technische universität dortmund

fakultät für informatik

Elias Kuthe
Roman Kalkreuth
Bernd Zey

Dortmund, den 18. Januar 2020

Praktikum zur Vorlesung Einführung in die Programmierung WS 19/20

Blatt 12

Es können 32 Punkte erreicht werden.

Allgemeine Hinweise

1. Bitte lesen Sie vor der Bearbeitung **alle** Aufgaben sorgfältig durch! Dies erspart Ihnen unnötige Arbeit und somit auch Zeit!
2. Lassen Sie sich fertiggestellte Aufgaben bitte möglichst **frühzeitig** testieren. In der letzten halben Stunde vor Schluss werden bei diesem Blatt nur noch **zwei** Teil-Aufgaben testiert!
3. Wir akzeptieren ein Testat nur, wenn die Lösung eigenständig auf Anhieb erklärt werden kann. Andernfalls müssen wir die entsprechende Teilaufgabe mit 0 Punkten bewerten.

Aufgabe 1: Dateiein-/-ausgabe (32 Punkte)

In dieser Aufgabe geht es darum, eine Klasse `PortableBitmap` für einfache Grafikdateien zu programmieren. Sie soll eine Datei sequenziell einlesen, den Inhalt modifizieren und den modifizierten Inhalt zurück in eine (andere) Datei schreiben können.

HINWEIS: Als Hilfsmittel dürfen Sie die Header zur Dateiein- und -ausgabe `fstream`, Stringverarbeitung `string`.

a) Legen Sie ein neues Projekt `Aufgabe_12` an und fügen Sie die Datei `cellular.cpp` aus dem Downloadbereich hinzu. Diese Datei enthält einen sogenannten zellulären Automaten, der in der `main`-Funktion eine Ausgabe auf der Konsole produziert. Ändern Sie den Programmcode so ab, dass die Ausgabe zusätzlich in einer Textdatei gespeichert wird.

_____ (3)

b) Erstellen Sie nun die Klasse `PortableBitmap` mit folgenden Eigenschaften:

- Private Attribute
  - `unsigned int hoehe`
  - `unsigned int breite`
  - `bool *pixel`[1]

[1] Wer möchte darf hier statt `bool*` auch `std::vector<bool>` verwenden.

- Ein Konstruktor `PortableBitmap(int hoehe, int breite)`, der ein entsprechend dimensioniertes `bool`-Array `pixel` allokiert und mit `false` initialisiert.
- Der Destruktor gibt dies frei.

_____ (5)

c) Fügen Sie der Klasse zwei Methoden `bool getPixel(unsigned int x, unsigned int y) const` und `void setPixel(bool wert, unsigned int x, unsigned int y)` hinzu. Diese dienen zum Auslesen und Setzen von einzelnen Pixeln. Überlegen Sie sich dazu, wie Sie die zweidimensionalen Koordinaten `x` und `y` in den richtigen Index im ein-dimensionalen Array `pixel` umrechnen. Die Koordinaten $(0, 0)$ sollen den Wert an Index `0` adressieren.

_____ (3)

d) Schreiben Sie eine Funktion

```
bool endswith(const std::string &txt, const std::string &suffix)
```

die `true` wird, wenn die Zeichenkette `txt` mit der Zeichenkette `suffix` endet. Testen Sie

- `endswith("woman", "man") == true`
- `endswith("text", "txt") == false`
- `endswith("ab", "bab") == false`

_____ (5)

e) Implementieren Sie eine Methode `void PortableBitmap::save(string const &dateiName)`, die ein `PortableBitmap` als Textdatei im PBM-Format speichert. Das PBM-Dateiformat ist wie folgt definiert:

- Die Datei hat die Endung `.pbm`. Wenn `dateiName` nicht auf `".pbm"` endet soll eine Ausnahme `std::invalid_argument` geschmissen werden mit einem entsprechenden Hinweis.
- Die erste Zeile enthält nur den Text `P1`. Dieser Wert gibt den Dateityp an.
- Die zweite Zeile enthält optional einen mit `#` eingeleiteten Kommentar. Tragen Sie hier
  `# EidP Productions 2019/20`
  ein.
- In der nächsten Zeile stehen zwei durch Leerzeichen getrennte Zahlen, die die Anzahl der Spalten und Zeilen der Bilddaten angeben.
- Der Rest der Datei besteht aus den Bilddaten der Grafik. Dabei enhält jede Zeile so viele durch Leerzeichen getrennte Pixelwerte wie in der dritten Zeile der Datei angegeben.

HINWEIS: Unter `https://de.wikipedia.org/wiki/Portable_Anymap#Bitmap` ist ein Beispiel.

_____ (4)

f) Testen Sie `PortableBitmap::save`, indem sie das Bild des zelluleren Automaten in die Datei `cell.pbm` speichern. Achten Sie darauf, dass alle Ausnahmen ordnungsgemäß gefangen werden.

_____ (4)

g) Implementieren sie eine Funktion `PortableBitmap load(const char *dateiName)`, die eine PBM-Datei einliest und als `PortableBitmap` zurück liefert.

HINWEIS: Zum zeilenweisen einlesen einer Datei könnte `getline` hilfreich sein.

_____ (8)