Contents

[**Chương 1: Collection trong java** 2](#_Toc117621324)

[**1.** **Giới thiệu** 2](#_Toc117621325)

[**2.** **Các giới hạn của việc sử dụng mảng (Array)** 2](#_Toc117621326)

[**2.1.** **Mảng rất cơ bản và quen thuộc.** 2](#_Toc117621327)

[**2.2.** **Mảng có kích cỡ và số chiều cố định.** 2](#_Toc117621328)

[**2.3.** **Các phần tử được đặt và tham chiếu một cách liên tiếp nhau trong bộ nhớ.** 2](#_Toc117621329)

[**2.4.** **Xóa phần tử ra khỏi mảng** 2](#_Toc117621330)

[**Chương 2: Bài tập** 4](#_Toc117621331)

# **Chương 1: Collection trong java**

1. **Giới thiệu**

Tập hợp chính là một ý tưởng cơ bản của chương trình và của ngôn ngữ lập trình. Một ứng dụng thường xuyên phải làm việc với tập hợp ví dụ như việc lưu trữ thông tin các nhân viên, tập hợp các ảnh ,…

Java cũng như các ngôn ngữ khác hỗ trợ mảng (array) như một tập hợp cơ bản nhất, xong việc làm việc với mảng là không thuận tiện trong nhiều trường hợp bởi vì trong thời gian sống của mảng việc tăng thêm phần tử hoặc xóa các phần tử của mảng rất khó khăn và phải trả giá đắt về hiệu năng chương trình nếu cố tình làm điều đó .

Hình minh họa một mảng:

Diagram

Description automatically generated

1. **Các giới hạn của việc sử dụng mảng (Array)**
   1. **Mảng rất cơ bản và quen thuộc.**

* Lưu trữ các kiểu tham chiếu (object), các kiểu nguyên thủy (primitive type).
* int[] myArray = new int[]{1,4,3};
* Object[] myArrayObj = new Object[] { “Object”, new Integer(100) };
  1. **Mảng có kích cỡ và số chiều cố định.**
* Khó khăn cho việc mở rộng mảng
  1. **Các phần tử được đặt và tham chiếu một cách liên tiếp nhau trong bộ nhớ.**
* Khó khăn cho việc xóa một phần tử ra khỏi mảng .
  1. **Xóa phần tử ra khỏi mảng**
* Các phần tử của một mảng được đặt liên tiếp nhau trong bộ nhớ điều đó là khó khăn khi bạn cố tình bỏ đi một phần tử nào đó trong mảng, nó mất tính liên tiếp. Thông thường một kỹ thuật mà thường sử dụng là tạo một mảng mới lưu trữ các đối tượng của mảng ban đầu và bỏ đi các phần tử không cần thiết, nhưng điều này làm giảm hiệu năng của chương trình. Với trường hợp mở rộng mảng cũng với kỹ thuật tương tự là khởi tạo một mảng mới với kích cỡ lớn hơn sau đó thì copy các phần tử mảng cũ sang cho mảng mới.

# **Chương 2: Bài tập**

1. Viết chương trình Java để tạo một danh sách mảng mới, thêm một số màu (chuỗi) và in ra tập hợp.
2. Viết chương trình Java để lặp qua tất cả các phần tử trong danh sách mảng.

3. Viết chương trình Java để chèn một phần tử vào danh sách mảng ở vị trí đầu tiên.

4. Viết chương trình Java để lấy một phần tử (tại một chỉ mục xác định) từ một danh sách mảng cho trước.

5. Viết chương trình Java để cập nhật phần tử mảng cụ thể theo phần tử đã cho.

6. Viết chương trình Java để xóa phần tử thứ ba khỏi danh sách mảng.

7. Viết chương trình Java để tìm kiếm một phần tử trong danh sách mảng.

8. Viết chương trình Java để sắp xếp một danh sách mảng cho trước.

9. Viết chương trình Java để sao chép một danh sách mảng này sang một danh sách mảng khác.

10. Viết chương trình Java để xáo trộn các phần tử trong danh sách mảng.

11. Viết chương trình Java để đảo ngược các phần tử trong danh sách mảng.

12. Viết chương trình Java để trích xuất một phần của danh sách mảng.

13. Viết chương trình Java để so sánh hai danh sách mảng.

14. Viết chương trình Java hoán đổi hai phần tử trong danh sách mảng.

15. Viết chương trình Java để nối hai danh sách mảng.

16. Viết chương trình Java để sao chép một danh sách mảng sang một danh sách mảng khác.

17. Viết chương trình Java để làm trống danh sách mảng.

**Danh sách có kết nối (Linked List)**

Danh sách có kết nối ( Linked List) là một trong các cách quản lý danh sách dữ liệu khắc phục được các nhược điểm của mảng. Tất nhiên để quản lý danh sách trong Java có nhiều cách khác ví dụ ArrayList.

Hãy xem các đặc điểm của LinkedList:

* Các phần tử trong danh sách này có thể nằm cách ly nhau (không liên tục) trong bộ nhớ .
* Nó thực sự là một liên kết có tính hai chiều giữa các phần tử.
  + Mỗi phần tử trong danh sách cầm giữ một tham chiếu đến đối phần tử đằng trước nó và tham chiếu đến phần tử ngay sau nó.

**LinkedList thực sự là một liên kết 2 chiều.**

Diagram

Description automatically generated

Phần tử Link là một đối tượng nó chứa dữ liệu bạn cần quản lý (data), và nó có 2 tham chiếu tới phần tử Link phía trước và phần tử Link phía sau nó. Cũng giống như một nhóm người xếp hàng, mỗi người chỉ cần nhớ người đứng trước họ là ai, và người đứng sau họ là ai.

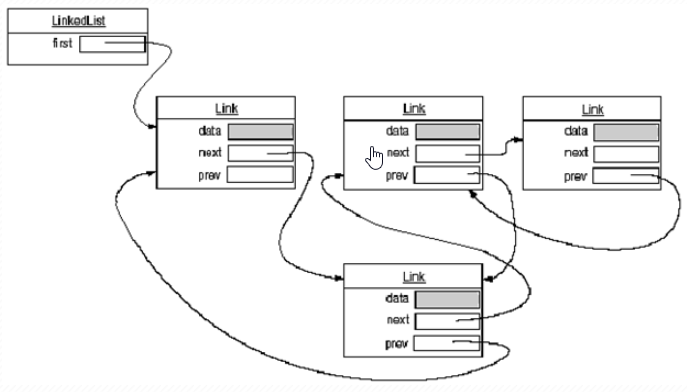
**Xóa một phần tử ra khỏi LinkedList**

Xóa một phần tử ra khỏi LinkedList cũng giống bỏ một người ra khỏi hàng đang sắp xếp, hai người đứng gần người này phải cập nhập lại thông tin người đứng trước, đứng sau họ là ai.

Diagram

Description automatically generated with medium confidence

Thêm phần tử vào LinkedList (Thêm vào cuối hoặc trèn vào giữa danh sách)



Chú ý: LinkedList là một trong các giải pháp giải quyết hạn chế của mảng, ArrayList cũng là cách quản lý tập hợp dữ liệu, giải quyết được các hạn chế của mảng, nhưng cách thức quản lý dữ liệu của nó khác.

**Bài Tập**

1. Viết chương trình Java để nối phần tử được chỉ định vào cuối danh sách liên kết.

2. Viết chương trình Java để lặp qua tất cả các phần tử trong danh sách liên kết.

3. Viết chương trình Java để lặp qua tất cả các phần tử trong danh sách liên kết bắt đầu từ vị trí đã chỉ định.

4. Viết chương trình Java để lặp một danh sách liên kết theo thứ tự ngược lại.

5. Viết chương trình Java để chèn phần tử được chỉ định vào vị trí xác định trong danh sách liên kết.

6. Viết chương trình Java để chèn các phần tử vào danh sách liên kết ở vị trí đầu tiên và cuối cùng.

7. Viết chương trình Java để chèn phần tử được chỉ định vào đầu danh sách liên kết.

8. Viết chương trình Java để chèn phần tử được chỉ định vào cuối danh sách liên kết.

9. Viết chương trình Java để chèn một số phần tử tại vị trí xác định vào danh sách liên kết.

10. Viết chương trình Java để lấy vị trí xuất hiện đầu tiên và cuối cùng của các phần tử được chỉ định trong danh sách liên kết.

11. Viết chương trình Java để hiển thị các phần tử và vị trí của chúng trong một danh sách liên kết.

12. Viết chương trình Java để xóa một phần tử được chỉ định khỏi danh sách liên kết.

13. Viết chương trình Java để xóa phần tử đầu tiên và cuối cùng khỏi danh sách liên kết.

14. Viết chương trình Java để xóa tất cả các phần tử khỏi danh sách liên kết.

15. Viết chương trình Java hoán đổi hai phần tử trong danh sách liên kết.

16. Viết chương trình Java để xáo trộn các phần tử trong danh sách liên kết.

17. Viết chương trình Java để nối hai danh sách liên kết.

18. Viết chương trình Java để sao chép một danh sách liên kết sang một danh sách liên kết khác.

19. Viết chương trình Java để loại bỏ và trả về phần tử đầu tiên của danh sách liên kết.

20. Viết chương trình Java để lấy nhưng không loại bỏ, phần tử đầu tiên của danh sách liên kết.

21. Viết chương trình Java để lấy nhưng không loại bỏ, phần tử cuối cùng của danh sách liên kết.

22. Viết chương trình Java để kiểm tra xem một phần tử cụ thể có tồn tại trong danh sách liên kết hay không.

23. Viết chương trình Java để chuyển một danh sách liên kết thành danh sách mảng.

24. Viết chương trình Java để so sánh hai danh sách liên kết.

25. Viết chương trình Java để kiểm tra danh sách liên kết có trống hay không.

26. Viết chương trình Java để thay thế một phần tử trong danh sách liên kết.