

Projekt Routenplaner

Traveling Salesman Problem | Sortieralgorithmen

Feix Hauser, Stefan Machhammer, Peter Weiß

FH Technikum Wien

2020-BMR2 B – Gruppe 1

03.06.2020

# Einleitung

# Projektbeschreibung/Aufgabenstellung

Ihr Team wird von einer Speditionsfirma beauftragt, die Routen Ihrer LKW Flotte zu organisieren. Ihrem Team steht eine CSV Datei zur Verfügung, welche folgende Städteinformationen enthält:

* Name der Stadt, Einwohnerzahl , Standort (Längen- & Breitengrad), Ländername & -kürzel, etc.

Der Kunde wünscht sich folgende Punkte von Ihrer Software:

* Die Suche nach Städtenamen soll schnell funktionieren
* Die Städte sollen nach Namen und nach Einwohnerzahl sortiert ausgegeben werden können
* Wenn der Benutzer N Städtenamen eingibt, soll die kürzeste Route ermittelt werden (Traveling Salesman Problem)
* Zusatzaufgabe: Aus logistischen Gründen sollen die Waren erst in die Hauptstädte und von dort in extra Fahrten zu den einzelnen Zielorten gebracht werden.

Beispiel (Graz, Dresden, Frankfurt, Linz):   
Route 1: Zentrale->Berlin->Wien->Zentrale   
Route 2: Wien->Graz->Innsbruck->Linz->Wien   
Route 3: Berlin->Dresden->Frankfurt->Berlin

* Zusätzlich zu den Routen können Sie die Anzahl benötigter Transporter und die gefahrenen Kilometer ausgeben. Auch die Kilometer pro Route wären interessant. Als Zentrale können Sie sich eine Stadt aussuchen und bitte achten Sie darauf, dass Städtenamen auch doppelt vorkommen können.
* Datenstrukturen als auch Algorithmen können von Ihrem Team gewählt werden
* Implementieren Sie in Ihrer Bibliothek zu jedem Anwendungsfall (Suche nach Namen, Sortieren nach Namen & Einwohnerzahl und Berechnung der kürzesten Route) mindestens zwei Algorithmen und vergleichen Sie diese miteinander
  + Best-Case, Average-Case, Worst-Case
  + Big O Notation bzw Landau-Symbole •
* Dokumentieren Sie Ihre Erkenntnisse
* Erstellen Sie Testprogramme, welche die Funktionen Ihrer Library testen
* Argumentieren Sie, warum Sie sich für einen bestimmten Algorithmus entschieden haben
* Zeitmessungen in Ihrer Testapplikation sollten Sinnvoll durchgeführt werden, um die Algorithmen vergleichen zu können
* Es sollte auch möglich sein neue Städte mittels Konsole einzulesen. Diese Städte sollen auch bei einem neuen Programmstart zur Verfügung stehen.
* Die Daten für die kürzeste Route können auf der Konsole ausgegeben werden und müssen zusätzlich in eine CSV Datei gespeichert werden
* Zusatzaufgabe: Eine grafische Ausgabe mittels externer Bibliothek (zB Graphviz) Alle allgemeinen Kriterien zur Bewertung gelten auch für diese Aufgabe

# Ausgangsituation/Startphase

# Ist-Analyse

Folgende Grundfunktion werden für die Software benötigt:   
• Die Suche nach Städtenamen soll schnell funktionieren   
• Die Städte sollen nach Namen und nach Einwohnerzahl sortiert ausgegeben werden können   
• Wenn der Benutzer N Städtenamen eingibt, soll die kürzeste Route ermittelt werden (Traveling Salesman Problem)

# Projektziele, -wünsche und –vorgaben

Mit diesem Projekt sollen folgende Ziele erreicht werden:  
**Grundfunktionen:**

1. Bibliothek zu jedem Anwendungsfall
   1. Einlesen der Daten aus einer CSV Datei
   2. Suche nach Namen
   3. Sortieren nach Namen & Einwohnerzahl
   4. Berechnung der kürzesten Route
2. Erstellen eines Testprogrammes (Grundfunktionen), welches die Funktionen der Library testet
   1. Testen des Einlesens
   2. Testen der Suche
   3. Testen des Ergebnis der Sortierung
   4. Testen der Ausgabe der Berechnung der kürzesten Route
3. Fehler: Wenn ein Fehler auftritt, sollte eine Fehlermeldung erscheinen

**Zusatzfunktion:**

1. Zeitmessungen in der Testapplikation zum Vergleichen der Algorithmen
2. Neue Städte mittels Konsole einlesen
   1. Diese Städte sollen nach einem neuen Programmstart zur Verfügung stehen
3. Ausgabe der Daten für die kürzeste Route auf der Konsole
4. Zusätzlich Speicherung in eine CSV Datei
5. Zusatzaufgaben:
   1. Grafische Ausgabe mittels externer Bibliothek
   2. Waren werden zuerst in die Hauptstädte gebracht und von dort, in extra Fahrten, zu den einzelnen Zielorten
   3. Ermittlung der Kilometer per Route
   4. Anzahl der benötigten Transporter
6. Erweiterung des Testprogramms - Testen der Zusatzfunktionen

# Team

Teamleiter (1. Programmierer): Stefan Machhammer  
2. Programmierer: Peter Weiß  
3. Programmierer: Felix Hauser

# Vorgehensweise/Projektplanungsphase

# Soll-Konzept

# Zeit-, Ablauf- und Terminplanung - Felix

In Tabelle Umwandeln!!!!

Projektantrag: 3h  
Bibliotheken (Σ60h)

* Einlesen der Daten aus einer CSV Datei: Σ10h
  + Einlesen & planen: 3h
  + Programm: 4h
  + Ausarbeiten: 2h
  + In Bibliothek umwandeln 1h
* Suche nach Namen: 10h
  + Einlesen & planen: 3h
  + Programm: 4h
  + Ausarbeiten: 2h
  + In Bibliothek umwandeln 1h
* Sortieren nach Namen & Einwohnerzahl: Σ10h
  + Einlesen & planen: 3h
  + Programm: 4h
  + Ausarbeiten: 2h
  + In Bibliothek umwandeln 1h
* Berechnung der kürzesten Route: Σ10h
  + Einlesen & planen: 3h
  + Programm: 4h
  + Ausarbeiten: 2h
  + In Bibliothek umwandeln 1h
* Einlesen der neuen Städte: Σ10h
  + Einlesen & planen: 3h
  + Programm: 4h
  + Ausarbeiten: 2h
  + In Bibliothek umwandeln Σ10h
* Ausgabe der Daten: Σ10h
  + Einlesen & planen: 3h
  + Programm: 4h
  + Ausarbeiten: 2h
  + In Bibliothek umwandeln 1h

Testdateien: Σ20h

* + Einlesen & planen: 3h
  + Programm: 10h
  + Testen: 3h
  + Ausarbeiten: 4h

Dokumentation: 6h  
Präsentation: 3h

# Zeitplan (Projektantrag)

Excel Datei

# Durchführungsphase

# Diagramm (Peter)

Ablauf Diagram!!

# Tests - Stefan

Durchführen und Ergebnisse Dokumentieren

0-8 Punkte….Bild + Beschreibung

# Einrichtungen (Cmake? DrMemmory? CodeBlocks) - Stefan

Dr memory Ergebniss

Cmake?

# Benutzerhandbuch (Programm ausführen mit Beispielen)

Handbuch für das Ausführen und einbinden Erstellen

Funktion + Beschreibung - Jeder

Einbinden der h-Datei- Peter

Main-Peter

# Ergebnisse/Projektabschlussphase

# Abnahme( Was geht? Wo treten Fehler? Was funktioniert?....)

Was funktioniert? Wo treten Fehler auf? Was wurde aus …(Gründen) nicht geschaft?.....

Leerzeichen,…..

# Soll-Ist-Vergelich => erfüllt/nicht erfüllt – Felix/Stefan

Von Oben übernehmen

# Nachkalkulation => Wie viele Stunden wurden wirklich benötigt – Felix + jeder

# Zusammenfassung/Ende

# Ausblick

# Rückblick

# Glossar

# Verzeichnisse

# Testprotokolle

# Projektplanung-Excel

# Anhang -> Code, Daten-CSV

<https://www.derpade.de/leitfaden-projektdokumentation-fachinformatiker-fuer-anwendungsentwicklungsystemintegration/>