## GIỚI THIỆU TỔNG QUAN VỀ JAVA

## CÁC LỆNH ĐIỀU KHIỂN CHƯƠNG TRÌNH TRONG JAVA

### Các lệnh rẽ nhánh trong java

Như các ngôn ngữ lập trình khác, java cung cấp các lệnh rẽ nhánh để có thể thực hiện một trong nhiều phương án khác nhau khi lập trình. Có 3 loại cấu trúc rẽ nhánh:

* Mệnh đề if
* Mệnh đề switch … case
* Toán tử điều kiện

#### Cấu trúc rẽ nhánh if

* **Phân loại**

Mệnh đề if trong java được chai thành các loại sau đây:

|  |  |  |
| --- | --- | --- |
| **STT** | **Phân loại** | **Mô tả** |
| 1 | Cấu trúc rẽ nhánh if | Cấu trúc if |
| Cấu trúc if...else |
| Cấu trúc if-else nhiều nhánh |

##### Cấu trúc if

* **Khái niệm**

Lệnh if hay còn gọi là lệnh if thiếu, dùng để thực hiện một hành động mà chỉ có một trường hợp duy nhất.

* **Cú pháp**

|  |
| --- |
| if (<Biểu thức điều kiện>) {  <Các câu lệnh>  **}** |

Trong đó:

* Từ khóa if phải viết bằng chữ thường
* Kết quả của biểu thức điều kiện phải có tính **Đúng** hoặc **Sai.**
* Biểu thức luận lý phải đặt trong cặp dấu ( ).
* Các câu lệnh từ 2 lệnh trở lên thì phải đặt trong dấu { }.
* Không đặt dấu chấm phẩy sau câu lệnh if.
* **Lưu đồ**

A diagram of a diagram

Description automatically generated

**Hình 1.1**: Lưu đồ hoạt động if thiếu

* **Hoạt động**

Nếu biểu thức điều kiện có giá trị là True (Đúng) thì thực hiện các câu lệnh. Ngược lại biểu thức điều kiện có giá trị là False (Sai) thì các câu lệnh sẽ bị bỏ qua và thực hiện tiếp các lệnh còn lại của chương trình.

|  |
| --- |
| **VD 1:** Viết chương trình in ra thông báo “Ban da du tuoi vi thanh vien.” nếu Tuổi >= 18 |
| public class **IfThieu** {      public static void **main**(**String**[] args){          int Tuoi = 20;          if(Tuoi >= 18) {  **System**.out.**println**("Ban da du tuoi vi thanh vien.");          }      }  } |
| **Hiển thị:** |
| Ban da du tuoi vi thanh vien. |

##### Cấu trúc if else

* **Khái niệm**

Lệnh if else hay còn gọi là lệnh if đủ, dùng để thực hiện một hành động mà có hai trường hợp.

* **Cú pháp**

|  |
| --- |
| if (<Biểu thức điều kiện>)  <Các câu lệnh của if>  else  <Các câu lệnh của else> |

Trong đó:

* Từ khóa if else phải viết bằng chữ thường
* Kết quả của biểu thức điều kiện phải có tính **Đúng** hoặc **Sai.**
* Biểu thức luận lý phải đặt trong cặp dấu ( ).
* Các câu lệnh từ 2 lệnh trở lên thì phải đặt trong dấu { }.
* Không đặt dấu chấm phẩy sau câu lệnh if.
* **Lưu đồ**

A diagram of a diagram

Description automatically generated

**Hình 1.2**: Lưu đồ hoạt động if đủ

* **Hoạt động**

Nếu biểu thức điều kiện mang giá trị True (Đúng) thì thực hiện các câu lệnh của if, ngược lại nếu biểu thức điều kiện mang giá trị False (sai) thì thực hiện các câu lệnh của else.

|  |
| --- |
| **VD 2:** Viết chương trình in ra thông báo. Nếu Tuổi >= 18 “Ban da du tuoi vi thanh vien.” Ngược lại, in ra thông báo “Ban chua du tuoi vi thanh vien. ” |
| public class **IfDu** {      public static void **main**(**String**[] args){          int Tuoi = 10;          if(Tuoi >= 18) {  **System**.out.**println**("Ban da du tuoi vi thanh vien");          }          else{  **System**.out.**println**("Ban chua du tuoi vi thanh vien");          }      }  } |
| **Hiển thị:** |
| Ban chua du tuoi vi thanh vien. |

##### Cấu trúc if-else nhiều nhánh

* **Khái niệm**

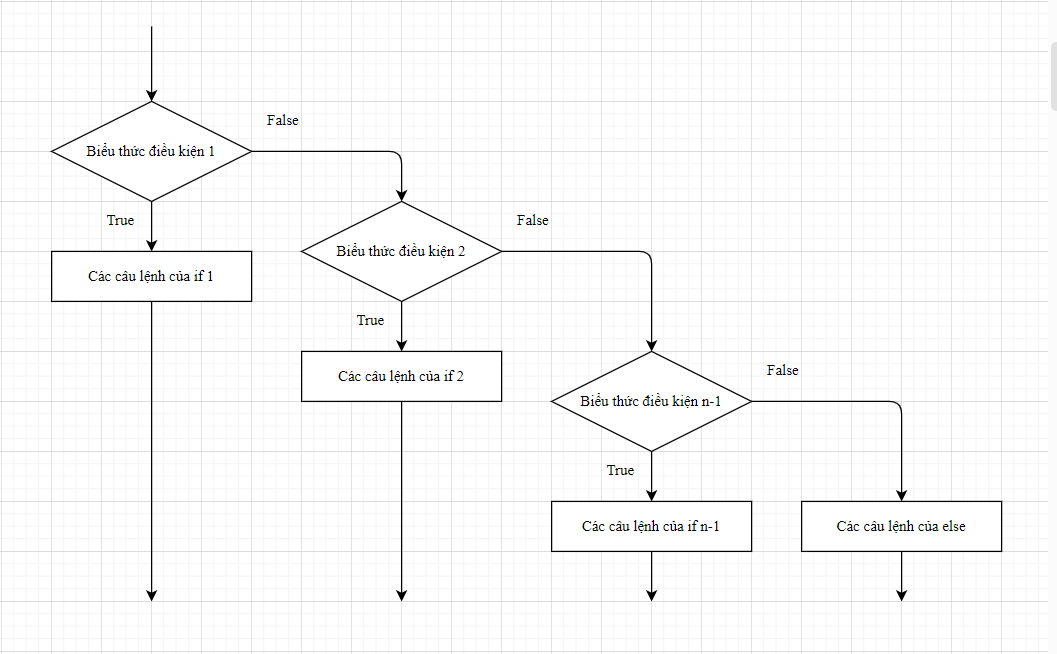
Lệnh if-else nhiều nhánh hay còn gọi là lệnh else if, dùng để thực hiện một hành động có từ ba trường hợp trở lên.

* **Cú pháp:** Giả sử có n trường hợp (n>=3)

|  |
| --- |
| if (<Biểu thức điều kiện 1>)  <Các câu lệnh của if 1>  else if (<Biểu thức điều kiện 2>)  <Các câu lệnh của if 2>  ...  else if (<Biểu thức điều kiện n-1>)  <Các câu lệnh của if n-1>  else  <Các câu lệnh của n> |

Trong đó:

* Từ khóa if, else if, else phải viết bằng chữ thường
* Kết quả của biểu thức điều kiện phải có tính **Đúng** hoặc **Sai.**
* Biểu thức luận lý phải đặt trong cặp dấu ( ).
* Các câu lệnh từ 2 lệnh trở lên thì phải đặt trong dấu { }.
* Không đặt dấu chấm phẩy sau câu lệnh if.
* **Lưu đồ**



**Hình 1.3**: Lưu đồ hoạt động if-else nhiều nhánh

* **Hoạt động**

Nếu biểu thức điều kiện mang giá trị True (Đúng) thì thực hiện các câu lệnh của if, ngược lại nếu biểu thức điều kiện mang giá trị False (sai) thì thực hiện các câu lệnh của else.

|  |
| --- |
| **VD 3:** Viết chương trình in ra xếp loại học tập của học sinh đó. |
| public class **IfElse** {      public static void **main**(**String**[] args){          int diem=9;            if((diem>10) || (diem<0)){  **System**.out.**println**("Diem nhap vao khong hop le!");          } else if(diem>=9){  **System**.out.**println**("Xuat sac!");          } else if(diem>=6.5){  **System**.out.**println**("Kha!");          } else if(diem>=5){  **System**.out.**println**("Trung binh!");          } else if(diem>=3.5){  **System**.out.**println**("Yeu!");          } else {  **System**.out.**println**("Kem!");          }      }  } |
| **Hiển thị:** |
| Xuat sac! |

#### Cấu trúc rẽ nhánh switch

* **Khái niệm**

Cấu trúc rẽ nhánh switch là một cấu trúc điều khiển và rẽ nhánh hoàn toàn có thể được thay thế bằng cấu trúc if-else nhiều nhánh. Tuy nhiên, nó cũng có mặt hạn chế là giá trị của biểu thức điều kiện phải là giá trị hằng nguyên (có giá trị cụ thể).

Việc sử dụng switch sẽ giúp code của chúng ta dễ viết, dễ đọc và có hiệu năng tốt hơn so với sử dụng if … else trong trường hợp có nhiều điều kiện có thể xảy ra.

Câu lệnh switch gồm 2 loại

* Lệnh switch case hay còn gọi là switch thiếu.
* Lệnh switch case default hay còn gọi là switch đủ.
* **Cú pháp**

|  |  |
| --- | --- |
| **Tên** | **Cú pháp** |
| switch thiếu | switch (Biểu thức điều kiện) {  case <Giá trị 1> :  <Khối lệnh 1>  break;  case <Giá trị 2> :  <Khối lệnh 2>  break;  case <Giá trị n> :  <Khối lệnh n>  break;  } |
| switch đủ | switch (Biểu thức điều kiện) {  case <Giá trị 1> :  <Khối lệnh 1>  break;  case <Giá trị 2> :  <Khối lệnh 2>  break;  case <Giá trị n-1> :  <Khối lệnh n-1>  break;  default :  <Khối lệnh n>  break;  } |

Trong đó:

* Từ khóa switch, case, default, break phải viết bằng chữ thường.
* Biểu thức bắt buộc là giá trị hằng, có thể là biểu thức nhưng kết quả cần là hằng số nguyên (char, int, long,…)
* Khối lệnh 1, 2… n có thể gồm nhiều câu lệnh, nhưng không cần đặt trong cặp dấu { }
* **Lưu đồ**

A diagram of a flowchart

Description automatically generated

**Hình 1.4**: Lưu đồ hoạt động swith case

A diagram of a flowchart

Description automatically generated

**Hình 1.5**: Lưu đồ hoạt động swith case default

* **Hoạt động**

Biểu thức điều kiện sẽ được so sánh lần lượt với giá trị của các case:

* Nếu có 1 case nào đó khớp giá trị thì khối lệnh tương ứng của case đó sẽ được thực hiện cho tới khi gặp lệnh break. Do đó, nếu chúng ta không sử dụng break thì tất cả các case kể từ case khớp giá trị đều được thực hiện.
* Đối với cấu trúc switch thiếu, nếu không có 1 case nào khớp giá trị biểu thức điều kiện thì sẽ không có một khối lệnh nào được thực hiện.
* Đối với cấu trúc switch đủ, nếu không có 1 case nào khớp giá trị biểu thức điều kiện thì khối lệnh của default sẽ được thực hiện.

|  |
| --- |
| **VD 4:** Viết chương trình nhập vào tháng và in ra quý. **(Sử dụng switch thiếu).** |
| **Giải thuật:** Nhập vào giá trị tháng, kiểm tra xem tháng có hợp lệ (trong khoảng 1 đến 12). Nếu hợp lệ in ra quý tương ứng (13: quý 1, 46: quý 2, 79: quý 3, 1012: quý 4). |
| import **java**.**util**.**Scanner**;  public class **Main** {      public static void **main**(**String**[] args) {  **Scanner** scanner = new **Scanner**(**System**.in);  **System**.out.**print**("Nhap vao thang: ");          int thang = scanner.**nextInt**();          if (thang > 0 && thang <= 12) {              switch (thang) {                  case 1:                  case 2:                  case 3:  **System**.out.**println**("Quy 1.");                      break;                  case 4:                  case 5:                  case 6:  **System**.out.**println**("Quy 2.");                      break;                  case 7:                  case 8:                  case 9:  **System**.out.**println**("Quy 3.");                      break;                  case 10:                  case 11:                  case 12:  **System**.out.**println**("Quy 4.");                          break;              }          }          else {  **System**.out.**println**("Thang khong hop le.");          }      }  } |
| **Hiển thị:** |
| Nhap vao thang: 5  Quy 2. |

|  |
| --- |
| **VD 5:** Viết chương trình nhập vào tháng và in ra quý. (**Sử dụng switch đủ).** |
| **Giải thuật:** Nhập vào giá trị tháng, in ra quý tương ứng (13: quý 1, 46: quý 2, 79: quý 3, 1012: quý 4). Nếu không có case nào khớp thì hiển thị ra ("Nhap Thang 1 -> 12."). |
| import **java**.**util**.**Scanner**;  public class **Main** {      public static void **main**(**String**[] args) {  **Scanner** scanner = new **Scanner**(**System**.in);  **System**.out.**print**("Nhap vao thang: ");          int thang = scanner.**nextInt**();          if (thang > 0 && thang <= 12) {              switch (thang) {                  case 1:                  case 2:                  case 3:  **System**.out.**println**("Quy 1.");                      break;                  case 4:                  case 5:                  case 6:  **System**.out.**println**("Quy 2.");                      break;                  case 7:                  case 8:                  case 9:  **System**.out.**println**("Quy 3.");                      break;                  default:  **System**.out.**println**("Quy 4.");                      break;              }          }          else {  **System**.out.**println**("Thang khong hop le.");          }      }  } |
| **Hiển thị:** |
| Nhap vao thang: 11  Quy 4. |

#### Toán tử điều kiện (? :) trong Java

Ngoài các lệnh kể trên, ngôn ngữ Java còn có một loại toán tử điều kiện giúp bạn kiểm tra nhanh các điều kiện và thực hiện phép gán giá trị cho một biến một cách rất nhanh chóng.

Toán tử này gồm ba toán hạng và được sử dụng để ước lượng các biểu thức quan hệ. Mục tiêu của toán tử là quyết định giá trị nào sẽ được gán cho biến. Toán tử này được viết như sau:

**Cú pháp:**

|  |
| --- |
| <Biểu thức 1> ? <Biểu thức 2> : <Biểu thức 3> |

**Hoạt động:**

Biểu thức 1 nhận giá trị đúng thì toán tử trả về giá trị là Biểu thức 2, ngược lại toán tử trả về giá trị là Biểu thức 3

|  |
| --- |
| **VD 6:** Toán tử điều kiện. |
| public class **Test** {     public static void **main**(**String** args[]){        int a , b;        a = 10;        b = (a == 1) ? 20: 30; *//Day la vi du ve toan tu dieu kien.*  **System**.out.**println**( "Gia tri cua b la : " +  b );        b = (a == 10) ? 20: 30; *//Day la vi du ve toan tu dieu kien.*  **System**.out.**println**( "Gia tri cua b la : " + b );     }  } |
| **Hiển thị:** |
| Gia tri cua b la : 30  Gia tri cua b la : 20 |

### Vòng lặp trong Java

Có thể có một tình huống khi chúng ta cần thực hiện một khối code một vài lần, điều này có thể được xem như một vòng lặp.

Java có 3 kỹ thuật lặp linh động. Bạn có thể sử dụng một trong ba vòng lặp sau:

* Vòng lặp while
* Vòng lặp do…while
* Vòng lặp for

#### Vòng lặp for

* **Khái niệm**

Vòng lặp for thường được sử dụng để giải quyết các hành động (khối code) có sự lặp đi lặp lại nhiều lần với số lần đã biết trước.

* **Cú pháp**

|  |
| --- |
| for (Khởi tạo giá trị biến lặp; Điều kiện lặp; Cập nhật biến lặp)  {  *// Khối lệnh lặp*  } |

Trong đó:

Vòng lặp for được định nghĩa bởi từ khóa for và được chia làm 3 phần chính, mỗi phần được ngăn cách bởi dấu chấm phẩy(;)

* **Phần khởi tạo biến lặp**: Kởi tạo biến vòng lặp có thể được khai báo và khởi tạo giá trị ngay bên trong phần khởi tạo của vòng lặp for. Phần khởi tạo biến được thực thi đầu tiên và chỉ thực thi 1 lần duy nhất trong vòng lặp for.
* **Điều kiện lặp:** Khối lệnh của vòng lặp for sẽ được thực hiện nếu **Biểu thức điều kiện lặp** cho giá trị đúng. Vòng lặp for kiểm tra biểu thức điều kiện trước khi thực hiện khối lệnh.
* **Cập nhật biến lặp:** Phần này sẽ được thực thi cuối mỗi lần lặp, sau khi khối lệnh của vòng lặp for được thực thi. Phần này thường chịu trách nhiệm thay đổi giá trị biến vòng lặp được sử dụng trong biểu thức điều kiện (nhằm tránh tình trạng lặp vô hạn). Sau khi thực thi xong phần cập nhật biến vòng lặp, chương trình quay trở lại đánh giá biểu thức điều kiện của vòng lặp **for** và cứ như thế đến khi biểu thức điều kiện lặp là sai.
* **Chú ý**
* Có dấu chấm phẩy sau lệnh for (i = 0; i <3; i++); → vòng lặp for sẽ không chạy đúng chương trình.
* Trong vòng lặp for có sử dụng từ 2 lệnh trở lên, nhớ sử dụng cặp ngoặc { } để bọc các lệnh đó lại.
* **Lưu đồ**

A diagram of a computer

Description automatically generated with medium confidence

**Hình 1.6**: Lưu đồ hoạt động vòng lặp for

**Bước 1**: Khởi tạo giá trị biến lặp (1 lần duy nhất)

**Bước 2**: Kiểm tra điều kiện lặp, nếu sai chuyển sang Bước 5, nếu đúng chuyển sang Bước 3.

**Bước 3**: Thực hiện nội dung trong thân vòng lặp

**Bước 4**: Cập nhật giá trị lặp và quay lại Bước 2

**Bước 5**: Kết thúc vòng lặp.

|  |
| --- |
| **VD 6:** Viết chương trình sử dụng vòng lặp for để in các giá trị từ 1 đến 5. |
| public class ForClass {      public static void main(String[] args) {          for (int i = 1; i <= 5; i++) {              System.out.println("Gia tri i: " + i);          }      }  } |
| **Hiển thị:** |
| Gia tri i: 1  Gia tri i: 2  Gia tri i: 3  Gia tri i: 4  Gia tri i: 5 |

#### Vòng lặp while

* **Khái niệm**

Được sử dụng để lặp đi lặp lại một hành động (khối lệnh) mà không biết trước số lần lặp.

* **Cú pháp**

|  |
| --- |
| while (Biểu thức điều kiện) {  // Khối lệnh lặp  } |

Trong đó:

* Từ khóa while phải viết bằng chữ thường
* Nếu khối lệnh bao gồm từ 2 lệnh trở lên thì phải đặt trong dấu { }
* **Lưu đồ**

A diagram of a diagram

Description automatically generated

**Hình 1.7**: Lưu đồ hoạt động vòng lặp while

* **Hoạt động**

Kiểm tra biểu thức điều kiện:

* Nếu biểu thức điều kiện là Đúng thì Thực hiện khối lệnh lặp, tiếp tục kiểm tra biểu thức điều kiện.
* Nếu biểu thức điều kiện là Sai thì kết thúc vòng lặp.

|  |
| --- |
| **VD 1:** Tính tổng các số từ 1 đến 5 bằng vòng lặp while. |
| public class Main {      public static void main(String[] args) {          int i = 1;          int sum = 0;          while (i <= 5) {              sum += i;              i++;          }          System.out.println("Tong tu 1 den 5 la: " + sum);      }  } |
| **Hiển thị:** |
| Tong tu 1 den 5 la: 15 |

#### Vòng lặp do while

* **Khái niệm**

Vòng lặp do…while là vòng lặp mà chương trình sẽ thực thi ít nhất một lần.

* **Cú pháp**

|  |
| --- |
| do {  // Khối lệnh lặp  }  while (Biểu thức điều kiện); |

Trong đó:

* Từ khóa do, while phải viết bằng chữ thường
* Nếu khối lệnh bao gồm từ 2 lệnh trở lên thì phải đặt trong dấu { }
* **Lưu đồ**

A diagram of a diagram

Description automatically generated

**Hình 1.8**: Lưu đồ hoạt động vòng lặp do while

* **Hoạt động**

**Bước 1**: Thực hiện khối lệnh lặp.

**Bước 2**: Kiểm tra biểu thức điều kiện:

* Nếu biểu thức là **Đúng** thì quay lại Bước 1.
* Nếu biểu thức là **Sai** thì chuyển sang Bước 3 (khối lệnh được thi hành 1 lần).

**Bước 3**: Kết thúc vòng lặp.

* **Chú ý**
* Biểu thức: có thể là một biểu thức hoặc nhiều biểu thức con. Nếu là nhiều biểu thức con thì cách nhau bởi dấu phẩy (,) và tính đúng sai của biểu thức được quyết định bởi biểu thức con cuối cùng.
* Trong thân do…while (khối lệnh) có thể chứa một hoặc nhiều cấu trúc điều khiển khác.
* Trong thân do…while có thể sử dụng lệnh **continue** để chuyển đến đầu vòng lặp (bỏ qua các câu lệnh còn lại trong thân).
* Muốn thoát khỏi vòng lặp do…while tùy ý có thể dùng các lệnh **break, goto, return.**

|  |
| --- |
| **VD 1:** Viết chương trình tính tổng các số từ 1 đến 5 bằng chu trình do-while |
| public class Main {      public static void main(String[] args) {          int i = 1;          int sum = 0;          do {              sum += i;              i++;          } while (i <= 5);          System.out.println("Tong tu 1 den 5 la: " + sum);      }  } |
| **Hiển thị:** |
| Tong tu 1 den 5 la: 15 |

### Các lệnh điều khiển vòng lặp trong Java

#### Câu lệnh break

Câu lệnh break được sử dụng trong các khối lệnh gán nhãn, trong các chu trình lặp (for, while, do-while) và câu lệnh switch để chuyển điều khiển thực hiện chương trình ra khỏi khối lệnh trong cùng chứa nó.

Được sử dụng cho những bài toán không biết trước khi nào thì dừng, hoặc biểu thức điều kiện quá phức tạp, phải biểu diễn trong thân vòng lặp.

* Trong trường hợp khối lệnh được gán nhãn, khi gặp lệnh break thì phần còn lại của khối lệnh sẽ bị bỏ qua và chương trình vẫn được tiếp tục thực hiện các lệnh dứng sau đó.
* Đối với các chu trình lặp, khi gặp lệnh break thì phần còn lại của thân chu trình lặp cũng được bỏ qua và kết thúc chu trình GẦN NÓ NHẤT.
* Đối với lệnh switch, khi lặp lệnh break thì phần còn của câu lệnh switch bị bỏ qua và thực hiện tiếp tục những lệnh sau nó.

|  |
| --- |
| **VD 1:** Viết chương trình tính tổng các số từ 0 đến 5 có sử dụng break. |
| public class Main {      public static void main(String[] args) {          int s = 0;          int i = 0;          while (true) {              if (i >= 5) {                  break;              }              s += i;              i++;          }          System.out.println("Tong cac so tu 0 den 5 la: " + s);      }  } |
| **Hiển thị:** |
| Tong cac so tu 0 den 5 la: 10 |

#### Câu lệnh continue

Câu lệnh continue được sử dụng trong các cấu trúc lặp for, while, do-while để dừng việc thực thi của lần lặp hiện tại và bắt đầu lần lặp tiếp theo nếu điều kiện vẫn còn được thỏa mãn.  
Đối với chu trình lặp while, do-while, khi gặp continue thì phần còn lại của thân chu trình sẽ bị bỏ qua và tiếp tục kiểu tra điều kiện lặp để thực hiện quá trình lặp tiếp theo.  
Đối với chu trình lặp for, khi gặp continue thì phần còn lại của thân chu trình cũng sẽ bị bỏ qua và tiếp tục thực hiện biểu thức tăng/giảm biến điều kiện sau đó thân chu trình được lặp lại nếu điều kiện đúng.

Được sử dụng khi giá trị trong thân vòng lặp không thích hợp để thực hiện tiếp công việc, cần bỏ qua để sang một lần lặp tiếp theo.

|  |
| --- |
| **VD 1:** Viết chương trình tính tổng các số chẵn từ 0 đến 10 có sử dụng continue. |
| public class Main {      public static void main(String[] args) {          int s = 0;          for (int i = 0; i < 10; i++) {              if (i % 2 != 0) {                  continue;              }              s += i;          }          System.out.println("Tong cac so chan tu 0 den 10 la: " + s);      }  } |
| **Hiển thị:** |
| Tong cac so chan tu 0 den 10 la: 20 |

## BÀI TẬP ỨNG DỤNG