**Vecto trong java**

**1. Vector là gì?**

- Vecto là 1 lớp con của List Interface, cũng giống như ArrayList là có cấu trúc của List và thừa kế những phương thúc của List Interface, nhưng điểm khác của vecto so với ArrayLisst là nó được đồng bộ và nó chưa phương thức legacy mà không là một phần của Collection Framwork.

**2. Những constructor của vector:**

- Vecto cung cấp 4 constructor:

+ Vecto(): tạo một vecto rỗng có kích cỡ = 10;

+ Vecto(int size): tạo 1 vecto rỗng với kích cỡ ra truyền;

+ Vector(int size, int icnt): tạo vecto rỗng với kích cỡ và lượng gia(LƯỢNG GIA LÀ SỐ BỘ NHỚ SẼ CẤP PHÁT MỖI LẦN THIẾU, BÌNH THƯỜNG LÀ 1) mà ta truyền vào.

+ Vector (Collection C): tạo 1 vector từ bộ collection ta truyền.

**3. Một số phương thúc của Vector**

|  |  |
| --- | --- |
| **STT** | **Phương thức và Miêu tả** |
| 1 | **void add(int index, Object element)**  Chèn *element* đã xác định tại vị trí đã cho trong Vector này |
| 2 | **boolean add(Object o)**  Phụ thêm phần tử đã cho vào cuối của Vector này |
| 3 | **boolean addAll(Collection c)**  Phụ thêm tất cả phần tử trong Collection đã xác định tới cuối của Vector này, để mà chúng được trả về bởi Iterator của Collection đã cho đó |
| 4 | **boolean addAll(int index, Collection c)**  Chèn tất cả phần tử trong Collection đã xác định vào trong Vector này tại vị trí đã cho |
| 5 | **void addElement(Object obj)**  Thêm phần tử đã cho tới cuối của Vector này, tăng kích cỡ nó thêm 1 |
| 6 | **int capacity()**  Trả về dung lượng hiện tại của Vector này |
| 7 | **void clear()**  Gỡ bỏ tất cả phần tử từ Vector này |
| 8 | **Object clone()**  Trả về một mô phỏng của Vector này |
| 9 | **boolean contains(Object elem)**  Kiểm tra nếu đối tượng đã cho là một phần tử trong Vector này |
| 10 | **boolean containsAll(Collection c)**  Trả về true nếu Vector này chứa tất cả phần tử trong Collection đã cho |
| 11 | **void copyInto(Object[] anArray)**  Sao chép các thành phần của Vector này vào trong mảng đã cho |
| 12 | **Object elementAt(int index)**  Trả về phần tử tại index đã cho |
| 13 | **Enumeration elements()**  Trả về một bản liệt kê các phần tử của Vector này |
| 14 | **void ensureCapacity(int minCapacity)**  Tăng dung lượng của Vector này, nếu cần thiết, để bảo đảm rằng nó có thể giữ ít nhất số các phần tử được xác định bởi tham số minCapacity |
| 15 | **boolean equals(Object o)**  So sánh Object đã cho với Vector này về sự cân bằng |
| 16 | **Object firstElement()**  Trả về phần tử đầu tiên (tại chỉ mục 0) của Vector này |
| 17 | **Object get(int index)**  Trả về phần tử tại vị trí đã cho trong Vector này |
| 18 | **int hashCode()**  Trả về giá trị hash code cho Vector này |
| 19 | **int indexOf(Object elem)**  Tìm kiếm sự xuất hiện đầu tiên của tham số đã cho, kiểm tra tính cân bằng bởi sử dụng phương thức equals |
| 20 | **int indexOf(Object elem, int index)**  Tìm kiếm sự xuất hiện đầu tiên của tham số đã cho, bắt đầu tìm kiếm tại index, kiểm tra tính cân bằng bởi sử dụng phương thức equals |
| 21 | **void insertElementAt(Object obj, int index)**  Chèn đối tượng đã cho như là một phần tử vào Vector này tại index đã cho |
| 22 | **boolean isEmpty()**  Kiểm tra nếu Vector này không có phần tử |
| 23 | **Object lastElement()**  Trả về phần tử cuối cùng của Vector này |
| 24 | **int lastIndexOf(Object elem)**  Trả về chỉ mục của sự xuất hiện cuối cùng của đối tượng đã cho trong Vector này |
| 25 | **int lastIndexOf(Object elem, int index)**  Tìm kiếm ngược về sau cho đối tượng đã cho, bắt đầu từ index đã xác định, và trả về một chỉ mục |
| 26 | **Object remove(int index)**  Gỡ bỏ phần tử tại vị trí đã cho trong Vector này |
| 27 | **boolean remove(Object o)**  Gỡ bỏ sự xuất hiện đầu tiên của phần tử đã cho trong Vector này. Nếu Vector này không chứa phần tử đó, nó không bị thay đổi |
| 28 | **boolean removeAll(Collection c)**  Gỡ bỏ tất cả phần tử, mà chứa trong Collection đã cho, từ Vector này |
| 29 | **void removeAllElements()**  Gỡ bỏ tất cả phần tử từ Vector này và thiết lập kích cỡ về 0 |
| 30 | **boolean removeElement(Object obj)**  Gỡ bỏ sự xuất hiện đầu tiên (chỉ mục thấp nhất) của tham số từ Vector này |
| 31 | **void removeElementAt(int index)**  removeElementAt(int index) |
| 32 | **protected void removeRange(int fromIndex, int toIndex)**  Gỡ bỏ từ danh sách này tất cả phần tử mà có index từ fromIndex tới toIndex |
| 33 | **boolean retainAll(Collection c)**  Chỉ giữ lại phần tử, mà ở trong Collection đã cho, trong Vector này |
| 34 | **Object set(int index, Object element)**  Thay thế phần tử tại vị trí đã cho trong Vector này với phần tử đã xác định |
| 35 | **void setElementAt(Object obj, int index)**  Thiết lập phần tử tại index đã cho của Vector này thành đối tượng đã xác định |
| 36 | **void setSize(int newSize)**  Thiết lập kích cỡ của Vector này |
| 37 | **int size()**  Trả về số phần tử trong Vector này |
| 38 | **List subList(int fromIndex, int toIndex)**  Trả về một danh sách phụ từ fromIndex tới toIndex |
| 39 | **Object[] toArray()**  Trả về một mảng chứa tất cả phần tử trong Vector này theo đúng thứ tự |
| 40 | **Object[] toArray(Object[] a)**  Trả về một mảng chứa tất cả phần tử trong Vector này theo đúng thứ tự; kiểu runtime của mảng trả về là mảng đã xác định |
| 41 | **String toString()**  Trả về một biểu diễn chuỗi của Vector này, chứa biểu diễn chuỗi của mỗi phần tử |
| 42 | **void trimToSize()**  Trim dung lượng của Vector này về kích cỡ hiện tại của vector |