Übungsblatt Nr. 13: in der Übungsstunde

Übungsblatt Nr. 13 Hausübung: Die folgenden Aufgaben sind Pflicht und zählen 1 Punkt!

1) a) Schreiben Sie eine Funktion `void test(int x)`, die auf `cout` ausgibt, ob `x` gerade oder ungerade ist.
   b) Schreiben Sie eine Funktion `void test(double x)`, die auf `cout` ausgibt, ob `x` positiv, negativ oder Null ist.
   c) Rufen Sie ihr Overloadset `test()` in `main` der Reihe nach mit `1, 1.0, '1', 1.0f, 1L, ¨eins¨` auf. Überlegen Sie vor dem Kompilieren, welche der 2 Funktionen diese Aufrufe tatsächlich verwenden und welche fehlerhaft sind.
   d) Kommentieren Sie alle fehlerhaften Aufrufe aus und testen Sie das Programm.
   e) Lesen Sie nun in einer Endlosschleife C++-Strings ein, solange das erfolgreich ist. Sie können daher die Schleife mit dem EndOfFile Zeichen beenden (STRG-Z unter Windows, STRG-D unter Linux). Wenn der String genau einen Punkt enthält (Algorithmus!), konvertieren Sie ihn in einen `double`, andernfalls in einen `int` und rufen jeweils `test()` auf.

2) a) Erstellen Sie den Datentyp `Konto`, der einen `int anfangs_saldo` und einen `std::vector<int> buchungen` enthält.
   b) Schreiben Sie eine konstante Methode `saldo()`, die den aktuellen Kontostand berechnet und zurückgibt (Anfangssaldo + Buchungen).
   c) Schreiben Sie die Methode `void operator+=(int i)`, die den Betrag `i` zu den Buchungen hinzufügt.
   d) Schreiben Sie eine Stream-Ausgabe als `friend`, die den Anfangssaldo, den aktuellen Kontostand und alle Buchungen aufsteigend sortiert ausgibt. Da das Konto dadurch nicht geändert werden soll, müssen Sie den Buchungsvektor kopieren, die Kopie sortieren und ausgeben!
   e) Erzeugen Sie in `main()` ein Konto mit Anfangssaldo 3. Wenn Sie keinen Konto-Konstruktor geschrieben haben, erzeugt `Konto k{3};` eine Warnung, die Sie mit `Konto k{3, {}};` umgehen können (alternativ können Sie auch einen Konstruktor schreiben!). Buchen Sie die Beträge `20, -3, 19` auf das Konto und geben Sie es danach aus.

3) a) Nehmen Sie die `Bruch`-Headerdatei der Übung 12, die eine Exception bei einem Bruch mit Nenner 0 wirft. Lesen Sie von `cin` so lange Brüche ein, wie das Einlesen erfolgreich ist oder bis die Streameingabe eine Exception wirft. Fangen Sie diese Exception auf und geben Sie `what()` der Exception aus.

b) Berechnen Sie in einer Schleife für $i \in \{0 \dots 25\}$ den Bruch-Ausdruck

$$x = (\frac{i}{i+1})/(\frac{i^2 - 40i + 391}{i^2 + 1})$$

in der Form

```
i = 13: x = ...
```

aus. Falls die Rechnung eine Exception wirft, geben Sie statt dessen das `what()` der Exception aus.

4) a) Programmieren Sie eine Funktion `f` mit 2 Integer-Argumenten `a, b`, die das Resultat $a^2 + 1 \mod 3b + 1$ zurückgibt.

b) Programmieren Sie die Funktion
`void write(const string& filename, int from, int to, int cols = 4)`.
Diese Funktion schreibt die Zahlen $f(n, n)$ für $n \in [\text{from}, \text{to})$ in die Datei, wobei in jeder Zeile genau `cols` Werte mit einem Tabstop getrennt stehen sollen.

c) Programmieren Sie die Funktion `void read(const string& filename)`, die alle Integer aus dieser Datei ausliest, und deren Anzahl, deren Summe und deren Mittelwert (als `double`) auf `cout` ausgibt.

d) Das Hauptprogramm soll folgende Aufrufe machen:

```
write(string{"Daten_1.txt"}, 0, 1000, 1)
read(string{"Daten_1.txt"})
write(string{"Daten_2.txt"}, 0, 2000, 2)
read(string{"Daten_2.txt"})
...
write(string{"Daten_6.txt"}, 0, 6000, 6)
read(string{"Daten_6.txt"})
```

Versuchen Sie, diese Aufrufe durch eine Schleife zu erzeugen.

e) Optional: Werfen Sie einen Runtime-Error, wenn die Dateien nicht geschrieben oder nicht gelesen werden können.

Extra-Aufgabe(n): Diese Aufgaben sind freiwillig und zählen 2 Punkte!

5) Erstellen Sie ein Polygon mit L-Form, das 8 Einheiten hoch und 6 Einheiten breit ist und dessen Balken 1 Einheit dick sind (siehe Übung 11, Aufgabe 4) als `vector` von Eckpunkten. und geben Sie es aus. Drehen Sie es um den Punkt (4,4) um 90 Grad im Uhrzeigersinn und geben Sie es erneut aus. Achtung, im Original fehlt noch der `operator-()`, der den negativen Vektor berechnet. Diese Funktion müssen Sie noch programmieren.