**ArrayList vs. LinkedList**

[Java Syntax](https://javarush.ru/quests/QUEST_JAVA_SYNTAX)

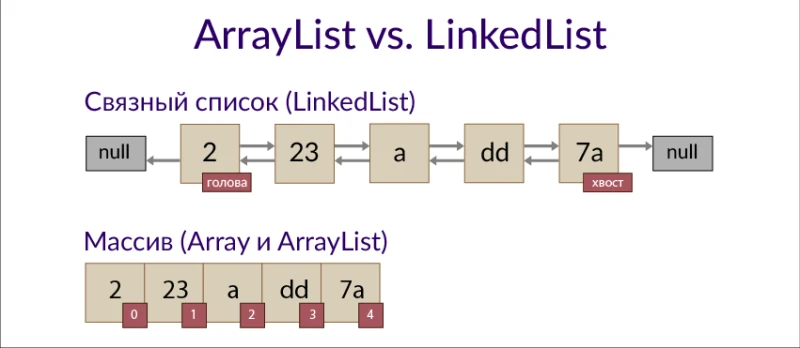
[Уровень 8](https://javarush.ru/quests/lectures?quest=QUEST_JAVA_SYNTAX&level=8), Лекция 5

— Как насчёт немного размять мозги? Надеюсь, они ещё не закипели.

— В таблице контейнеров и коллекций ты ранее видел, что у одного и того же интерфейса может быть несколько реализаций. Сейчас я расскажу тебе, зачем это нужно. И в чем отличие **ArrayList** от **LinkedList**.

— Все дело в том, что коллекции могут быть реализованы разными способами и нет единственного – самого правильного. При одном подходе одни операции являются быстрыми, а остальные медленными, при другом – все наоборот. Нет одного идеального, подходящего всем решения.

— Поэтому было решено сделать несколько реализаций одной и той же коллекции. И каждая реализация была оптимизирована для какого-то узкого набора операций. Так появились разные коллекции. Давай рассмотрим это на примере двух классов – [**ArrayList**](https://javarush.ru/quests/lectures/questsyntax.level07.lecture05) и **LinkedList**.

[](https://cdn.javarush.ru/images/article/bb532538-c090-4b06-a882-eb3c50d43f9e/original.jpeg)

— ***ArrayList*** реализован внутри в виде обычного массива. Поэтому при вставке элемента в середину, приходится сначала сдвигать на один все элементы после него, а уже затем в освободившееся место вставлять новый элемент. Зато в нем быстро реализованы взятие и изменение элемента – операции get, set, так как в них мы просто обращаемся к соответствующему элементу массива.

— ***LinkedList*** реализован внутри по-другому. Он реализован в виде связного списка: набора отдельных элементов, каждый из которых хранит ссылку на следующий и предыдущий элементы. Чтобы вставить элемент в середину такого списка, достаточно поменять ссылки его будущих соседей. А вот чтобы получить элемент с номером 130, нужно пройтись последовательно по всем объектам от 0 до 130. Другими словами операции set и get тут реализованы очень медленно. Посмотри на таблицу:

|  |  |  |  |
| --- | --- | --- | --- |
| **Описание** | **Операция** | **ArrayList** | **LinkedList** |
| Взятие элемента | get | Быстро | Медленно |
| Присваивание элемента | set | Быстро | Медленно |
| Добавление элемента | add | Быстро | Быстро |
| Вставка элемента | add(i, value) | Медленно | Быстро |
| Удаление элемента | remove | Медленно | Быстро |

— Ага. Кое-что начинает проясняться. А есть какие-нибудь критерии или правила, когда какая коллекция лучше?

— Ну, для простоты, я бы сформулировала такое правило: если ты собираешься вставлять (или удалять) в середину коллекции много элементов, то тебе лучше использовать **LinkedList**. Во всех остальных случаях – **ArrayList**.

— Как они устроены мы разберем в старших уровнях, а пока будем учиться ими пользоваться.