Algorithmen und Datenstrukturen – Labor

AUD-26

**Inhaltsverzeichnis**

1. Programspezifikation
   1. Problemstellung
   2. Festlegung der Dialogführung
   3. Reaktion auf Bedienfehler
2. Testplan
3. Beschreibung des Algorithmus
4. Systementwurf
   1. Datenstruktur
   2. Struktur des Programms
5. **Programmspezifikation** 
   1. Problemstellung

Ein Logistikunternehmen möchte seinen Sortiermethode verbessern in dem sie ein Programm implementieren, so dass die Nutzlast von seinen LKWs optimal ausgenutzt wird. Das Unternehmen verwendet zur Eingabe eine Liste von zufälliger Anzahl an Paketen mit jeweils zugehörende Volumen und Transportpreis.

Die Pakete werden in LKWs gebracht, die bestimmten Ladevolumen besitzen, was Ebenfalls angegeben wird. Diese Ladevolumen soll komplett aufgefüllt werden.

Außerdem gibt eine Priorisierung der Pakete anhand Transportpreis, so dass pro Ladung das höchste Gesamtpreis sinnvoll erzielt werden kann.

* 1. Festlegung der Dialogführung
* „bitte geben Sie den Pfad zu der importierenden Datei an“
* „Ihre Datei wurde erfolgreich importiert“
* „bitte geben Sie das maximale Ladevolumen an“
* „Ihre angegebene Liste wurde anhand Transportpreis und Volumen sortiert“
* „das Optimale Ergebnis ist Folgende: ……“
* „der erzielte Gesamtpreis ist …“
* „Folgende Pakete bleiben stehen: …“
  1. Reaktion auf Bedienfehler
* Falsches Dateiformat eingelesen: „es können nur CSV-Dateien importiert werden, bitte überprüfen Sie ihre Dateiformat“.
* Falsche Formatierung innerhalb des Importierten Datei: „bitte Formatierung des Importierten Datei überprüfen.“.
* Fehler bei der Eingabe des Ladevolumen: „bitte geben Sie nur Zahlen an, die größer als 0 sind“.

1. **Testplan**

**2. 1 Datei wird nicht im „csv.“ Importiert.**

*Rückmeldung an den User:* „Es können ausschließlich Dateien im „csv.“ Format importiert werden. Bitte überprüfen sie ihr Datei – Format.

2.2 **Es wird keine Datei importiert.**

*Rückmeldung an den User:* „Bitte Importieren Sie eine Datei“

2.3 Fehler bei der Eingabe des Zielvolumens .

*Rückmeldung an den User:* „Bitte Überprüfen Sie Ihre Eingabe des Zielvolumens.“.

*Es dürfen nur positive Zahlen verwendet werden*

*2.4 Positiver Testfall :*

Priorisierte Pakete: …..

Gesamtpreis: ……..

Übrige Pakete: …….

# *Beschreibung des verwendeten Algorithmus*

# *inkl. Komplexitätsbetrachtung*

der Algorithmus ,der zur Lösung der Aufgabe verwendet wird ,liest eine CSV-Datei mit Packungsinformationen ein, sortiert die Packungen nach Preis und wählt diejenigen aus, die in ein bestimmtes Ladevolumen passen. Dadurch werden die teuersten Packungen priorisiert und ein maximaler Gesamtpreis erzielt. Die restlichen Packungen, die nicht ins Ladevolumen passen, werden separat aufgelistet.

Laufzeitkomplexität in O(n log n )

# 4.Systementwurf des Programms

# 4.1Datenstrukturen

# Der Algorithmus verwendet eine Liste, um die Packungen aus einer CSV-Datei zu speichern. Eine sortierte Liste wird verwendet, um die Packungen nach ihrem Preis absteigend zu sortieren. Es gibt auch Ergebnislisten, um die priorisierten Packungen und die verbleibenden Packungen zu speichern. Diese Datenstrukturen ermöglichen eine effiziente Speicherung und Verarbeitung der Packungen während des Algorithmus.

* 1. **Struktur des Programms**

# START

# Eingabe der verfügbaren Kapazität (capacity) und CSV-Datei durch Benutzer

# Testen, ob die Eingabe fehlerfrei ist

# WHILE True

# VERSUCHEN

# capacity = float(input("Bitte geben Sie das maximale Ladevolumen ein: "))

# BREAK

# AUSSER ValueError

# DRUCKE "Ungültige Eingabe. Bitte geben Sie eine numerische Zahl ein."

# Testen, ob die Eingabe der CSV-datei fehlerfrei ist

# WHILE True

# VERSUCHEN

# csv\_file = input("Bitte geben Sie den Namen der CSV-Datei ein: ")

# with open (csv\_file, 'r') as file:

# BREAK

# AUSSER FileNotFoundError

# DRUCKE "Datei nicht gefunden. Bitte überprüfen Sie den Namen."

# Folgende Variablen warden initialisiert

# prioritized\_packs = []

# total\_price = 0

# remaining\_packages = []

# ÖFFNE die CSV-Datei (csv\_file) im Lese-Modus ALS file

# LESE den Inhalt der CSV-Datei mit einem CSV-Reader ALS reader

# ÜBERSPRINGE die Spaltennamen mit next(reader)

# FÜR jede Zeile in reader

# pack\_number = Umwandlung von row[0] in Integer

# volume = Umwandlung von row[1] in Float

# price = Umwandlung von row[2] in Float

# rate = price / volume

# FÜGE das Tupel (pack\_number, volume, price, rate) zu packs hinzu

# SORTIERE packs nach dem Wert von rate (absteigend) und speichere das Ergebnis in sorted\_packs

# Folgendes wird difiiniert :

# total\_volume = 0

# FÜR jedes pack in sorted\_packs

# WENN total\_volume + pack[1] <= capacity

# FÜGE pack[0] zu prioritized\_packs hinzu

# total\_volume += pack[1]

# total\_price += pack[2]

# SONST

# FÜGE pack[0] zu remaining\_packages hinzu

# DRUCKE "Priorisierte Pakete:" gefolgt von prioritized\_packs

# DRUCKE "Gesamtpreis:" gefolgt von total\_price und "Euro"

# DRUCKE "Übrige Pakete:" gefolgt von remaining\_packages

# ENDE