Tuyệt vời! Dưới đây là **toàn bộ kiến thức + bộ đề bài tập (đề bài chi tiết) của từng cấu trúc dữ liệu: Array, List, Stack, Queue, Tree, Set, Map** từ dễ đến khó — cực chuẩn để luyện phỏng vấn hoặc tự học. Bạn chỉ việc copy và dán vào Word để in ra hoặc lưu lại.

**📘 MỤC LỤC**

1. Array (Mảng)
2. List (Danh sách)
3. Stack (Ngăn xếp)
4. Queue (Hàng đợi)
5. Tree (Cây)
6. Set (Tập hợp)
7. Map (Ánh xạ – key/value)

**1. Array (Mảng)**

**Lý thuyết**

* Mảng là tập hợp các phần tử cùng kiểu, kích thước cố định, truy cập O(1), không thêm/xóa ở giữa dễ dàng.
* Khai báo: int[] arr = new int[5]; hoặc int[] arr = {1,2,3,4,5};
* Duyệt: for, for-each.

**Bài tập từ dễ đến khó**

1. **Tính tổng các phần tử trong mảng**  
   Viết hàm tính tổng giá trị các phần tử trong mảng số nguyên.
2. **Tìm phần tử lớn nhất, nhỏ nhất**  
   Viết hàm tìm giá trị lớn nhất và nhỏ nhất trong mảng.
3. **Đếm số chẵn, số lẻ**  
   Đếm xem có bao nhiêu phần tử chẵn, lẻ trong mảng.
4. **Tính trung bình cộng**  
   Tính trung bình cộng các phần tử mảng.
5. **Kiểm tra mảng có tăng/giảm dần không**  
   Viết hàm kiểm tra xem mảng có tăng/giảm dần hoàn toàn không.
6. **Đảo ngược mảng**  
   Viết hàm đảo ngược thứ tự các phần tử trong mảng.
7. **Sắp xếp mảng tăng/giảm dần**  
   Viết hàm sắp xếp mảng tăng/giảm dần (không dùng hàm có sẵn).
8. **Tìm phần tử lớn thứ hai**  
   Tìm giá trị lớn thứ hai (khác với giá trị lớn nhất) trong mảng.
9. **Xóa phần tử tại vị trí chỉ định**  
   Viết hàm xóa phần tử ở chỉ số k, dồn các phần tử phía sau lên.
10. **Chèn phần tử vào vị trí chỉ định**  
    Viết hàm chèn số x vào vị trí k trong mảng (phải tạo mảng mới nếu mảng đầy).
11. **Loại bỏ phần tử trùng lặp**  
    Loại bỏ các phần tử trùng trong mảng, chỉ giữ lại 1 bản sao của mỗi số.
12. **Tách mảng thành mảng số chẵn, số lẻ**  
    Viết hàm tách mảng ban đầu thành 2 mảng mới chứa số chẵn và số lẻ.
13. **Tìm cặp phần tử có tổng bằng K**  
    Cho mảng và số K, in ra mọi cặp số có tổng bằng K.
14. **Di chuyển tất cả số 0 về cuối mảng**  
    Giữ nguyên thứ tự các số khác 0.
15. **Dãy con liên tiếp có tổng lớn nhất (Kadane’s Algorithm)**  
    Tìm tổng lớn nhất của 1 dãy con liên tiếp trong mảng.
16. **Tìm phần tử xuất hiện đúng 1 lần (mọi phần tử khác xuất hiện 2 lần)**  
    Dùng XOR tối ưu.
17. **Tìm độ dài dãy con tăng liên tiếp dài nhất**  
    Trả về độ dài dãy con liên tiếp có giá trị tăng dài nhất.
18. **Tìm giá trị thiếu đầu tiên trong mảng từ 1 đến n**  
    Mỗi số xuất hiện một lần, 1 số bị thiếu.
19. **Rotate array (dịch phải/trái k vị trí)**  
    Dịch chuyển mảng sang phải (hoặc trái) k vị trí.
20. **Tìm phần tử majority (xuất hiện > n/2 lần)**  
    Nếu có, in ra phần tử đó.

**2. List (Danh sách)**

**Lý thuyết**

* List là interface Java (ArrayList, LinkedList…), cho phép lưu trữ động, trùng lặp, duy trì thứ tự.
* Các thao tác: add, remove, set, get, contains, size, clear.

**Bài tập từ dễ đến khó**

1. **Tạo danh sách, nhập và xuất các phần tử**  
   Viết hàm nhập 1 list số nguyên và in ra.
2. **Tìm phần tử lớn nhất/nhỏ nhất trong list**  
   Tìm và in ra giá trị lớn nhất, nhỏ nhất.
3. **Tính tổng, trung bình cộng list**  
   Tính tổng và trung bình các phần tử trong list.
4. **Đếm số lần xuất hiện của phần tử x**  
   Cho list, đếm x xuất hiện mấy lần.
5. **Tìm vị trí đầu/cuối của phần tử x**  
   In index đầu/cuối x xuất hiện.
6. **Kiểm tra list có chứa x không**  
   Trả về true/false.
7. **Sắp xếp list tăng/giảm dần**  
   Không dùng Collections.sort.
8. **Loại bỏ phần tử trùng lặp**  
   Trả về list mới chỉ chứa các số khác nhau.
9. **Đảo ngược list**  
   Viết hàm đảo ngược thứ tự phần tử.
10. **Chèn phần tử vào vị trí bất kỳ**  
    Thêm x vào index k.
11. **Gộp 2 list thành 1 list**  
    Gộp và loại trùng (nếu cần).
12. **Tách list thành nhiều list nhỏ kích thước k**  
    VD: list [1,2,3,4,5], k=2 → [1,2],[3,4],[5]
13. **Tìm cặp số tổng bằng K**  
    In mọi cặp có tổng đúng bằng K.
14. **Tìm phần tử xuất hiện nhiều nhất**  
    In giá trị và số lần xuất hiện lớn nhất.
15. **Kiểm tra list là Palindrome không**  
    Đọc xuôi ngược đều giống nhau?
16. **Rotate list phải/trái k vị trí**  
    Dịch phải (hoặc trái) k vị trí.
17. **Chuyển list sang set để loại trùng**  
    Sau đó chuyển lại sang list.
18. **Gom nhóm theo điều kiện (group by, vd chia nhóm tuổi)**  
    Trả về Map<Integer, List<Integer>> group by độ tuổi.

**3. Stack (Ngăn xếp)**

**Lý thuyết**

* Stack là cấu trúc LIFO (vào sau ra trước).
* Thao tác: push, pop, peek, isEmpty, size.

**Bài tập từ dễ đến khó**

1. **Push, pop, peek, kiểm tra rỗng**  
   Thực hiện và in kết quả từng thao tác.
2. **Đảo ngược chuỗi**  
   Nhập chuỗi, in ra chuỗi đảo ngược dùng stack.
3. **Kiểm tra dấu ngoặc hợp lệ**  
   Chuỗi ngoặc (, {, [ có hợp lệ không?
4. **Tính giá trị biểu thức hậu tố (postfix)**  
   VD: "2 3 1 \* + 9 -" → 2 + 3\*1 - 9 = -4
5. **Chuyển biểu thức infix sang postfix**  
   Biểu thức trung tố → hậu tố.
6. **Thiết kế ngăn xếp hỗ trợ lấy min O(1) (MinStack)**  
   Push, pop, getMin đều O(1).
7. **Reverse từng từ trong câu bằng stack**  
   VD: "I love Java" → "I evol avaJ"
8. **Kiểm tra chuỗi palindrome bằng stack**  
   VD: "level" → true
9. **Cài đặt queue dùng 2 stack**  
   Mô phỏng hàng đợi bằng 2 stack.
10. **Tìm phần tử next greater trong mảng dùng stack**  
    In ra với mỗi phần tử, phần tử đầu tiên bên phải lớn hơn nó.
11. **Tính diện tích lớn nhất của histogram**  
    Dùng stack tối ưu.
12. **Thiết kế stack dùng linked list tự tạo**

**4. Queue (Hàng đợi)**

**Lý thuyết**

* FIFO (vào trước ra trước).
* offer (enqueue), poll (dequeue), peek, size, isEmpty.

**Bài tập từ dễ đến khó**

1. **Thêm, lấy ra, kiểm tra rỗng, in ra hàng đợi**
2. **Đảo ngược queue**  
   Đảo ngược thứ tự các phần tử queue.
3. **Duyệt cây theo BFS dùng queue**  
   In ra node theo từng mức.
4. **Mô phỏng hàng chờ khách hàng**  
   Cho biết thời điểm khách đến, xử lý từng người.
5. **Cài đặt stack dùng 2 queue**  
   Ngược lại với queue bằng 2 stack.
6. **Đảo ngược K phần tử đầu của queue**  
   K nhập từ bàn phím.
7. **Josephus problem (vòng tròn loại người)**  
   Cho N người, loại bỏ theo quy tắc, ai sống sót cuối cùng.
8. **Sliding window max/min dùng deque**  
   Tìm max/min trong mỗi cửa sổ kích thước k.
9. **LRU cache dùng queue + map**  
   Cache dữ liệu loại bỏ phần tử ít dùng nhất.

**5. Tree (Cây)**

**Lý thuyết**

* Cấu trúc phân cấp: root, child, leaf, parent.
* Duyệt: Pre-, In-, Post-, Level-order.
* Binary Tree, BST, Heap, Trie,…

**Bài tập từ dễ đến khó**

1. **Duyệt cây các kiểu (Pre, In, Post, Level order)**
2. **Tính chiều cao (height) của cây**
3. **Đếm số node, số lá**
4. **Tìm node lớn nhất/nhỏ nhất BST**
5. **Kiểm tra cây có là BST không**
6. **Tính tổng tất cả giá trị node**
7. **Tìm ancestor chung thấp nhất (LCA)**
8. **Đếm số node ở level K**
9. **Kiểm tra cây cân bằng**
10. **Kiểm tra cây đối xứng**
11. **Tìm đường đi có tổng lớn nhất/nhỏ nhất**
12. **Flatten binary tree thành linked list**
13. **Invert/mirror tree**
14. **Đếm số đường đi từ root tới leaf có tổng bằng K**
15. **Kiểm tra cây là Complete/Full/Perfect**

**6. Set (Tập hợp)**

**Lý thuyết**

* Không trùng lặp, HashSet, TreeSet, LinkedHashSet.

**Bài tập từ dễ đến khó**

1. **Loại bỏ phần tử trùng lặp trong list/array**
2. **Đếm số lượng giá trị khác nhau**
3. **Kiểm tra mảng có phần tử trùng lặp không**
4. **Tìm giao, hợp, hiệu của 2 tập hợp**
5. **Tìm phần tử đầu tiên bị lặp lại**
6. **Tìm các giá trị chỉ xuất hiện đúng 1 lần**
7. **Cho 2 list, tìm các phần tử chung**
8. **Kiểm tra chuỗi có chứa đủ 26 ký tự không (pangram)**
9. **Tìm số lượng từ khác nhau trong văn bản**
10. **Tìm tất cả cặp số có tổng bằng K dùng set**
11. **Bài toán sliding window đếm distinct trong mỗi cửa sổ**
12. **Tìm số bị thiếu từ 1...n**

**7. Map (Bản đồ khóa–giá trị)**

**Lý thuyết**

* Lưu cặp key–value, key duy nhất.
* HashMap, LinkedHashMap, TreeMap.

**Bài tập từ dễ đến khó**

1. **Đếm tần suất xuất hiện của phần tử**
2. **Tìm phần tử xuất hiện nhiều nhất**
3. **Tìm key theo value và ngược lại**
4. **Kiểm tra 2 mảng có phần tử chung không**
5. **Gom nhóm (group by) phần tử theo điều kiện**
6. **Tìm value lớn nhất/nhỏ nhất**
7. **Đảo key-value của Map (chú ý value trùng)**
8. **So sánh 2 Map có giống nhau không**
9. **Tìm majority element (xuất hiện > n/2 lần)**
10. **Sắp xếp map theo value**
11. **Bài toán Two Sum: tìm 2 số tổng bằng k dùng map**
12. **LRU cache dùng LinkedHashMap**
13. **Anagram group: gom nhóm chuỗi cùng ký tự**
14. **Đếm từ trong file văn bản**
15. **Tạo bản đồ id → object (StudentId → Student)**