# 22. Algoritmy pro generování permutací a řetězců

## Permutace n prvků

Skupina všech prvků, které jsou uspořádány v jakémkoliv možném pořadí (výběr prvků závisí na pořadí).

Pokud se prvky ve výběru nemohou opakovat, pak počet všech možných výběrů je určen vztahem

Pokud se hovoří o permutacích prvků, jsou tím obvykle myšleny permutace bez opakování.

### Brutal Force

* Prochází již vygenerované číslo
* Pokud najde vygenerované číslo, generuje dál
* + ze začátku rychlé
* – ke konci dost pomalé

### Memory Force

Používá pole boleanů, kde jsou všechny prvky false, pokud je zadán prvek 🡪 true, když se zadá další prvek a už má nastaveno true, vygeneruje další

* + neprohledává pořád dokola celé pole
* – ke konci pomalé

### Dvouprůchodový

* + pořád stejně rychlý 🡪 nejrychlejší
* – 2 průchody
  + Vyplní A–C
  + Prohází prvky (složitější)

## Generování řetězců

* Rekurzivní algoritmus
* Čísla jsou klíčové znaky… lze jen od 0-9
* Středníky jsou klíčové znaky (nejde je rozšiřovat)
* Musí být také platný vstup
* Vnořování je realizováno rekurzí
* 2a; 🡪 aa
* 32ab;c; 🡪 2x ab + 3x c 🡪 ab ab c ab ab c ab ab c

### Průběh generování

* Používají se dva jezdce
  + Pravý se zvyšuje a hledá klíčový znak (0 - 9)
    - Pokud nenajde a je na úrovni nula (není vnořen) 🡪 znak jde do výstupu
    - Pokud najde a není to středník (je to číslo), a je na nulté úrovni tak posune levý jezdec na stejnou pozici jako pravý
* Zvedne úroveň
* Pokud najde středník
  + Sníží úroveň
  + Zkontroluje, jestli není zrovna na úrovni nula
    - Pokud ano tak se použije for s počtem opakování jako hodnota znaku na pozici levého jezdce (bude zákonitě na čísle)
    - Do výstupu potom uloží rekurzivní volání s parametrem substringu od levého jezdce + 1 (tedy hned za číslem) až po pravý jezdec (tedy ke středníku)
* Úroveň se mezi rekurzí nepřenáší! Ta je jen pomocná

### Formatter

Při použití formatter je klíčový znak 3 a některé další znaky.

Například „Test %03d“ je decimální číslo na tři místa a volná místa vyplní nulou. To znamená

#### System.out.format(„Test %03d“, 46);

Vypíše Test 046

#### public static String generate(String s){

#### int level = 0;

#### int l = 0, r = 0;

#### String finalStr = "";

#### for(r = 0; r < s.length(); r++) {

#### Boolean cond1= "0123456789;".contains(String.*valueOf*(s.charAt(r)));

#### if(cond1) {

#### if(s.charAt(r) == ';') {

#### level--;

#### if(level == 0) {

#### int cond = Integer.valueOf(String.valueOf(s.charAt(l))); for(int i = 0; i < cond; i++) {

#### finalStr += generate(s.substring(l + 1, r));

#### }

#### } else {

#### if(level == 0) {

#### l = r;

#### }

#### level++;

#### }

#### } else {

#### if(level == 0) {

#### finalStr += s.charAt(r);

#### }

#### }

#### }

#### finalStr += " ";

#### return finalStr;

#### }