Алгоритмы и структуры данных

Введение

Обо мне

Крахмалёв Денис Сергеевич

Аспирант ФПМИ МФТИ

CV-Engineer в PicsArt

krakhmalev23@gmail.com

vk.com/d_krakhmalev

tg: @krakhmalyov

Чат курса
https://t.me/joinchat/R_TkIPBYtH8wNWIy

Алгоритм

Это последовательность команд, предназначенная исполнителю, в результате выполнения которой он должен решить поставленную задачу.

Примеры простых алгоритмов

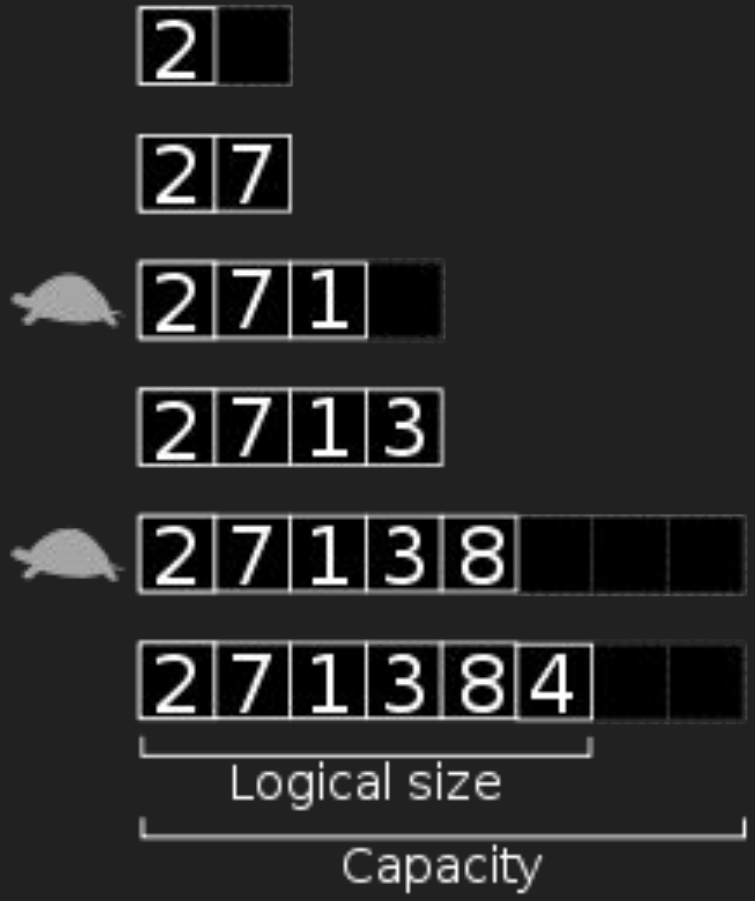
- вычисление факториала числа
- проверка числа на простоту
- быстрое возведение в степень

Структура и Абстрактный тип данных

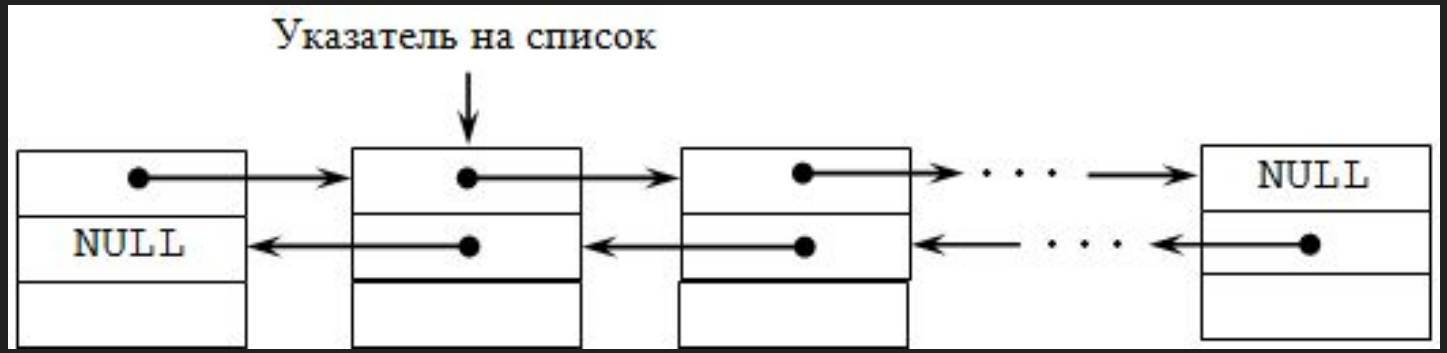
Структура -- программная единица, позволяющая хранить и обрабатывать множество однотипных и/или логически связанных данных в вычислительной технике. Для добавления, поиска, изменения и удаления данных структура данных предоставляет некоторый набор функций, составляющих её интерфейс.

Абстра́ктный тип да́нных (АТД) — это математическая модель для типов данных, где тип данных определяется поведением (семантикой) с точки зрения *пользователя* данных, а именно в терминах возможных значений, возможных операций над данными этого типа и поведения этих операций.

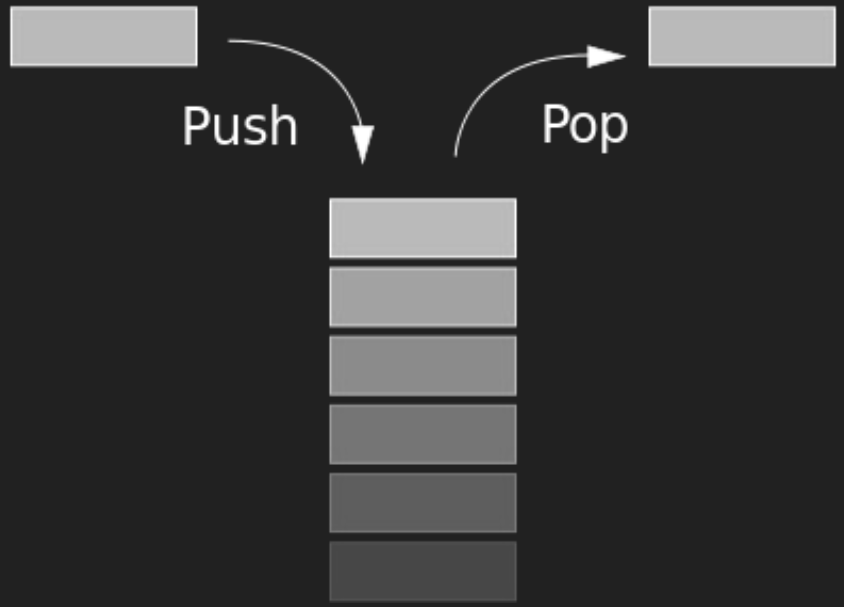
Динамический массив

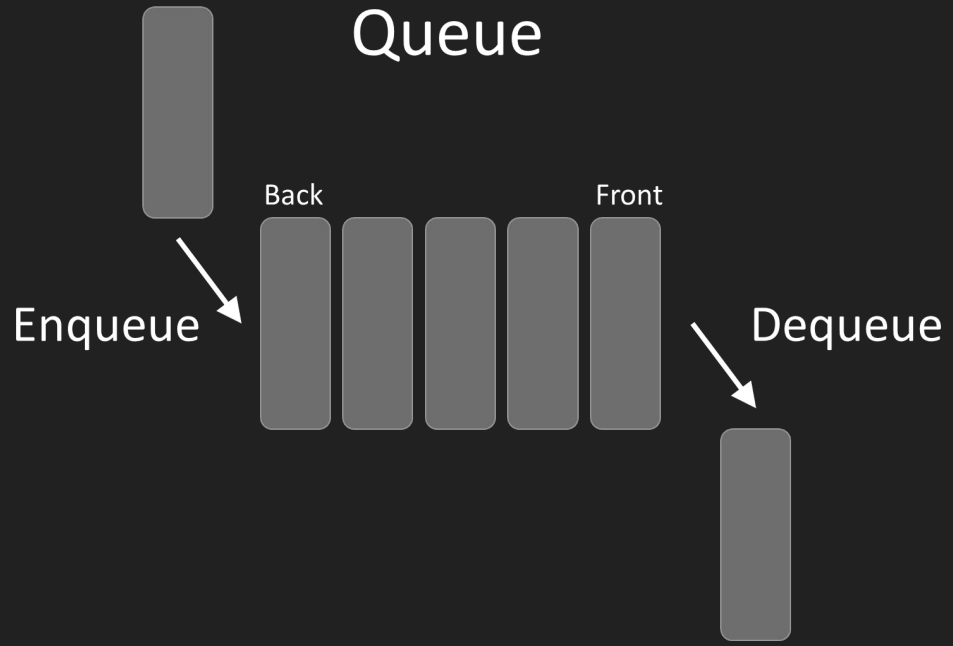


Двусвязный и односвязный списки



Стек и очередь





Двусторонняя очередь (дек)

Хранение стека, дека, очередь в массиве/списке

Поиск элемента в массиве

- Линейный
- Бинарный

Поддержка минимума в стеке

Асимптотические обозначения

Для функций $f(n)$ и $g(n)$ при $n \to n_0$ используются следующие обозначения:

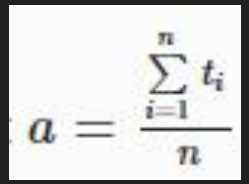
Обозначение	Интуитивное объяснение	Определение
$f(n) \in O(g(n))$	f ограничена сверху функцией g (с точностью до постоянного множителя) асимптотически	$\exists (C > 0), U : \forall (n \in U)\ \lvert f(n)\rvert \leq C\lvert g(n)\rvert$
$f(n) \in \Omega(g(n))$	f ограничена снизу функцией g (с точностью до постоянного множителя) асимптотически	$\exists (C > 0), U : \forall (n \in U)\ C\lvert g(n)\rvert \leq \lvert f(n)\rvert$
$f(n) \in \Theta(g(n))$	f ограничена снизу и сверху функцией g асимптотически	$\exists (C > 0), (C' > 0), U : \forall (n \in U)\ C\lvert g(n)\rvert \leq \lvert f(n)\rvert \leq C'\lvert g(n)\rvert$
$f(n) \in o(g(n))$	g доминирует над f асимптотически	$\forall (C > 0), \exists U : \forall (n \in U)\ \lvert f(n)\rvert < C\lvert g(n)\rvert$
$f(n) \in \omega(g(n))$	f доминирует над g асимптотически	$\forall (C > 0), \exists U : \forall (n \in U)\ C\lvert g(n)\rvert < \lvert f(n)\rvert$
$f(n) \sim g(n)$	f эквивалентна g асимптотически	$\lim\limits_{n \to n_0} \dfrac{f(n)}{g(n)} = C$

где U — проколотая окрестность точки n_0.

Асимптотика простых алгоритмов

Амортизационный анализ

Средняя амортизационная стоимость операций — величина a, находящаяся по формуле:



t1,t2…tn — время выполнения операций 1,2…n, совершённых над структурой данных.

1. Метод усреднения
2. Метод предоплаты
3. Метод потенциалов

Метод усреднения

Считаем среднюю амортизационную стоимость по формуле

Метод предоплаты

Каждой операции над структурой присваивается “стоимость”

При проведении операции от этой стоимости отнимается фактическое время работы операции, остаток кладётся в “банк”
Если стоимости не хватает на проведение операции, то можно взять часть из банка

Тогда амортизационная стоимость операций -- все затраченные деньги/количество операций

Метод потенциалов

Теорема (О методе потенциалов):

Введём для каждого состояния структуры данных величину Φ — потенциал. Изначально потенциал равен Φ_0, а после выполнения i-й операции — Φ_i. Стоимость i-й операции обозначим $a_i = t_i + \Phi_i - \Phi_{i-1}$. Пусть n — количество операций, m — размер структуры данных. Тогда средняя амортизационная стоимость операций $a = O(f(n, m))$, если выполнены два условия:

1. Для любого $i : a_i = O(f(n, m))$
2. Для любого $i : \Phi_i = O(n \cdot f(n, m))$

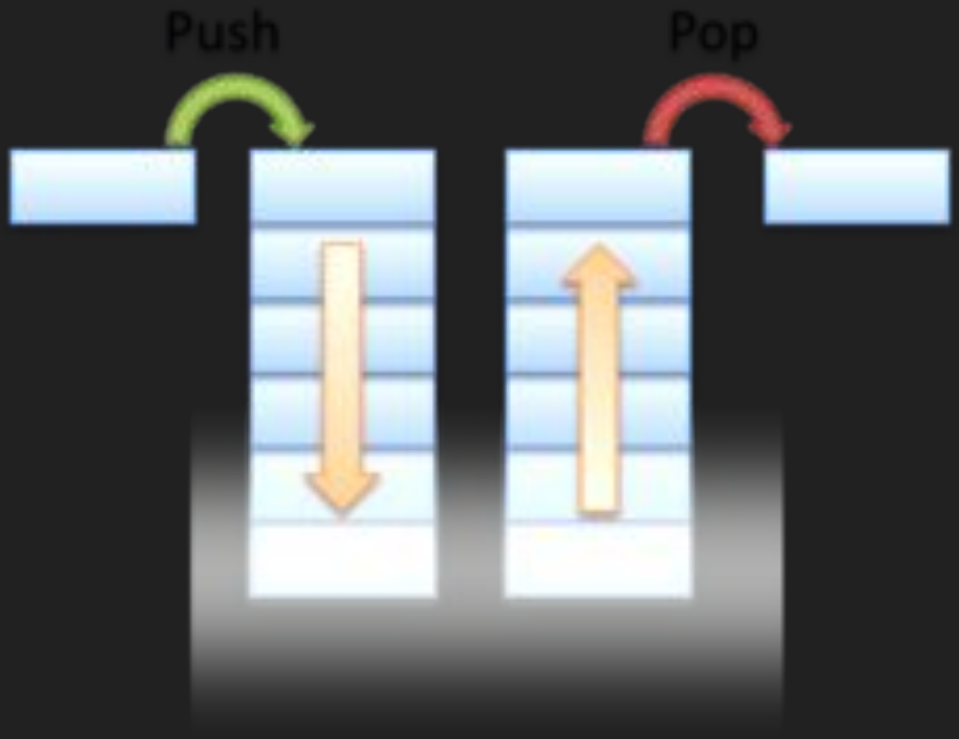
Доказательство:

▷

$$a = \frac{\sum\limits_{i=1}^{n} t_i}{n} = \frac{\sum\limits_{i=1}^{n} a_i + \sum\limits_{i=0}^{n-1} \Phi_i - \sum\limits_{i=1}^{n} \Phi_i}{n} = \frac{n \cdot O(f(n,m)) + \Phi_0 - \Phi_n}{n} = O(f(n, m))$$

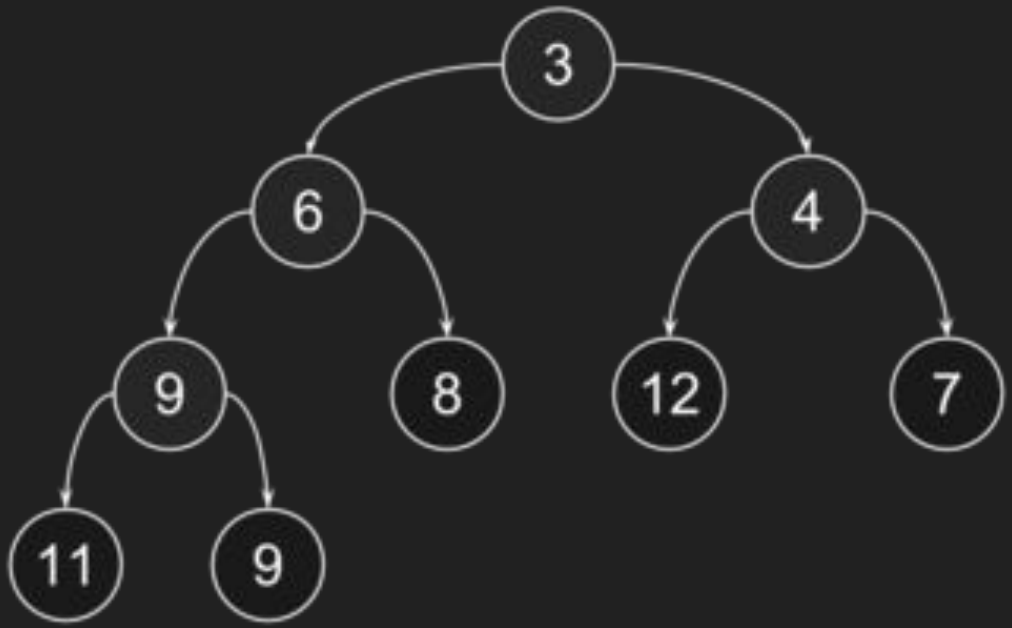
◁

Представление очереди в виде двух стеков



Поддержка минимума в очереди

Двоичная куча. АТД “Очередь с приоритетом”



Персистентный стек

- Персистентный -- значит, хранит все свои состояния, начиная с создания объекта
- Интерфейс:
  - int push(Object, int state)
  - int pop(int state)
- Квадратичная реализация
- Линейная реализация