

# Algorithmen und Datenstrukturen in der Bioinformatik

## 5. Programmieraufgabe

Abgabe Do, 19.12. bis 23:59 Uhr per GIT
---

### P1: Q-Gram Index (10 Punkte)

**Programm-Spezifikation** Schreiben Sie ein Programm mit dem Namen `aufgabe5_main.cpp`, das die folgenden Schritte in der `main()` implementiert:

- Einlesen des Genoms (siehe `getline()` Funktion aus Aufgabe 1); das Alphabet ist  $\Sigma = \{A, C, G, T\}$
- Aufbau eines qGram-Indexes über das Genom mittels Counting Sort (siehe `QGramIndex.hpp`).
- Ausgeben der Matches für eine bestimmte Anfrage.

und folgendes Kommandozeileninterface hat:

```
./aufgabe5 <GENOME_FILE> <QUERY>
```

Beispiel:

```
./aufgabe5_main mini.text GATTACA  
GATTACA: 7 0
```

Das  $q$  leitet sich also direkt aus der Query ab (hier:  $q = 7$ ).  
Das Genom-File liegt als plain-text (einzeiliges Textfile) vor.

Implementieren Sie dafür alle Methoden aus der gegebenen Header-Datei `QGramIndex.hpp` in einer `QGramIndex.cpp` und benutzen Sie eine Instanz dieser Klasse in ihrer `main()` Methode, um die Ergebnisse zu generieren. Evtl. nützliche Hilfsfunktionen, welche Sie in `QGramIndex` benutzen sollten, finden Sie in `a5_util.hpp/.cpp`.

Beispieldaten für `<GENOME_FILE>` sind im ZIP file (wenn möglich, checken Sie bitte diese Text/ZIP Dateien aufgrund ihrer Dateigrösse nicht wieder in ihr GIT Verzeichnis ein).

Checken Sie `QGramIndex.hpp`, `QGramIndex.cpp` und `aufgabe5_main.cpp` unter `./aufgabe5/` ins GIT ein.

## Praktikumshinweise

- Im Header *QGramIndex.hpp* sind die Funktionen genau geschrieben, auch evtl. exceptions die geworfen werden sollen.
- Implementieren sie unbedingt counting-sort zum Aufbau des qgram-Index. Andere Implementierungen bringen keine Punkte.
- Benutzen Sie bitmasking in `hashNext()` (kein `pow()` oder `vgl`) und benutzen Sie `hashNext` so oft es geht beim Erstellen ihrer Datenstrukturen.
- Bauen Sie den QGI und SA im Konstruktor einmalig auf. Das erst in `getHits()` zu tun ist grober Unsinn.
- Am Montag zum Tutorium wird eine Test-Klasse online gestellt, mit der Sie ihre Implementierung überprüfen können.

## Zusatzaufgabe: 5 Punkte

Implementieren Sie counting sort mit Hilfe von OpenMP in `QGramIndex`. Die Anzahl der Threads soll direkt von der Kommandozeile als zusätzliches (3.) Argument gelesen werden und in der `main()`-methode mittels `omp_set_num_threads(x)` gesetzt werden (also insbesondere nicht in `QGramIndex`). Evtl. Anforderungen an die Geschwindigkeit finden Sie in der Test-Klasse (Tutorium).