Комбинаторика

Принцип на Дирихле

- Нека имаме 15 предмета и 10 чекмеджета.
  Трябва да разпределим всички предмети, така че да не останат празни чекмеджета. При това положение винаги ще има чекмедже с поне 2 предмета в него.
- Друг пример: От биологията е известно, че човешката коса се състои приблизително от 200000 косъма.
  - => В Бургас има поне 2-ма човека с еднакъв брой косми на главата си
    - населението на Бургас е малко над 203 000 души към 2019-та
- Нарича се още Принцип на чекмеджетата или Pigeonhole principle

Принцип на Дирихле

Комбинаторика

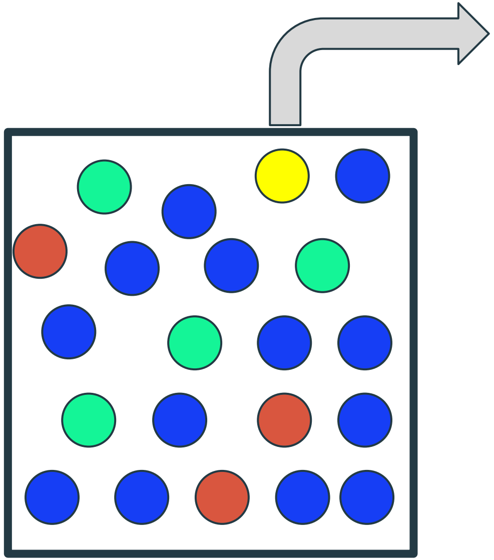
- Комбинаториката е изкуството да броим неща (групи от неща)
- Примери:
  - Въпросът „по колко начина?”
  - Въпросът „колко …, които … съществуват?”
  - Въпросът „съществува ли …, така че… ?” и др.
  - Има сериозно значение за игрите с карти, хазартните игри и др.
- Комбинаториката е пряко свързана с Теорията на вероятностите, както и Теорията на графите и редица други клонове на математиката
- Комбинаториката е от сериозно значение и за работата на програмиста.

Означение в комбинаториката

Комбинаториката обикновено работи с множества и техните елементи. В този смисъл:

- N - общ брой елементи в множеството
- K - избран брой елементи от множеството
  - Обикновено N е фиксирано, а K е в зависимост от разглеждания проблем
- Комбинаторно съединение - избрани елементи от дадено множество, съгласно правилата на комбинаториката
  - С повторение и без повторение
  - Подредени и неподредени

Какво означава с повторение и без повторение?



Ако вземем топче от кутията ще го върнем ли вътре след като сме видели цвета му?

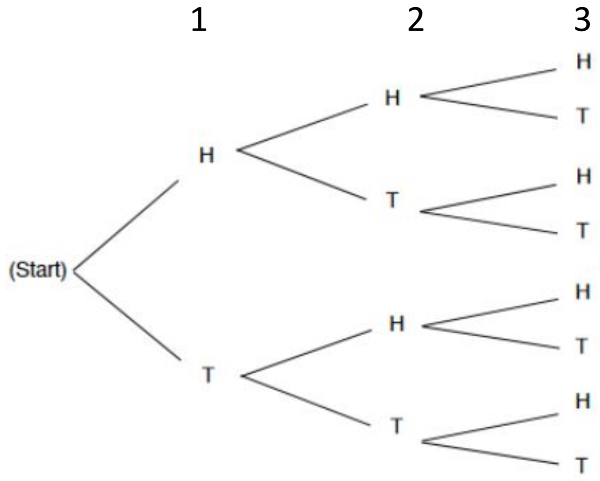
- Да => **с повторение**
  - Възможно е отново да изтеглим същото топче при последващо теглене
- Не => **без повторение**
  - Не е възможно да изтеглим отново същото топче

Правила в комбинаториката

- Правило на събирането
  - Възможно е да направим избор на обект A по **n** начина, а на друг обект B - по **m** начина. Тогава общия брой на възможните избори, включващи A и B е **n + m**
- Правило на умножението
  - Възможно е да направим избор на обект A по **n** начина, след всеки такъв избор можем да изберем обект B по **m** начина. Тогава общия брой възможни избори е **n * m**

Пример: 3 хвърляния на монети

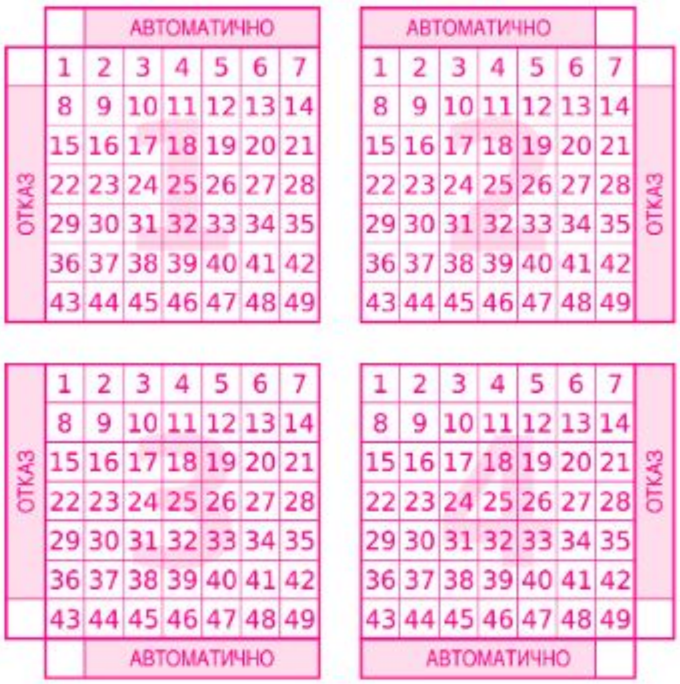
- Част от комбинаторните задачи могат да се решат чрез диаграма на дървото
    - Рисуват се всички междинни резултати
    - Път през дървото показва един възможен изход
    - Удачен начин, когато възможностите са малко
- Пример: Каква е вероятността при 3 хвърляния да се падне тора (tails) и трите пъти?
    - Отговор: 1/8



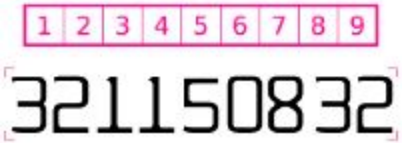
Пример: Хапване в ресторант

- Ресторант предлага:
  - 5 предястия
  - 8 основни ястие
  - 4 десерта
  - Може да изберете 1 курс, 2 курса или всичките 3 курса едновременно.
- По колко възможни начина може да се стори това?
  - Един курс = предястие или основно или десерт = 5 + 8 + 4 = 17
  - Два курса = предястие и основно, предястие или десерт, основно или десерт
    - Предястие и основно = 5 * 8 = 40
    - Предястие и десерт = 5 * 4 = 20
    - Основно и десерт = 8 * 4 = 32
  - Три курса = предястие, основно и десерт = 5 * 8 * 4 = 160
  - Общо = 17 + 92 + 160 = 269 начина

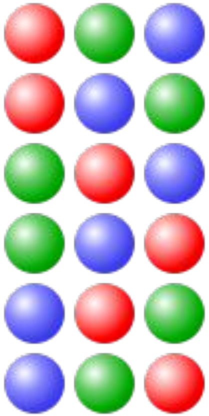
Какво означава подреден и неподреден?



- Има ли значение в какъв ред ще изберем 6 от 49 числа за евентуалната ни печалба от лотарията или не?
    - Не => неподреден
        - редът на участие на елементите не от значение
    - Да => подреден
        - Редът на участие на елементите е от значение



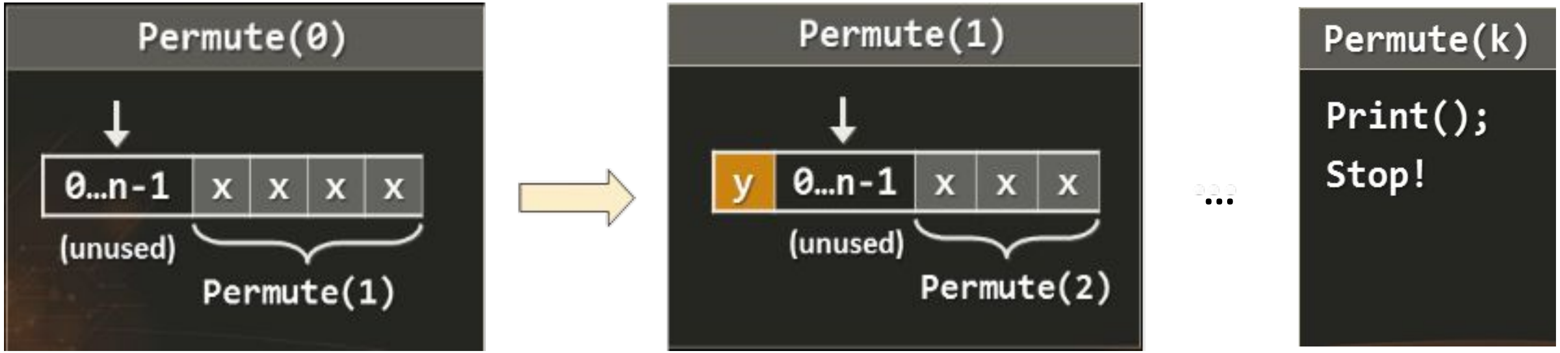
Пермутации



- **Пермутация** - всяко едно разместване на елементите на едно множество
  - Участват всички елементи
  - Редът е от значение
  - Означение: $\mathbf{P_n}$
- Брой на пермутациите:
  - Първият елемент може да бъде избран по **n** начина
  - Вторият елемент може да бъде избран по **n-1** начина
  - Третият елемент може да бъде избран по **n-2** начина и т. н.
- Общо: **n!** = 1 * 2 * … * **n**

Алгоритъм за генериране на пермутации

- Използва се функция с един параметър **Permute**(index)
- Обхождат се елементите на индекси i=0…n-1 и се проверяват дали са използвани
- Маркират се всички използвани елементи
- Извиква се рекурсивно функцията **Permute**(index + 1), за да се генерира останалата част от масива



Генериране на пермутации в Python

- Python разполага с библиотека itertools, която позволява лесно и удобно генериране на пермутации:

```
from itertools import permutations


items = ['red', 'green', 'blue']
shuffles = permutations(items)


for shuffle in shuffles:
    print(shuffle)
```

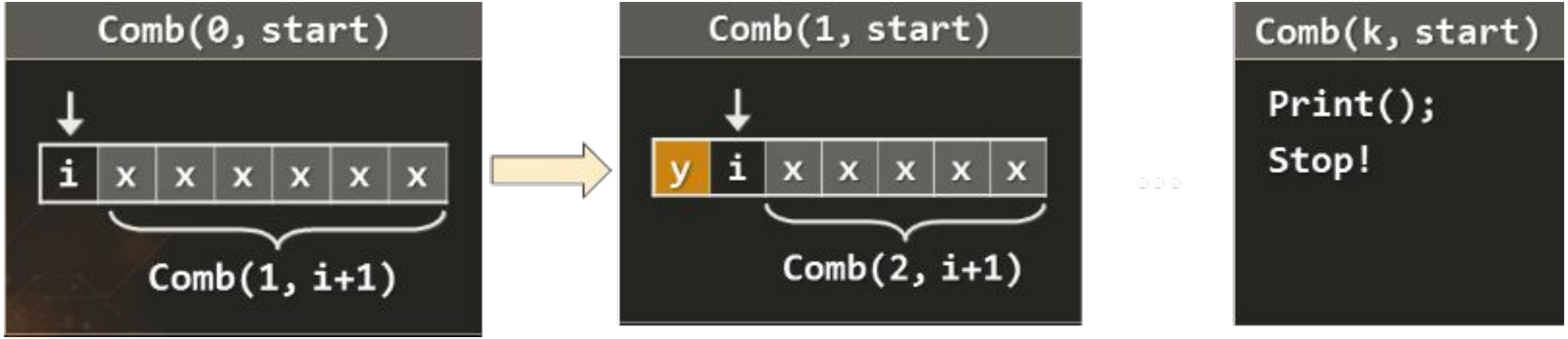
Комбинации

- Комбинация - неподредено подмножество от k елемента избрани от общо n елемента
- Означение: C_n^k
- Брой на комбинациите на **n** елемента от **k**-ти кла $C_n^k = \frac{n!}{(n-k)!k!}$
- Известно и като „**n choose k**” (n избира k, биномен коефициент):

$$\binom{n}{k} = \frac{n!}{(n-k)!k!}$$

Алгоритъм за генериране на комбинации

- Използва се функция с два параметъра comb(index, start)
- Индекса i има за начална стойност start, и крайна n-1
- Извиква се рекурсивно функцията comb(index + 1, i + 1), за да се генерира останалата част от масива



Генериране на комбинации в Python

```
from itertools import combinations

items = ['cian', 'magenta', 'yellow', 'key']
combos = combinations(items, 3)

for combo in combos:
    print(combo)
```

Комбинации от 3 клас от общо 4 елемента

Вариации

- Вариация - подредено подмножество от k елемента избрани от общо n елемента
- Означение: V_n^k
- Брой на вариациите на **n** елемента от **k**-ти клас:

$$V_n^k = n.(n-1).\cdots.(n-k+1) = \frac{n!}{(n-k)!}$$

Биномни коефициенти

- За комбинация на n елемента от k- ти клас използваме означението $\binom{n}{k}$
- Броят на всички различни комбинации на n елемента от k-ти клас съвпада с $\binom{n}{k}$
- Връзка между биономните коефициенти: $\binom{n}{k} = \binom{n}{n-k}$
- Тези числа са известни като биномни коефициенти, тъй като те участват в Нютоновия бином (математическа теорема за разлагане на двучлен, повдигнат на степен).

$$\left(x+y\right)^n = \sum_{k=0}^{n} \binom{n}{k} x^k y^{n-k}$$

Биномни коефициенти

- Някои свойства на биномните коефициенти:

$$C_n^0 = C_n^n = 1 \qquad C_n^1 = C_n^{n-1} = n$$

$$\binom{n}{k} = \binom{n-1}{k} + \binom{n-1}{k-1}$$

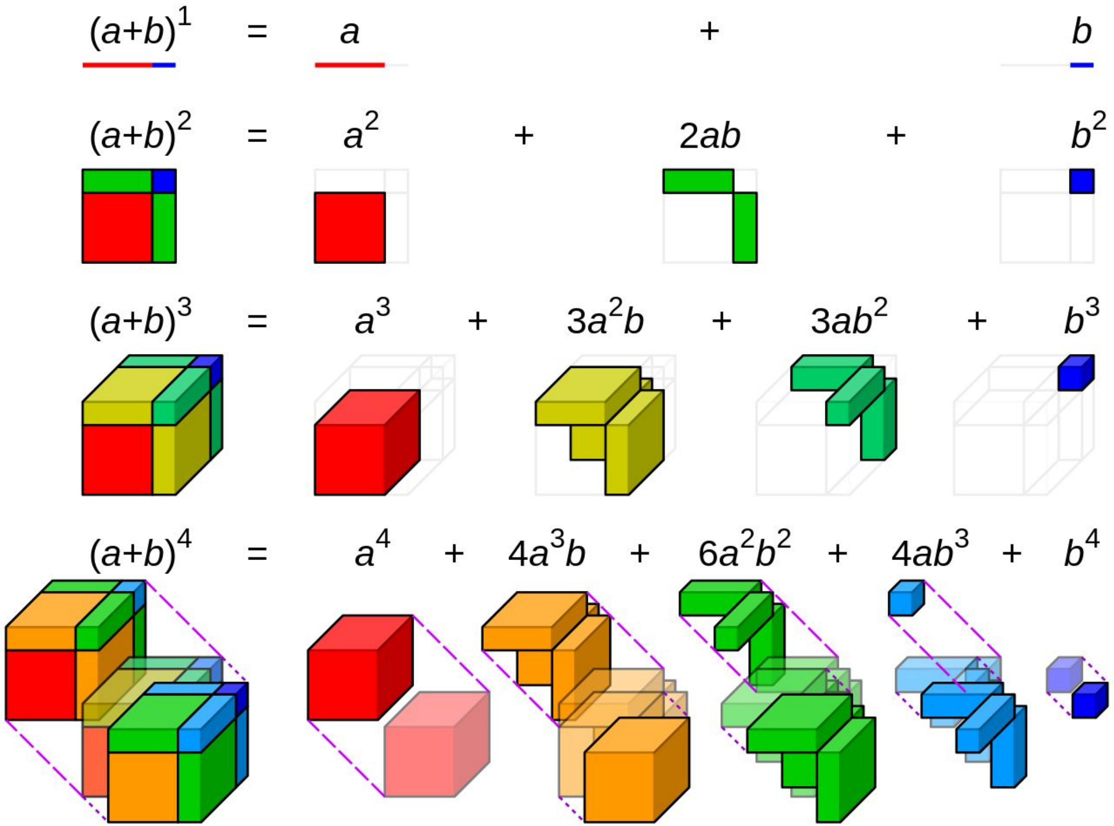
Биномни коефициенти

- Код на Python:

```
import scipy.special
n, k = 10, 5
result = scipy.special.comb(n, k, exact=True)
print(result)
```

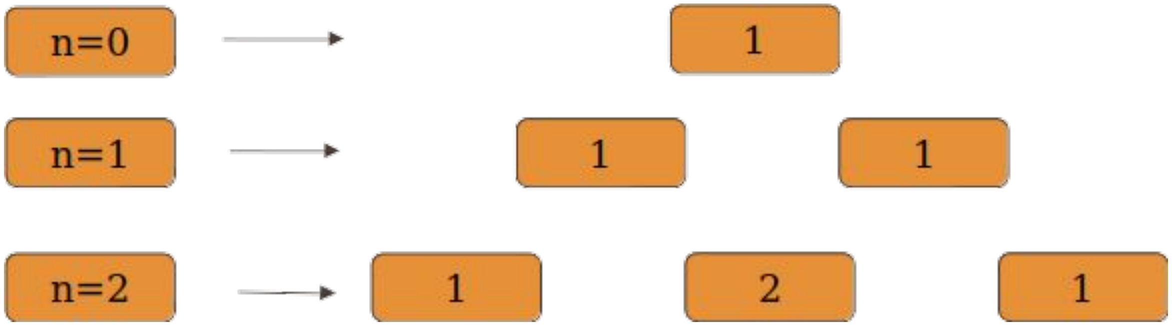
- Важно: Считано от версия 3.8 нататък, math библиотеката в Python също притежава comb функция

Триъгълник на Паскал



Триъгълник на Паскал

- Аритметичен триъгълник, съдържащ биномните коефициенти.
- Позволява да разположите биномните коефициенти, като всяко число е равно на сумата от двете числа над него.



Триъгълник на Паскал

							1							
						1		1						
					1		2		1					
				1		3		3		1				
			1		4		6		4		1			
		1		5		10		10		5		1		
	1		6		15		20		15		6		1	
1		7		21		35		35		21		7		1

Благодаря за вниманието!

Автор: Петър Р. Петров, учител по програмиране, ПГЕЕ „Константин Фотинов”, гр. Бургас