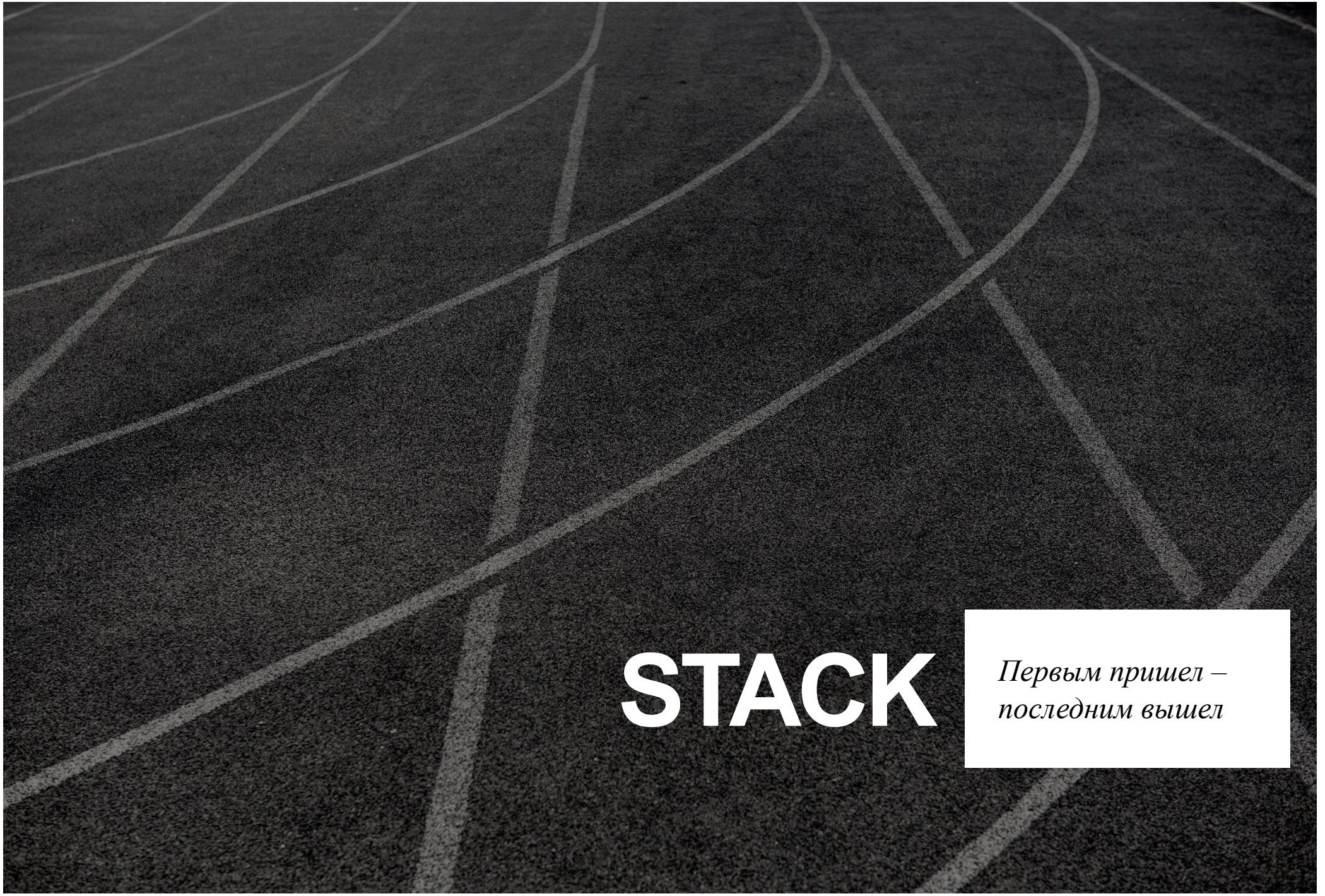
Сергей Савчук
software engineer

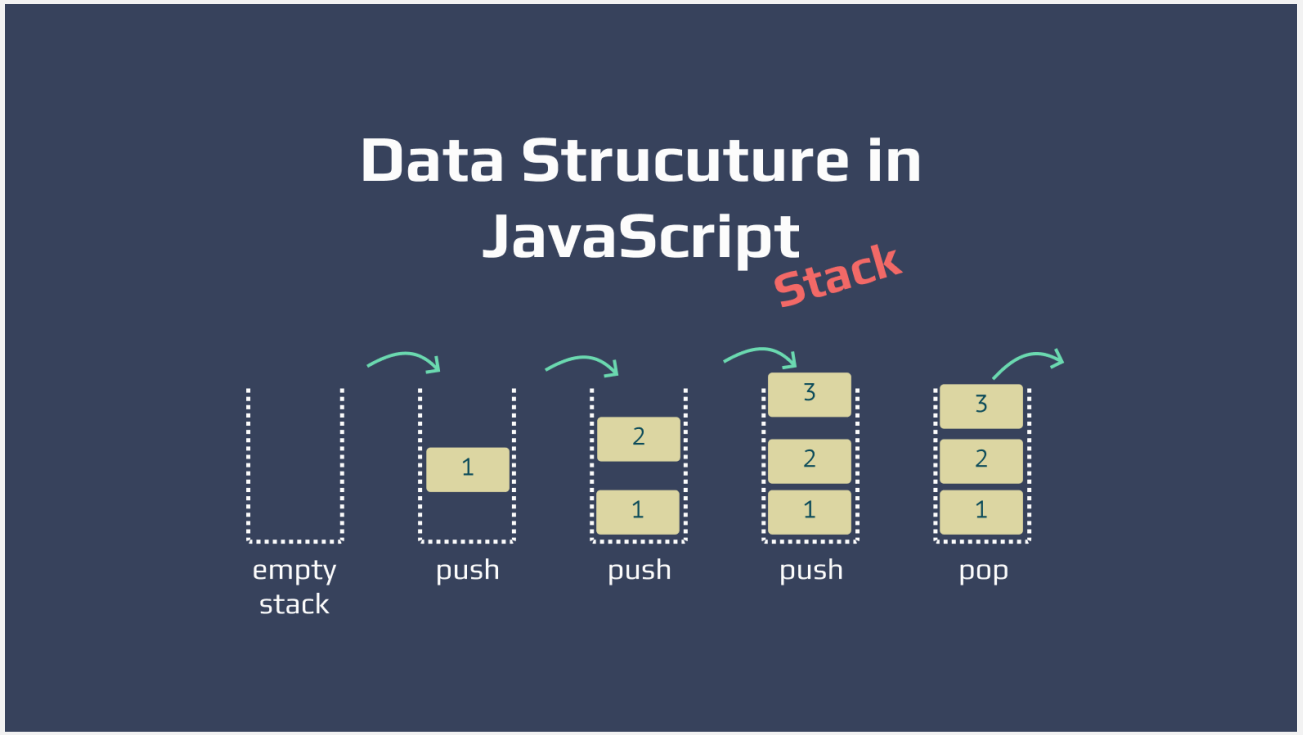
СТРУКТУРЫ ДАННЫХ

цели структуры данных призваны помочь нам организовать данные.



STACK

Первым пришел – последним вышел



STACK

это базовая, линейная структура данных, которая позволяет добавлять или удалять элементы только в её начале.

Основные операции

- Push – добавить.
- Pop – удалить.
- toArray – преобразовать в массив и показать.
- Peek – забрать.

СЛОЖНОСТЬ ОПЕРАЦИЙ

Временная сложность стека

Алгоритм	Среднее значение	Худший случай
Space	O(n)	O(n)
Search	O(n)	O(n)
Insert	O(1)	O(1)
Delete	O(1)	O(1)

ВЫВОД

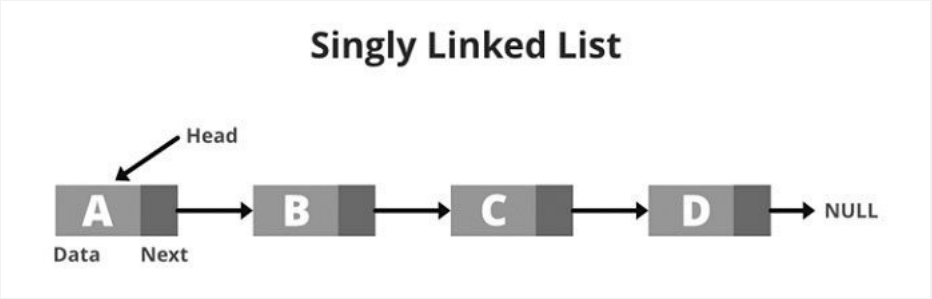
Стэк полезен

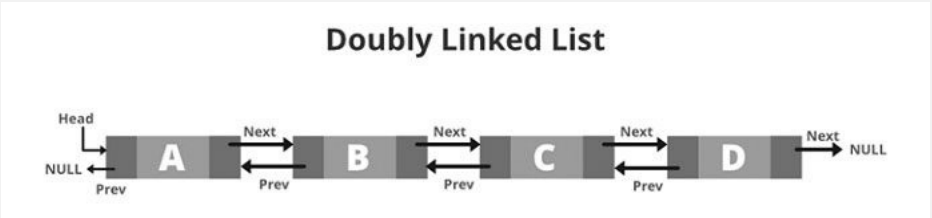
когда мы хотим добавить данные в последовательном порядке и удалить эти данные. Основываясь на своем определении, стек может удалить только самые последние добавленные данные.

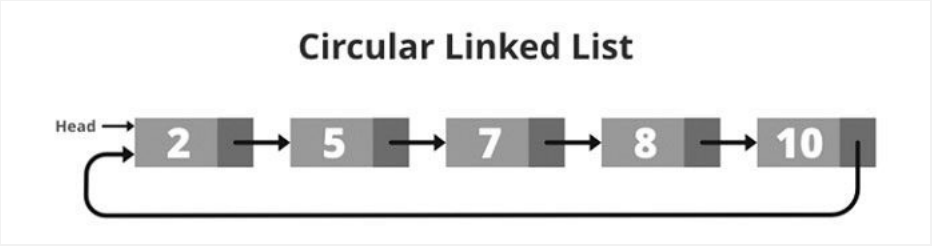
Пример: call stack

СВЯЗАННЫЙ СПИСОК

несколько значений хранятся последовательно







СВЯЗАННЫЙ СПИСОК

группы узлов, которые вместе образуют последовательность. Каждый узел содержит две вещи: фактические данные, которые в нем хранятся и указательна следующий узел в последовательности.

Основные операции

- Добавление.
- Удаление.
- Поиск элемента.
- Получение длинны
- Проверка на пустоту
- Трансформация в массив

СЛОЖНОСТЬ ОПЕРАЦИЙ

Временная сложность связного списка

Алгоритм	Среднее значение	Худший случай
Space	O(n)	O(n)
Search	O(n)	O(n)
Insert	O(1)	O(1)
Delete	O(1)	O(1)

ВЫВОД

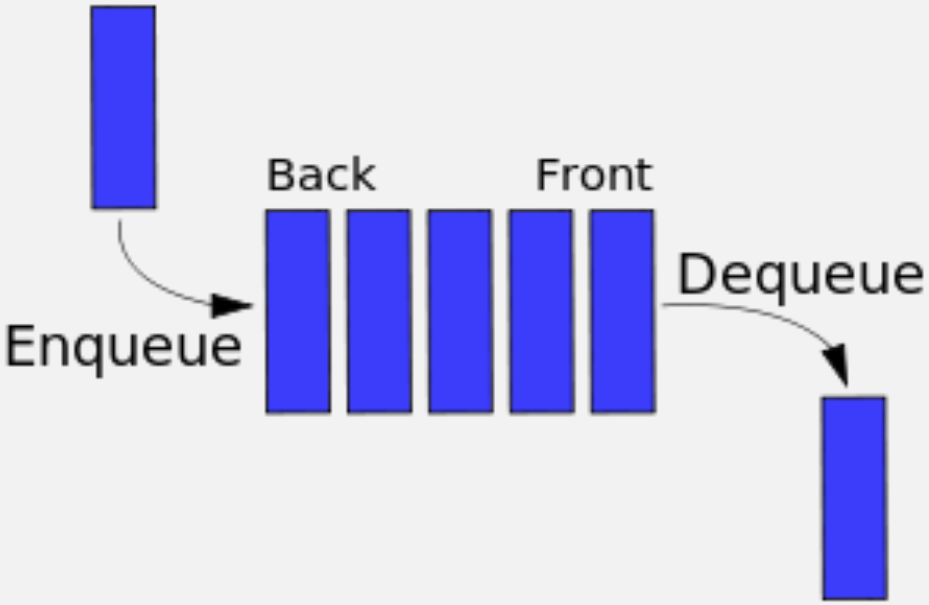
Связные списки

Концепция использования узлов, которые указывают друг на друга, используется во многих других структурах данных, встроенных во многие языки программирования более высокого уровня. Хорошее понимание того, как работают связные списки, важно для общего понимания о создании и использовании других структур данных.

Пример: слушатели событий в C++

QUEUE

Первым пришел – первым вышел



QUEUE

Это значит, что удалить элемент можно только после того, как были убраны все ранее добавленные элементы.

Основные операции

- Enqueue – добавить в конец.
- Dequeue – удалить первый элемент.
- Peek – забрать первый.

СЛОЖНОСТЬ ОПЕРАЦИЙ

Временная сложность очереди

Алгоритм	Среднее значение	Худший случай
Space	O(n)	O(n)
Search	O(n)	O(n)
Insert	O(1)	O(1)
Delete	O(1)	O(1)

ВЫВОД

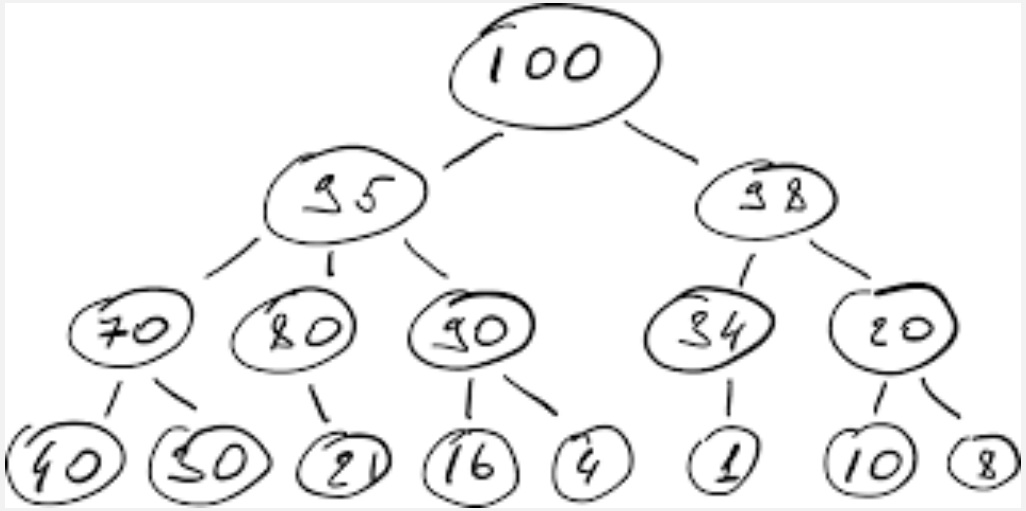
Стэк полезен

Очередь хранит данные в последовательном порядке, но удаляет самые старые добавленные данные.

Пример: события веб браузера

БИНАРНОЕ ДЕРЕВО

двоичное



БИНАРНОЕ ДЕРЕВО

Это структура данных, состоящая из узлов. Ей присущи следующие свойства:

1. Каждое дерево имеет корневой узел (вверху).

2. Корневой узел имеет ноль или более дочерних узлов.

3. Каждый дочерний узел имеет ноль или более дочерних узлов, и так далее.

Основные операции

- Добавление.
- Поиск

СЛОЖНОСТЬ ОПЕРАЦИЙ

Временная сложность двоичного дерева поиска

Алгоритм	Среднее значение	Худший случай
Space	O(n)	O(n)
Search	O(log n)	O(n)
Insert	O(log n)	O(n)
Delete	O(log n)	O(n)

ВЫВОД

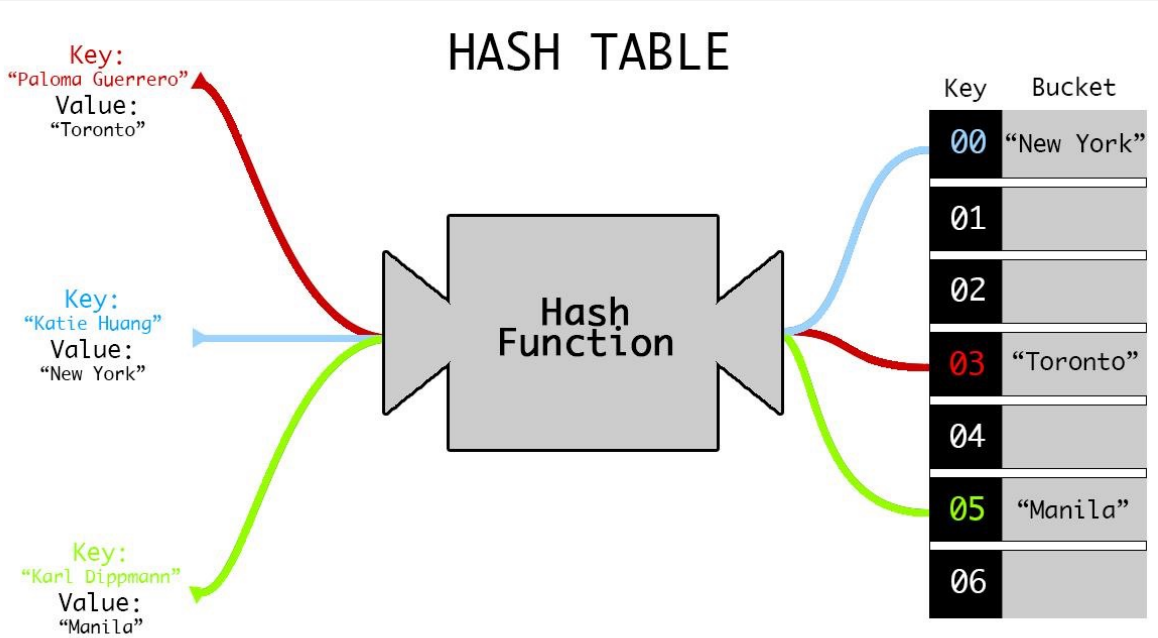
Двоичное дерево

Двоичные деревья поиска позволяют быстро находить, добавлять и удалять элементы. Они устроены так, что время каждой операции пропорционально логарифму общего числа элементов в дереве.

Пример: слушатели событий в C++

ХЭШ ТАБЛИЦЫ

Первым пришел – первым вышел



ХЭШ ТАБЛИЦЫ

представляет собой структуру данных, которая отображает ключи на значения.

Она использует хэш-функцию для вычисления индекса в массиве из блоков данных, чтобы найти желаемое значение.

Обычно хэш-функция принимает строку символов в качестве вводных данных и выводит числовое значение. Для одного и того же ввода хэш-функция должна возвращать одинаковое число.

Основные операции

- Insert – добавить.
- Search – найти.
- Remove – удалить.
- Print - показать

СЛОЖНОСТЬ ОПЕРАЦИЙ

Временная сложность хэш-таблицы

Алгоритм	Среднее значение	Худший случай
Space	O(n)	O(n)
Search	O(1)	O(n)
Insert	O(1)	O(n)
Delete	O(1)	O(n)

ВЫВОД

Стэк полезен

Очередь хранит данные в последовательном порядке, но удаляет самые старые добавленные данные.

Пример: события веб браузера

THANK YOU