|  |  |  |
| --- | --- | --- |
| **No.** | **throw** | **throws** |
| 1) | Từ khóa throw trong java được sử dụng để ném ra một ngoại lệ rõ ràng. | Từ khóa throws trong java được  sử dụng để khai báo một ngoại lệ. |
| 2) | Ngoại lệ checked không được truyền ra nếu chỉ sử dụng từ khóa throw. | Ngoại lệ checked được truyền ra  ngay cả khi chỉ sử dụng từ khóa throws. |
| 3) | Sau throw là một instance. | Sau throws là một hoặc nhiều class. |
| 4) | Throw được sử dụng trong phương thức. | Throws được khai báo ngay sau dấu đóng ngoặc đơn của phương thức. |
| 5) | Bạn không thể throw nhiều exceptions. | Bạn có thể khai báo nhiều exceptions, Ví dụ: public void method()throws IOException,SQLException. |

HashTable là gì?

Design pattern trong java lalf

Serialization trong Java là gì? https://huongdanjava.com/vi/noi-ve-serialization-trong-java.html

* Serialization là một khái niệm giúp chúng ta có thể chuyển đổi trạng thái của một Java object thành một định dạng nào đó để Java object này có thể được lưu trữ ở đâu đó và sau đó, nó sẽ được sử dụng bởi một tiến trình khác.
* Thông thường, khi sử dụng Serialization, Java object của chúng ta sẽ được chuyển đổi qua byte streams và chúng ta có thể lưu byte stream này trong bộ nhớ, trên ổ đĩa, truyền qua mạng đến một server nào đó hoặc cũng có thể lưu chúng vào database.

https://o7planning.org/vi/10393/huong-dan-lap-trinh-java-socket

Tại sao lại chọn JAVA? java có điểm gì hay

Mô hình 3 lớp, MVC, session cookie,

***Interviewer:*** đồ án ngành của em làm về Thread, vậy em hãy cho anh biết có mấy cách tạo thread  
***Mình:*** có hai cách anh ạ, là: kế thừa từ class Thread và implement từ interface Runable  
***Interviewer:*** thế 2 cách đó có khác nhau gì không  
***Mình:*** em nghĩ kế thừa từ class Thread thì sẽ có nhìu chức năng hơn là implement từ Runable Interface anh ạ   
***Interviewer:*** em có thể cho anh biết vấn đề nào liên quan khi nói đến thread  
***Mình:*** dạ là synchronize anh ạ  
***Interviewer:*** vậy synchronize là gì  
***Mình:*** là cơ chế trong 1 thời điểm chỉ cho 1 thread được truy xuất tài nguyên thôi anh ạ  
***Interviewer:*** **vậy chỗ nào được đặt synchronize   
*Mình:* có thể synchronize 1 object, 1 method hoặc 1 block anh ạ**

Kiểu nguyên thủy: int, byte,.. là từ C, có các lớp bao tương ứng Integer, Byte,..

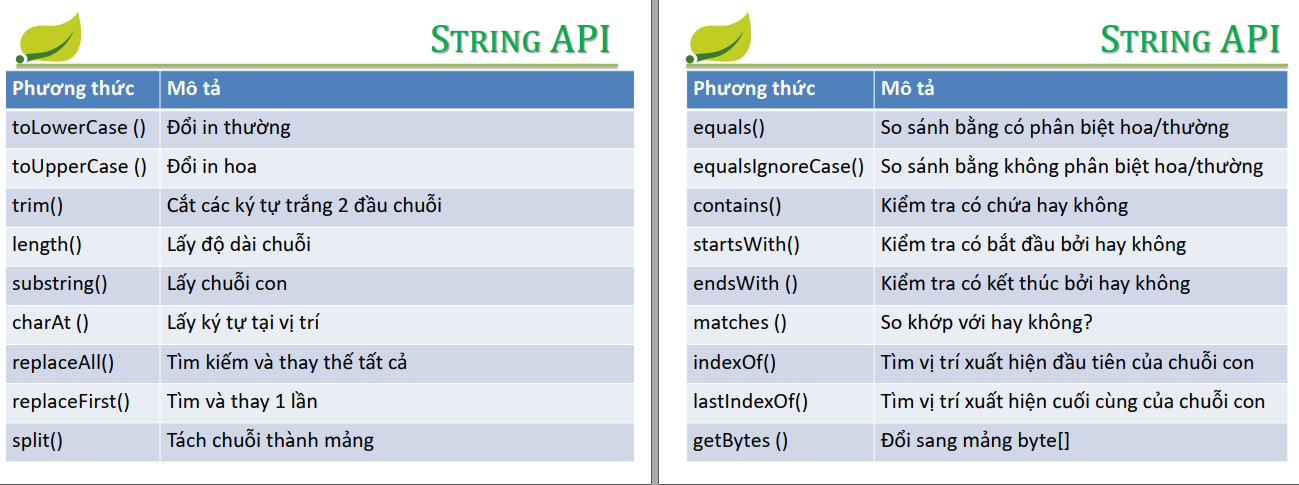
2 cách ép kiểu:

1. Làm tròn số từ nhỏ hơn -> lớn hơn: ko bị mất dl float f = i; //i = integer; -> ép kiểu tự động
2. Thu hẹp: ngược lại: có thể bị mất dl int i = (int)f; //f= float -> ép kiểu tường minh

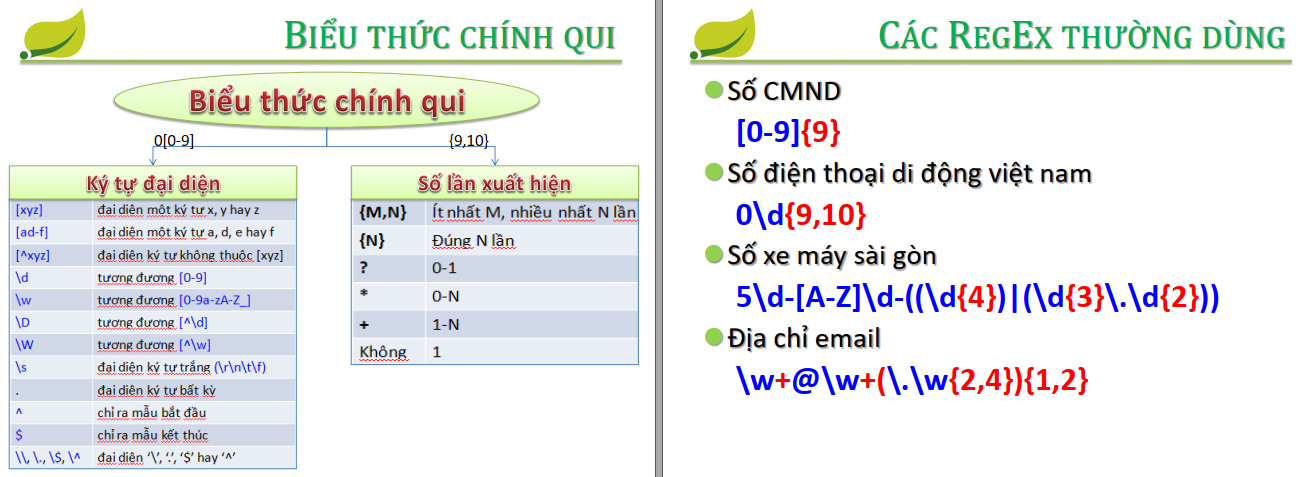
Chuyển đổi

Nguyên thủy -> wrap: wrap a = wrap.**valueOf**(nt)

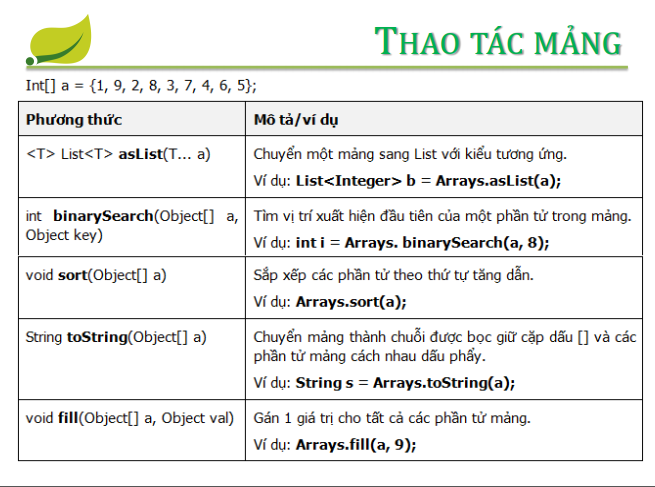
String -> nguyên thủy: nt = wrap.**parse**Wrap(chuỗi)

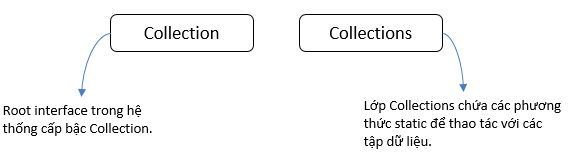


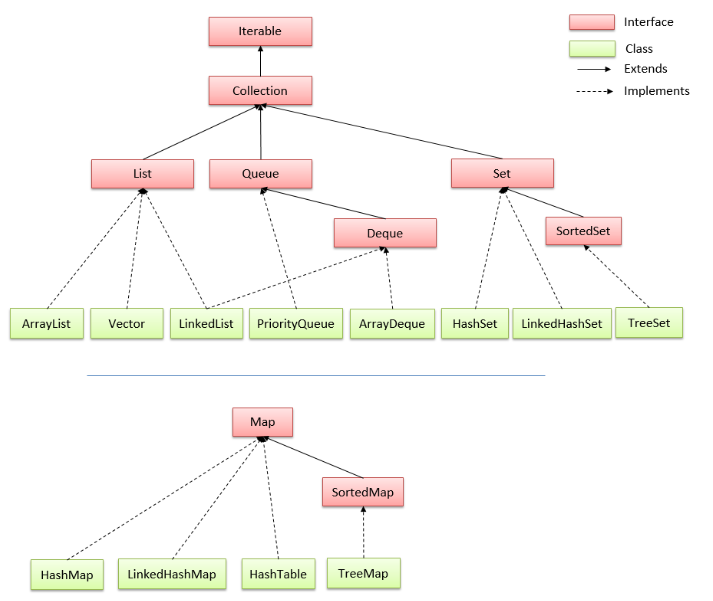
Thay đổi chuối: StringBuffer x= new StringBuffer();



Dùng hàm matches(bt) để so khớp







Các điểm quan trọng về lớp HashSet trong java là:

* HashSet chỉ chứa các phần tử duy nhất.
* HashSet lưu trữ các phần tử bằng cách sử dụng một cơ chế được gọi là **băm**.

Các điểm quan trọng về lớp TreeSet trong java là:

* Chỉ chứa các phần tử duy nhất giống như HashSet.
* Thời gian truy xuất nhanh.
* Duy trì thứ tự tăng dần.

Những điểm quan trọng về lớp **HashMap trong java** là:

* HashMap lưu trữ dữ liệu dưới dạng cặp key và value.
* Nó chứa các key duy nhất.
* Nó có thể có 1 key là null và nhiều giá trị null.
* Nó duy trì các phần tử KHÔNG theo thứ tự.

Những điểm quan trọng về lớp **Hashtable trong java** là:

* Hashtable là một mảng của list. Mỗi list được biết đến như một xô chứa các phần tử. Ví trí của một xô được xác định bằng việc gọi phương thức hashcode(). Hashtable cũng lưu trữ dữ liệu dưới dạng cặp key và value.
* Nó chứa các key duy nhất.
* Nó KHÔNG thể có bất kỳ key hoặc giá trị nào là null.
* Nó được đồng bộ.

Enum: kiểu dữ liệu tập hợp các hằng số

|  |
| --- |
| enum Season { |

|  |
| --- |
| SPRING, SUMMER, FALL, WINTER; |

|  |
| --- |
| } |

|  |
| --- |
|  |

|  |
| --- |
| public static void main(String[] args) { |

|  |
| --- |
| Season season = Season.WINTER; |

|  |
| --- |
| System.out.println(season); |

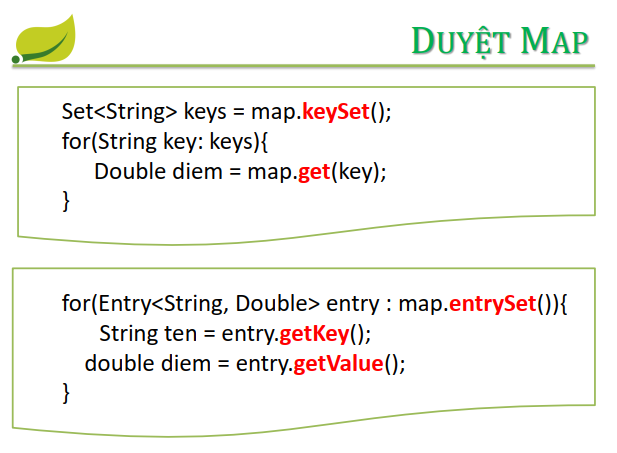
|  |
| --- |
| } |

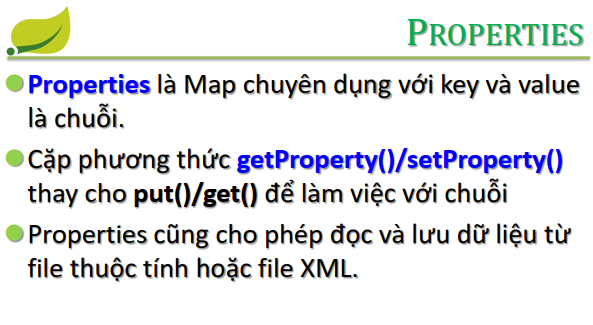
List: cho phép xuất hiện nhiều lần, Set thì ko

Cả 2 đều truy xuất = chỉ số

**Tại sao lại học JAVA? Java có đặc điểm gì hay**



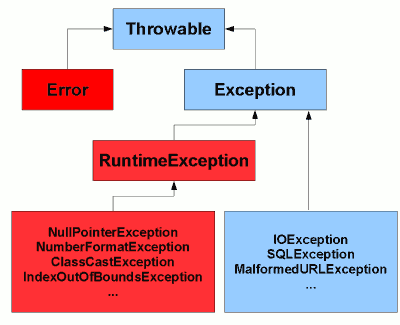




"So sánh Abstract class với Interface?" Những câu hỏi liên quan đến 2 thằng này mình thường xuyên gặp khi đi pv zzzz.  
  
"So sánh Array List với Linked List" câu này cũng tương tự  
  
Nhiều khi họ hỏi một số câu về Recursive(đệ quy), những thuật toán sort, search như quicksort, breadth first search, dept first search.... nếu bạn trả lời đc thì đó là một điểm cộnghttps://forums.voz.vn/images/smilies/Off/go.gif  
  
Ngoài ra thông thường là những câu về những tính chất của OOP, phân biệt các từ khóa private,static, final...  
  
Nâng cao thì có những câu về multi thread và garbage collection.  
  
Sau phần java core là những phần liên quan về công nghệ mà bạn đã từng có kinh nghiệm, cái này thì cũng zống như là project leader hỏi bạn, nhưng có phần detail về việc bạn implement như thế nào.https://forums.voz.vn/images/smilies/Off/shame.gif  
  
Ví dụ như "mô tả struts framework, spring framework" điểm mạnh điểm yếu, cấu trúc gồm những gì và thường họ đưa ra những tình huống cụ thể và iu cầu mình sử dụng framework đó để giải quyết vấn đề.https://forums.voz.vn/images/smilies/Off/adore.gif

OOP : Kiến thức cơ bản về lập trình hướng đối tượng (thừa kế, trừu tượng, đa hình,đóng gói)  
  
- Java Core : ví dụ Array List, Linked List là gì. Static, final là gì. Garbage collection, Multi-thread là gì.  
  
- Framework :ví dụ mô hình MVC, MVC2, 3-layer là gì. Những framework thông dụng như Spring,Struts,Hibernate bạn càng biết nhiều kiến thức về framework thì cơ hội càng cao. Advance hơn nữa thì bạn có thể tìm hiểu khái niệm về design pattern.  
  
- Database : bạn phải có kiến thức cơ bản về db như viết câu query và biết rõ tất cả những câu lệnh như Inner Join, Outer Join.  
  
Kinh nghiệm làm việc : Bạn có thể dễ dàng có được nó qua việc xin thực tập ở những cty outsource thông qua sự giới thiệu của trường hoặc là trên vietnamworks.

* 1. Tại sao ngôn ngữ Java lại là ngôn ngữ thuần hướng đối tượng , C hay pascal lại không phải
  2. 2. OOP bao gồm những tính chất nào, em hãy nêu 1 ví dụ về tính đa hình cho anh.  
     3. Interface là gì, tại sao người ta lại dùng / hay tạo ra Interface .  
     4. Tính kế thừa trong java là gì, ghi đè là gì , tại sao phải làm như vậy   
     SQL: em biết NVarchar khác kiểu Char ở điểm nào ko , khi nào dùng Nvarchar , khi nào dùng Char. Lệnh select, join  
     Hibernate : ánh xạ One to many, Many to one là gì,   
     JSP và Servlet : Em hiểu Servlet là gì, cách thông tin đi từ Servlet như thế nào, nó sẽ chui đi đâu ( đoạn này mình cũng ko hiểu là hỏi về cái gì https://forums.voz.vn/images/smilies/Off/pudency.gif) , đại khái là ví dụ như mình có 1 form có lệnh Sumbit , click vào nó ra trang khác thì tại sao lại được như vậy  
     Do get và Do post khác nhau điểm nào, luồng thông tin đi vào do get và do post ra sao.  
     Session với Cookie khác nhau điểm nào   
     HTTP Protocol là gì  
     Mô hình MVC, em biết Struts đúng ko, tại sao mô hình MVC lại có 3 lớp ,   
     giờ anh viết Core java vào phần View cũng được đúng ko, nhưng mà sao ko làm như vậy. Em hiểu cách tổ chức trong Strust 2 như thế nào,   
     Ví dụ khi cần xử lý 1 action đến JSP , tại sao Struts 2 có thể làm được như vậy ( cái này thì mình đúng ko hiểu, vì khi gọi 1 action gì đó, thì cách thức xử lý trong Struts nằm trong Method, rồi nó trả về Result nào thì, map nó vào Location = ".JSP" thôi )  
     Tóm lại cái đoạn này là ngoài phải hiểu luồng đi trong Struts 2 còn phải hiểu tại sao Struts 2 có thể làm được như vậy  
     String khác với String builder ở điểm nào
* thuật toán sắp xếp nào đó (sủi bọt, vun đống....); CSDL thì có thể là hỏi trigger là gì, index.... Ngôn ngữ sẽ hỏi cơ bản kiểu tính chất của ngôn ngữ hướng đối tượng là gì, lấy ví dụ.....



#### **2.1. Checked exception**

Các lớp được extends lớp Throwable ngoại trừ RuntimeException và Error được gọi là các checked exception. Ví dụ: IOException, SQLException, … Checked exception được kiểm tra tại thời gian compile.

#### **2.2. Unchecked exception**

Các lớp được extends lớp RuntimeException được gọi là unchecked exception. Ví dụ như ArithmeticException, NullPointerException, **IndexOutOfBoundsException**, … Unchecked exception không được kiểm tra tại thời gian biên dịch.

### 7. Tại sao java không support con trỏ?

Con trỏ là một biến tham chiếu tới địa chỉ ô nhớ. Nó không được sử dụng trong java vì nó không an toàn và phức tạp.

Bất cứ một ngôn ngữ lập trình nào nếu muốn được xem là một ngôn ngữ lập trình thuần hướng đối tượng nó phải thỏa mãn các điều kiện sau:

1. Nó phải hỗ trợ được 4 đặc điểm của phương pháp lập trình hướng đối tượng đó là:
   * Tính đóng gói dữ liệu (Encapsulation)
   * Tính kế thừa (Inheritance)
   * Tính trừu tượng (Abstraction)
   * Tính đa hình (Polymorphism)
2. Những kiểu dữ liệu cơ bản được định nghĩa sẵn như int, string, float, ... phải ở dạng **đối tượng**.
3. Những kiểu dữ liệu được người dùng định nghĩa cũng phải ở dạng **đối tượng**.
4. Những toán tử thực thi trên đối tượng phải ở dạng các **phương thức**.

Vậy ngôn ngữ lập trình Java vi phạm hai điều kiện (2) và (4).

Vi phạm điều kiện (2): Java hỗ trợ bạn khai báo các kiểu dữ liệu định nghĩa sẵn dưới dạng int, byte, long, ...v.v mà trong lập trình hướng đối tượng các kiểu dữ liệu này phải được đóng gói lại thành các đối tượng như Integer, String, ...v.v, mặc dù Java cũng hỗ trợ đóng gói các kiểu dữ liệu được định nghĩa sẵn này nhưng đó chỉ là lớp ngoài, thực chất nó vẫn dưới dạng kiểu dữ liệu cơ bản.

Vi phạm điều kiện (4): Java có thể sử dụng toán tử để thực hiện một phép tính nào đó giữa hai đối tượng như String s = "123" + "ABC", điều này vi phạm nguyên tắc thực thi trên đối tượng phải ở dạng các phương thức của phương pháp lập trình hướng đối tượng.