Practicum 3 Sorting & Searching

Substring search en regular expressions

a. Implementatie van substring in programmeertalen

Ga voor Java en minstens nog één andere programmeertaal na hoe het substring algoritme is geïmplementeerd in die taal. Vertel om welke taal het gaat, toon de source code van het algoritme en verklaar om welke variant het gaat (brute force, Knuth-Morris-Pratt, Boyer-Moore, Rabin-Karp of misschien nog wel iets anders).

Let op: de methode waarnaar je op zoek bent hoeft natuurlijk niet substring te heten. In Java heet het bijvoorbeeld indexOf().

b. Vergelijken van algoritmes

In deze deelopgave ga je de algoritmes Knuth-Morris-Pratt en Boyer-Moore uit (Sedgewick & Wayne, 2011) met elkaar vergelijken. De code mag je natuurlijk overnemen uit het boek. De code moet enigszins worden aangepast, want we zijn niet geïnteresseerd waar de substring zich bevindt, maar wel hoe vaak deze voorkomt in de tekst.

Je moet van tien zelf gekozen woorden bepalen hoe vaak ze in de tekst voorkomen. Neem van alle smaken wat: korte/lange woorden die wel/niet in de tekst voorkomen. Per woord tel je hoeveel karaktervergelijkingen het algoritme doet. Uiteindelijk krijg je dus een tabel waarin per woord staat hoe vaak het in de tekst voorkomt en voor beide algoritmes het aantal karaktervergelijkingen geeft. Ga na of je resultaten in lijn zijn met de tabel “cost summary for substring search implementations” in (Sedgewick & Wayne, 2011, p. 779).

Neem als tekst waar in je gaat zoeken het beroemde gedicht “Mei” van Herman Gorter, dat uit 4000 verzen bestaat. Het is te vinden op http://cf.hum.uva.nl/dsp/ljc/gorter/mei.boek1.html, http://cf.hum.uva.nl/dsp/ljc/gorter/mei.boek2.html en http://cf.hum.uva.nl/dsp/ljc/gorter/mei.boek3.html. LET OP: dit is bij elkaar één tekst.

c. Reguliere expressies

Gebruik de java class `java.util.regex.Pattern` om een Java methode

```
boolean checkNummer(String telNr)
```

te maken die controleert of een telefoonnummer voldoet aan de volgende regels.

- Een telefoonnummer heeft 10 cijfers en bestaat uit een netnummer gevolgd door een abonneenummer. Een mobiel nummer heeft 10 cijfers en bestaat uit 06 gevolgd door het abonneenummer.
- Een netnummer is 3 of 4 cijfers en start met een 0. Een netnummer van 4 cijfers heeft nooit een nul als 2e of 3e cijfer.
- Er mag een streepje (-) na het netnummer of de 06.
- Een abonneenummer begint nooit met een 0.
- Uitzonderingen zijn de korte servicenummers 112, 0900xxxx en 0800xxxx nummers. Ook hier mag een streepje (-) na 0800 of 0900.
- Alle nummers zijn ook mogelijk in de varianten met het landnummer (+31 of 0031) ervoor. In dat geval vervalt de startnul van het abonneenummer.
  Bijvoorbeeld: 0205951645 bestaat ook in de variant +31205951645 of 0031205951645.

Maak de regex zo kort mogelijk. Gebruik commentaar en goede naamgeving om de regex helemaal toe te lichten. Maak unittests aan om de methode te testen met diverse testwaarden. Verzin goede testgevallen waarin je zoveel mogelijk randgevallen test. Neem tests die slagen en tests die falen. Beschrijf waarom je juist deze hebt toegevoegd. Toon ook de output van de unit test in je verslag.

Geciteerde werken

Sedgewick, R., & Wayne, K. (2011). *Algorithms.* Pearson Education.