Einleitung

In der Informatik kommt es beim Speichern und Übertragen von Dateien häufig darauf an, die zu speichernden Dateien möglichst klein zu halten. Dazu verwenden Nutzer Programme um die Redundanz (Daten die keine Information enthalten) der enthaltenen Daten zu reduzieren und diese Dateien zu komprimieren (zip, gzip, rar...)

Der Runlength-Algorithmus

Ein recht einfaches jedoch häufig wirkungsvolles Verfahren zur Reduzierung der Dateigröße stellt der sogenannte Runlength-Algorithmus dar. Hierbei wird geprüft, ob Zeichen mehrfach hintereinander vorkommen. Diese werden dann durch ein "Ausscheidungszeichen", einen Zähler und das entsprechende Zeichen ersetzt. Das Ausscheidungszeichen sollte dabei so gewählt werden, dass es möglichst selten im Original-Text vorkommt.

Algorithmus 1 (RLed2):

Es werden nur ASCII-Zeichen (im Bereich 0-127/0x0-0x7F) verwendet. Der Zahlenbereich 0x80-0xFF wird für den Zähler verwendet. Sich wiederholende Zeichen werden gespeichert als Zähler und Zeichen. Der Zähler der Wiederholungen muss somit um 0x80 erhöht werden. Fünf Wiederholungen werden gespeichert als 0x80 + 5 = 0x85. Auf diese Weise können maximal 128 Wiederholungen mit einem Zähler gespeichert werden. Wir vereinbaren, das **jede** Wiederholung mittels Zähler erfasst wird. Auch lediglich zwei Wiederholungen. z:B: 0x41 0x41 wird zu 0x82 0x41

Übungsaufgabe: Wie wird die Textdatei mit dem folgenden Inhalt komprimiert?

AAAAAAAAAABBBBAABBABAAAAA

Algorithmus 2 (RLed3)::

Sich wiederholende Zeichen werden gespeichert als 0x90 + Zähler und Zeichen. Die Zahl 0x90 ist also beim späteren Expandieren das Ausscheidungskriterium zwischen komprimierten und unkomprimierten Daten.

Sollte tatsächlich das Zeichen 0x90 im Quelltext vorkommen, so wird dies durch das Zeichen 0x90 mit einer folgenden 0 im Zieltext gespeichert. Für die Speicherung der 0x90 benötigt man also in der kodierten Datei ein Byte mehr. Dadurch sind Missverständnisse ausgeschlossen. 0x90 + 0 werden beim Dekodieren also wieder zu 0x90. Alle anderen Kombinationen mit 0x90 stellen Zähler dar. Wir vereinbaren, das auch das Ausscheidungszeichen 0x90 mittels Zähler "komprimiert" werden kann.

0x90 → 0x90 0x0

0x90 0x90 0x90 0x90 → 0x90 0x04 0x90

Alle Zeichen werden erst ab einem Vorkommen von drei Zeichen komprimiert.

0x42 0x42 → 0x42 0x42 aber 0x42 0x42 0x42 → 0x90 0x03 0x42

Übungsaufgabe:Wie wird die Datei mit dem folgenden Inhalt komprimiert?

0x41 0x41 0x41 0x41 0x41 0x41 0x41 0x41 0x41 0x41 0x41 0x41 0x90 0x90 0x42 0x42

Aufgabe

Erstellen Sie ein Programm zur Reduzierung der Größe von Dateien, welches nach dem Prinzip des sogenannten „Runlength-Encoders“ funktioniert.

Das Programm kann optional Parameter aufnehmen.
Der Aufruf des Programms erfolgt mit mehreren Argumenten:

```
-c / --compress  → komprimieren
-d / --uncompress → expandieren
-i / --inputfile=<inputfile>
-o / --outputfile=<outputfile>
[-t / --type + 2 bzw 3 /  -t2 oder --type3] → Typ des
                                              Algorithmus
-e / --extension]=<extension>]
[-q / --quiet]
```

Einer der ersten beiden Parameter dient dazu festzulegen, ob die Datei komprimiert (-c/--compress) oder expandiert (--uncompress) werden soll.

Der zweite und dritte Parameter –i/-o geben den Namen und den Speicherort der Dateien an.

Der vierte Parameter legt den Kompressionsalgorithmus fest. Entweder Variante RLed2 oder RLed3. Ohne Angabe des Schalters -t/--type wird RLed3 verwendet.

Der fünfte Parameter (-e/--extension) ist optional. Mit ihm legt man die Namenserweiterung der komprimierten Datei fest. Standardmäßig ist diese "rld". Der Name der Datei (ohne extension) bleibt erhalten.

Der letzte optionale Parameter (-q/--quiet) steht für den stillen Modus. In diesem Modus werden keinerlei Bildschirmausgaben erzeugt, weder Hilfetexte noch Meldungen über den Erfolg oder Misserfolg der Umwandlung.

Beispielaufruf: `myRunlength --compress -i="test.bmp" -o="./test/test" -e=rl3 -t3`

Um die erstellten Dateien mit dem eigenen Datenformat zu kennzeichnen und später wieder erkennen zu können, werden wir

- die Bytes 1-3 der neuen Datei für die eigene Kennung "rl2" bzw. „rl3“ nutzen
- das Byte 4 nutzen, um die Anzahl der Buchstaben der Extension zu speichern
- die Byte 5-x verwenden, um die Extension der Datei zu speichern (z.B. "bmp".)

Ihr Programm soll durch einen entsprechenden Rückgabewert darüber informieren, ob der Programmaufruf erfolgreich war. Dabei gelten folgende Vereinbarungen:

Rückgabewert	Meldung/Bedeutung
0	compress/expand successful
1	wrong usage of parameters
2	file error: source file
3	file error: destination file
4	wrong file format
5	other error

Zu den Klassen und Interfaces die erstellt werden müssen, werden wir Vereinbarungen treffen. Dazu wird Ihnen ein Interface IRLed zur Verfügung gestellt. Dieses Interface ist durch die Klassen RLed2 und RLed3 zu implementieren. Runlength-Exceptions werden durch eine speziell dafür angelegte Exception Klasse mit dem Namen RLedException behandelt.

`IRLed` : RLed Interface

`RLed2`: Implementation des Rled-Interfaces mit dem RLed2 Algorithmus

`RLed3`: Implementation des Rled-Interfaces mit dem RLed3 Algorithmus

`RLedException`: Runlength Exception Klasse

Um die korrekte **Funktion ihres Programms zu überprüfen**, sind Tests erforderlich, die sowohl die richtige Nutzung der Parameter als auch das korrekte Komprimieren / Expandieren überprüfen.

Ein Minimum an erforderlichen Tests ist in der folgenden Liste dargestellt:

Durchzuführende Tests:

- Falsche Anzahl der Parameter übergeben
- Falsche Parameter übergeben
- Fehlende Parameter übergeben
- Korrekte aber vertauschte Parameter übergeben
- Rückgabewerte und Quite-Modus kontrollieren
- Die Datei zum Lesen lässt sich nicht öffnen
- Die Datei zum Lesen hat das falsche Dateiformat
- Eine größere Datei erfolgreich komprimieren und expandieren
- Eine kleinere Datei erfolgreich komprimieren und expandieren
- Eine Datei komprimieren und expandieren die ein 0x90 Zeichen enthält
- Eine Datei komprimieren und expandieren die mehrere 0x90 Zeichen hintereinander enthält
- Eine Datei komprimieren und expandieren die die zu komprimierenden Zeichen am Dateianfang enthält
- Eine Datei komprimieren und expandieren die die zu komprimierenden Zeichen in der Mitte enthält
- Eine Datei komprimieren und expandieren die die zu komprimierenden Zeichen am Dateiende enthält
- Eine Datei komprimieren und expandieren die mehr als 255 zu komprimierende Zeichen hintereinander enthält

Viel Erfolg!