# Въпроси за интервю – -Структури от данни

1. **Опишете Linked List? – Линейна структура от данни, в която всеки елемент има стойност и референция към следващия/предишният**
    - **Single Linked List – всеки елемент има стойност и референция към следващият елемент, последният е NULL**
    - **Double Linked List – всеки елемент има стойност и референция към предишния и следващия елемент, първият и последният са NULL**
    - **Circular Linked List – последният елемент има референция към първият и няма NULL елемент**
    - **Time Complexity - Lookup: O(n), Insert: O(1), Delete: O(1)**
2. **Опишете Stack? – Линейна структура от данни, която се базира на метода LIFO (Last in First out), т.е. последният влязал елемент ще излезе първи (Рекурсията е имплементирана със Стек)**
    - **.push(E element), .pop(), .peek()**
3. **Опишете Queue? – Линейна структура от данни, която се базира на метода FIFO (First in First out), т.е. първият влязал елемент излезе първи**
    - **.add(E element), .poll()**
4. **Опишете ArrayDeque? – resizable масив, който имплементира Deque интерфейса, който е по-бърз от Стек и Свързан списък, когато е използван като опашка**
5. **Опишете PriorityQueue? – Опашка, в която елементите се връщат в сортират ред, но може да не бъдат сортиране, връща се елементът с най-висок приоритед**
6. **Опишете BinarySearchTree? – Нелинейно, базирано на nodes, двойчно дърво, което не съдържа дублиращи елементи**
    - **Лявото поддърво съдържа само елементи с по-малка стойност от стойността на корена**
    - **Дясното поддърво съдържа само елементи с по-голяма стойност от стойността на корена**
    - **Average: O(log N)**
    - **Worst: O(n), когато има разлика във височините на поддървета**

7. **Опишете BinaryHeap? – Complete дърво (всички левели са запълнение, възможно е единствено последният да не бъде запълнен, но елементите са възможно най-вляво**
8. **Кое дърво е балансирано? – Балансирано дърво е дърво, чиито листа са с не повече от една единица дистанция от корена, сравнение с другите листа**
9. **Опишете AVL Tree/Red-Black Tree? – Self-Balancing Binary Search Tree**
    - **Worst & Average Time Complexity: O(log N)**
10. **Опишете Hash Map? – Линейна структура от данни, която предоставя mapping на key & value с помощта на hashing (keys are unique)**
    - **Resizable – ресайзва се, когато се запълни threshold-да (75% от капацитета, който по подразбиране е 16 и се увеличава със степени на двойката)**
    - **Average Time Complexity: O(1)**
    - **Worst Time Complexity: O(n), постига се, когато имаме hash collision**
    - **Hash Collision – когато два обекта имат еднакакъв хеш код**
11. **Как се справяме с Hash Collison?**
    - **Open Addressing – намираме ново място в хеш таблицата за обекта чрез probing функция. Използва се, когато броят на елементите е ясен**
    - **Separate Chaining – всеки ‘bucket’ е някаква структура от данни (често това е свързан списък) и елементите се добавят в него. Използва се, когато броят не елементите не е ясен**
12. **Каква е разликата между Array List & Linked List?**

| Array List | Linked List |
|---|---|
| Lookup: O(1) | Lookup: O(n) |
| Insert & Delete: O(n) | Insert & Delete: O(1) |
| Съдържат индекси | Съдържат стойност и референция към следващ( повече memory consumption) |
| Използва се, когато имаме по-рядко добавяне и изкарване на елемент, но по-често lookup | Използва се, когато имаме често добавяне и изкарване на елемент |