FH-OÖ Hagenberg/HSD
Software Entwurf 2, SS 2013
Übung 10

Name: Abgabetermin: KW 25

Mat.Nr: Punkte:

Übungsgruppe: korrigiert:

Aufwand in h:

Beispiel 1 (10 Punkte) Funktoren: Implementieren Sie einen Funktor `LetterFrequency`, der die Anzahl der Vorkommen eines bestimmten Buchstaben in einem String bestimmt, unabhängig von Klein- oder Großschreibung. Verwenden Sie zur Implementierung den STL-Algorithmus `count_if` in Kombination mit einem selbstgeschriebenen STL-Funktor, der zwei Zeichen ohne Berücksichtigung von Groß- und Kleinschreibung vergleicht (mittels einer der beiden Funktionen `tolower` oder `toupper` aus dem Header `<locale>`). Geben Sie mit Hilfe des Funktors `LetterFrequency` die Häufigkeit eines vom Benutzer eingegebenen Buchstabens für alle Elemente eines mit Strings befüllten Vektors aus.

Unter Verwendung von `LetterFrequency` implementieren Sie einen weiteren Funktor `CompareByLetterFrequency`, der zwei Strings nach der Anzahl der Vorkommen eines bestimmten Buchstabens vergleicht. Sortieren Sie mit Hilfe dieses Funktors den String-Vektor nach der Häufigkeit des eingegebenen Buchstabens.

Beispiel:

Strings: one two three four five
Buchstabe: e
Häufigkeiten: 1 0 2 0 1
Sortiert nach Häufigkeiten: two four one five three

Beispiel 2 (7 Punkte) Erzeugung von Zahlenfolgen: Implementieren Sie einen Funktor in Form einer generischen Klasse `SequenceGenerator`, der in Verbindung mit den STL-Algorithmen `generate` oder `generate_n` zur Erzeugung einer Folge von Zahlen verwendet werden kann. Der Funktor soll die Angabe eines Startwertes und einer optionalen Schrittweite (Standardwert 1) erlauben.

Nehmen Sie folgende Anweisungen in Ihr Testprogramm auf:

```
// Numbers from 1 to 10
generate_n(ostream_iterator<int>(cout, " "), 10, SequenceGenerator<int>(1));
cout << endl;
// Numbers from 10 to 1 (descending order)
generate_n(ostream_iterator<int>(cout, " "), 10, SequenceGenerator<int>(10, -1));
cout << endl;
// Numbers from 0 to 5 (increment 0.5)
generate_n(ostream_iterator<double>(cout, " "), 11, SequenceGenerator<double>(0, 0.5));
cout << endl;
// Letters from A to Z
generate_n(ostream_iterator<char>(cout, ""), 26, SequenceGenerator<char>('A'));
cout << endl;
```

Verwenden Sie zusätzlich noch eigene Testfälle, um auch Sonderfälle abzudecken.

Beispiel 3 (7 Punkte) Stringlängen: Gegeben sei ein Vektor von Strings. Zeigen Sie, wie Sie mittels eines einzigen Aufrufs von `std::for_each` und einem geeigneten selbst geschriebenen Funktor die minimale, maximale und mittlere Länge aller Strings ermitteln können.

Allgemeine Hinweise: Legen Sie bei der Erstellung Ihrer Übung großen Wert auf eine **saubere Strukturierung** und auf eine **sorgfältige Ausarbeitung!** Verwenden Sie immer **Module**, um den Testtreiber und die eigentliche Implementierung zu trennen! Dokumentieren Sie alle Schnittstellen und versehen Sie Ihre Algorithmen an entscheidenden Stellen ausführlich mit **Kommentaren! Testen** Sie ihre Implementierungen ausführlich! Geben Sie **Lösungsideen** an!