2. Übung

Allgemeine Hinweise

- Verwenden Sie für die Kompilierung g++ mit den Parametern:
  -std=c++17 -Wall -Wextra
- Achten Sie darauf, dass Sie keine Compiler-Warnungen erhalten
- Halten Sie sich an die Best Practices bzgl. Clean Code
- Es ist nicht erlaubt, externe Libraries (außer Standardlibraries) zu verwenden
- Legen Sie für jede der drei Aufgaben einen Unterordner an: u2_1, u2_2 und u2_3
- Entfernen Sie vor dem Hochladen die ausführbaren Kompilate, sowie die Object-Files (*.o)
- Erstellen Sie ein komprimiertes Archiv (.zip oder .tar.gz), das die drei Unterordner enthält und laden Sie dies rechtzeitig auf Moodle
- Spätester Abgabezeitpunkt: **Mi., 22.05.2024, 23:59**

Aufgabe #1 – Best Practices

Wenden Sie Best Practices in Hinblick auf Clean Code an.

Verwenden Sie dazu ein C++ Source File Ihrer Wahl (z.B. eigener Code oder aus einer Online-Quelle) mit ca. 30-50 Quellcodezeilen.
Wenden Sie die Best Practices an und beschreiben Sie die verwendeten Best Practices und deren Nutzen kurz in einem Textfile.

Abgabe:
1. Original-Code
2. Verbesserter Code
3. Report im Textfile oder als Kommentar-Block am Beginn des verbesserten Codes

Hinweis: Der verbesserte Code muss kompilierbar sein und darf keine Warnungen erzeugen ggfs. sind dazu fremde Referenzen zu entfernen oder in den Code mitaufzunehmen; auf den Original-Code muss dies nicht zutreffen!

Aufgabe #2 – ORF-Suche

Erstellen Sie ein Programm, das Open Reading Frames (ORF) in einer Gensequenz sucht.

Eingabe: RNA-Sequenz (String, der nur aus C, G, A und U besteht) als Command-Line-Arguments (argc, argv)
Ausgabe: ORF (inkl. Start- und Stoppcodon) pro Zeile

Als Start-Codon wird AUG verwendet, als Stopp-Codon sind UAG, UGA und UAA möglich.

Hinweise: Zur Vereinfachung können Sie davon ausgehen, dass es keine Überlappungen gibt und dass das Start-Codon nicht innerhalb des ORF erneut vorkommt.
Es muss auch nicht geprüft werden, ob innerhalb des ORF nur Basen-Triplets vorkommen.

Beispiel-Outputs

$ ./ue2_2 CUCAUGCAUACAGUGUAAGCAGGAAGUAUGUAUUUUGUAUAGGCC
AUGCAUACAGUGUAA
AUGUAUUUUGUAUAG

CUC**AUG**CAUACAGUG**UAA**GCAGGAAGU**AUG**UAUUUUGUA**UAG**GCC

Aufgabe #3 – Schnittmenge finden

Erstellen Sie ein Programm, das Wörter aus zwei Dateien einliest und jene Wörter ausgibt, die in beiden Dateien vorkommen.

Die Dateinamen (-pfade) werden als zwei Command-Line-Arguments an das Programm übergeben.

Hinweise: Sie können davon ausgehen, dass die Wörter in Kleinbuchstaben und ohne Satzzeichen in den Dateien stehen.
Die Wörter sind durch Leerzeichen oder Zeilenumbrüche voneinander getrennt.
Mehrfachausgabe des Wortes (bei Mehrfachvorkommen in einer Datei) ist erlaubt.

$ ./ue2_3 input1.txt input2.txt
wort2
worty
wort4
wort5

input1.txt:

wort1 wort2
wort3 worty
wort4 wort5

input2.txt:

wort2 wortx
worty
wort4 wort5