Изпитна тема № 7

Алгоритми и структури от данни

**План-тезис:** Алчни алгоритми. Рекурсия и рекурсивни алгоритми. Комбинаторни алгоритми: вариации, комбинации, пермутации. Динамично оптимиране.

* Познава алчни алгоритми

**Greedy** подходът включва изграждане на решение чрез избиране на последователни стъпки, всяка от които произвежда частично решение на задачата, до получаване на цялостното решение. В същото време на всяка стъпка изборът трябва да бъде допустим, т.е. да отговаря на ограниченията на задачата, локално оптимален, т.е. да е най-добрият локален избор между всички възможни варианти, налични на всяка стъпка, окончателен, т.е. веднъж направен, не може да променя следващите стъпки на алгоритъма. Обикновено тези алгоритми по-ефективни от другите алгоритми, но може да доведат и до не толкова оптимален резултат.

* Дефинира понятието рекурсия и познава рекурсивни алгоритми

**Рекурсията** е техника за решаване на задачи чрез разделя задачата на подзадачи от същия тип. Решението включва функция, извикваща себе си, която трябва да има граничен случай. Всяка стъпка от рекурсията трябва да се движи към граничния случай.

В пряката рекурсия методът А директно се обръща към себе си.

В непряката рекурсия А се обръща към В, В се обръща към А.

Някои задачи, които се решават с рекурсия:

* Редица на Фибоначи
* Намиране на най-голям общ делител
* Намиране на най-малко общо кратно

**Backtracking** - клас алгоритми за намиране на всички решения на някаква комбинаторна задача (напр. намиране на всички пътища от София до Варна). На всяка стъпка рекурсивно се преглеждат всички перспективни възможности. Всички не перспективни възможности се отхвърлят възможно най-рано. Времето на работа е експоненциално.

**Backtracking алгоритъм:**

1 стъпка: Конструира се едно частично решение.

2 стъпка: Проверява се дали частичното решение е общо (търсеното). Ако е така, решението се запомня или извежда и процесът на търсене или завършва, или продължава по същата схема, докато бъдат генерирани всички възможни решения.

3 стъпка: В противен случай се прави опит текущото частично решение да се продължи (според условието на задачата).

4 стъпка: Ако на някоя стъпка се окаже невъзможно ново разширяване, извършва се връщане назад към предишното частично решение и се прави нов опит то да се разшири (продължи) по друг, различен от предишния, начин.

* Различава комбинаторни алгоритми: вариации, комбинации, пермутации

**Пермутации**

Определение: Нека А е множество и |А|=n. Всяко подреждане на всичките n елемента на А (или всички различни подреждания на първите n естествени числа) се нарича пермутация без повторение от n-ти ред. Две пермутации се различават една от друга по реда на елементите, участващи в тях.

Теорема: Броят на всички различни пермутации от n-ти ред е:

Pn = 1\*2\*3…n= n!

По определение се приема, че 0!=1.

**Комбинации без повторение**

Определение: Всички различни ненаредени извадки без повторение на n елемента от k-ти клас се наричат комбинации без повторение на n елемента от k-ти клас. Две комбинации без повторение се различават една от друга по елементите, участващи в тях.

**Вариации без повторения**

Определение: Всички различни наредени извадки без повторение на n елемента от k-ти клас наричаме вариации без повторение на n елемента от k -ти клас. Две вариации без повторение се различават една от друга или по реда на участващите в тях елементи или по елементите, участващи в тях.

* Разпознава задачи за динамично оптимиране

**Методът „разделяй и владей“** е един от най-важните и най-широко приложими методи за проектиране на ефективни алгоритми. Той е метод на декомпозицията - разделя дадена задача с размер N на по-малки задачи, на чиято основа се получава решението на първоначалната задача.

**Динамичното оптимиране** е метод за решаване на задачи с припокриващи се подзадачи. Изграждат на рекурентни връзки, свързващи решението на задачата с решенията на по-малки подзадачи от същия тип. Решава се всяка една от по-малките подзадачи само веднъж, записват се резултатите в таблица, от която след това се получава решение на първоначалната задача.

Динамичното решение използва вече пресметнатите стойности за по-малки стойности на търсения аргумент, без да се пресмята наново. Използват се рекурсивни формули, но стойностите се вземат от някаква структура от данни, в която са били попълнени в момента на пресмятането си. Сложността на такива задачи е линейна, с изключение на задачите, които изискват поддържането на матрица за запазване на стойности, за които сложността е квадратна.