StringBuilder.

**Ví dụ: Chuẩn hóa tên:**


```
public class Main {
    public static void main(String[] args) {
        String s = "NgUYEN   ThUY  liNh";
        StringBuilder sb = new StringBuilder("");
        String[] arr = s.split("\\s+");
        for(String x : arr){
            sb.append(Character.toUpperCase(x.charAt(0)));
            for(int j = 1; j < x.length(); j++){
                sb.append(Character.toLowerCase(x.charAt(j)));
            }
            sb.append(" ");
        }
        sb.deleteCharAt(sb.length() - 1); // xóa dấu cách thừa
        System.out.println(sb.toString());
    }
}
```

**OUTPUT**

Nguyen Thuy Linh