Aufgabenblatt 6 Ausnahmebehandlung und Standardbibliothek

Aufgabe 1

Durchführung von Experimenten zum Studium der Ausnahmebehandlung in C++

a) Schreiben Sie ein Programm mit einer Funktion werfeAusnahme(int nr), die Ausnahmen von verschiedenen Typen werfen soll. Eine Ausnahme soll von einem in der Sprache C++ implementieren Typ sein, eine von einem in der Standardbibliothek definierten, von exception abgeleiteten Ausnahmetyp und eine von einem beliebigen, selbst definierten Typ. Die Funktion muss also mindestens drei verschiedene Ausnahmen werfen. Mit dem Argument der Funktion geben Sie die zu werfende Ausnahme an. Demonstrieren Sie das Abfangen der Ausnahmen durch eine geeignete main().
b) Experimentieren Sie mit der Division durch 0 bei Ganzzahlen. Wird eine Ausnahme geworfen? Wenn ja, welche Ausnahme wird geworfen? Wenn nein, warum glauben Sie wird keine Ausnahme geworfen?
c) Der Logarithmus ist im Reellen nur für positive Argumente definiert. Verwenden sie die log() Funktion, die den natürlichen Logarithmus berechnet. Sie brauchen dafür die Headerdatei <cmath>. Wie wird der Fall behandelt, dass das übergebene Argument null oder negativ ist? Probieren Sie es aus und lesen Sie die Dokumentation. Entwickeln Sie eine eigene Ausnahmebehandlung. Entwickeln Sie dazu eine Wrapper Funktion um die log() Funktion, die bei positivem Argument log() aufruft und bei Argumentwert 0 oder negativ eine entsprechende Ausnahme wirft. Leiten Sie die Ausnahme von einer geeigneten Ausnahmeklasse der Standardbibliothek ab.
d) Schreiben Sie eine Funktion index0bis5(), die eine Ganzzahl als Parameter entgegen nimmt. Ist die Zahl außerhalb des Bereichs [0-5] soll die Funktion eine Ausnahme vom Typ Index0bis5Error werfen. Leiten Sie die Ausnahme Index5Error von der Ausnahme std::out_of_range ab. Überladen Sie what() so, dass what() eine geeignete Erklärung liefert.
e) Erzeugen Sie bei der Verwendung von std::vector eine Ausnahme vom Typ std::length_error und fangen diese ab.

Jeder Ausnahmefall muss vorgeführt werden. Dokumentieren Sie schriftlich Ihre Versuche.

Aufgabe 2

Schreiben Sie ein Programm, das Daten einliest, sie transformiert und dann ausgibt. Das Programm liest aus einer Datei und schreibt in eine (andere) Datei. Der erste Kommandozeilenparameter ist ein Flag (z.B.: -H) mit dem die Transformation angegeben wird. Dieser Parameter muss vorhanden sein. Ist ein zweiter Kommandozeilenparameter vorhanden, wird dieser Parameter als Name der einzulesenden Datei genommen. Ist noch ein dritter Kommandozeilenparameter vorhanden, wird in eine Datei mit diesem Namen geschrieben. Wählen Sie voreingestellte Namen für die Ein- und Ausgabedatei. Diese werden genommen, wenn auf der Kommandozeile kein Name angegeben wurde. Programmieren Sie Ihre Anwendung robust, das heißt prüfen Sie die Kommandozeilenparameter und behandeln Sie mögliche Ausnahmen. Beispiele sind:

- Das Flag fehlt, d.h. es ist kein erster Kommandozeilenparameter vorhanden oder der angegebene Kommandozeilenparameter existiert nicht.
- Die Datei, aus der gelesen wird existiert nicht.

Hinweis: Verwenden Sie ifstream oder fstream für die Eingabe und ofstream oder fstream für die Ausgabe.

Die Eingabedaten sind Text. Transformationen, die Sie implementieren müssen sind:

a) Der Text enthält Leerzeichen. Gelesen wird zeilenweise. Mehrfache, aufeinander folgende Leerzeichen werden zu einem Leerzeichen reduziert. Es wird gelesen bis zur End-of-File Bedingung. Ausgabe ist Text. (Flag –L)
b) Gelesen wird zeilenweise, d.h. der Filter ist zeilenorientiert. Jede gelesene Zeile wird mit einer Zeilennummer ausgegeben, die Nummer steht am Anfang der Zeile, die Nummerierung beginnt mit 1. Es wird gelesen bis zur End-of-File Bedingung. Ausgabe ist Text. (Flag: -Z)
c) Es handelt sich um Text, der im ASCII Code kodiert ist. Gelesen wird zeichenweise bis zur End-of-File Bedingung. Jedes Zeichen wird hexadezimal kodiert ausgegeben. Beispiel: das Zeichen A (ASCII Code 65) wird 41 ausgegeben. (Flag: -H)

Hinweise: Nutzen Sie die Standardbibliothek. Bevor Sie etwas selbst implementieren, schauen Sie, ob Sie eine geeignete Funktionalität in der Standardbibliothek finden. Finden Sie z.B. eine geeignete Unterstützung für die Ausgabe in hexadezimaler Darstellung. Sie brauchen das Rad nicht neu zu erfinden. Prüfen Sie geeignet auf die End-of-File Bedingung.

Testen Sie die Funktionalität. Dazu brauchen Sie geeignete Testfälle. Führen Sie Ihre Tests mit geeigneten Dateien durch. Dokumentieren Sie Ihre Tests **schriftlich**.

Aufgabe 3

Erarbeiten Sie für die Klasse map kleine, einfache, ausführbare Beispiele. Die Typen für Schlüssel und Element sollen Sie selbst als Klassen definieren. Erarbeiten Sie Beispiele für die folgende Funktionalität:

a) Erzeugen eines leeren Objekts vom Typ map.
b) Einfügen von Elementen
c) Kopieren einer map mit ihrem Inhalt
d) Suchen eines Elementes anhand des Schlüssels
e) Herausnehmen eines Elementes, d.h. das Element wird aus dem Behälter entfernt
f) Verwenden Sie einen Iterator, um durch alle Elemente der map zu gehen und demonstrieren Sie die Funktionsweise auf eine geeignete Art.

Die Programme sind in Felix abzugeben. Dokumentierten Sie die C++-Texte, z.B mit Kommentaren. Der Abgabetermin ist im Felix Kurs hinterlegt.