|  |  |  |
| --- | --- | --- |
| **Zentralabitur 2024** | **Informatik** | **Material für Prüflinge** |
| **Block 1: Aufgabe A** | **gA** | **Prüfungszeit\*: 250 min** |

**Name:** **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Klasse:** **\_\_\_\_\_\_\_**

Inhaltsverzeichnis

[Aufgabe A (50 BE) 3](#_Toc162335556)

## Aufgabe A (50 BE)

Im Rahmen des Wolfsmonitorings werden für einen Wolf regelmäßig Umweltdaten erfasst. In Abbildung 1 im Material ist eine Klassenkarte der Klasse **Umwelt** dokumentiert, die für das Speichern der Daten verwendet wird.

a) Beschreiben Sie anhand der Klasse **Umwelt** den grundsätzlichen Aufbau einer Klasse.  
**[4 BE]**

Mithilfe der Tabelle in Abbildung 2 kann anhand der aktuellen Position des Wolfes (gegeben durch **xPos** und **yPos**) bestimmt werden, in welcher Art von Gebiet er sich gerade aufhält: Feld, Wald oder Stadt.

Die Operation **gebietBestimmen()** der Klasse **Umwelt** bestimmt anhand der aktuellen Position dieses zugeordnete Gebiet und setzt das Attribut **gebiet** entsprechend als Zeichen **F**, **W** bzw. **S**. Liegt die Position außerhalb der in Abbildung 2 angegebenen Gebiete, wird dem Attribut **gebiet** der Wert **N** zugeordnet.

b) Implementieren Sie in einer im Unterricht verwendeten Programmiersprache die Operation **gebietBestimmen**.  
**[4 BE]**

Die fortlaufend gesammelten Umweltdaten eines Wolfes werden in der global definierten dynamischen Reihung **data** vom Typ **Umwelt** verwaltet.

c) Für die Verwaltung der Umweltdaten stellen das Einfügen und das Sortieren von Objekten aus **data** häufig benutzte Systemanforderungen dar.

Vergleichen Sie die Eignung der Datenstruktur Schlange hinsichtlich dieser Anforderungen mit der Eignung einer dynamischen Reihung.  
**[4 BE]**

d) Die Operation **sucheBevGebiet(): Zeichen** ermittelt, welches der drei Gebiete **F**, **W** oder **S** am häufigsten in den Umweltdaten vorkommt und gibt dieses zurück. Kommen mehrere Gebiete mit größter Häufigkeit vor, wird nur eines zurückgegeben. Sie können davon ausgehen, dass die untersuchte Reihung nicht leer ist. Enthält die Reihung nur Elemente mit dem Wert **N**, wird **X** zurückgegeben.

Entwickeln Sie ein Struktogramm für die Operation **sucheBevGebiet**.  
**[6 BE]**

Die Operation **gSort**, die in Abbildung 3 als Struktogramm dargestellt ist, sortiert die Elemente von **data** absteigend nach ihren Gebieten.

e) Dokumentieren Sie in der Tracetabelle in Abbildung 5 die Arbeitsweise der Operation **gSort** am Beispiel der dynamischen Reihung **data** aus Abbildung 4, indem Sie jeweils die Belegung der dynamischen Reihung an den mit "Protokollierung" im Struktogramm gekennzeichneten Stellen angeben. Für die erste Protokollierung sind die Werte bereits eingetragen  
**[6 BE]**

f) Erläutern Sie die Bedeutung der im Struktogramm grau markierten Anweisung „verringere pos um 1“ für die Funktionsweise der Operation **gSort**.  
**[2 BE]**

g) Beschreiben Sie, wie der Algorithmus **gSort** weiterentwickelt werden kann, so dass die Umweltdaten vorrangig absteigend nach dem Wert des Attributs **gebiet** und zweitrangig aufsteigend dann nach dem Wert des Attributs **luftTemperatur** sortiert werden.  
**[4 BE]**

Um das Jagdverhalten des Wolfes zu analysieren, filtert die Operation **jagdPhasen(datum: Zeichenkette): Stapel vom Typ Umwelt** für einen gegebenen Tag alle Umweltdaten aus **data**, in denen die Herzfrequenz des Wolfes größer als 110 war. Die Operation speichert die ausgewählten Objekte in einem lokal definierten Stapel vom Typ **Umwelt** und gibt diesen zurück, wenn mindestens 5 Elemente enthalten sind. Ansonsten wird ein leerer Stapel zurückgegeben.

h) Beschreiben Sie die Bedeutung von Parametern  
und Rückgabewerten am Beispiel der Operation **jagdPhasen**.  
**[4 BE]**

i) Implementieren Sie in einer im Unterricht verwendeten Programmiersprache die Operation **jagdPhasen**.  
**[6 BE]**

Zur genaueren Analyse der Bewegungsmuster liegen für einen Wolf alle erfassten Werte des Attributs **gebiet** in einer Zeichenkette **route** vor.

#### Beispiel:

**route =  
"FFFWWFFFFFFFFFFFFFSFFSSFFFFWWWFWFWNWWWFFFFSSFFNNF"**

j) Entwickeln Sie eine algorithmische Vorgehensweise in Wortform für eine Operation, mit der eine übergebene, nichtleere Zeichenkette **route** in die in Abbildung 6 gegebene binäre Codierung umgewandelt wird.  
**[4 BE]**

Die durchschnittlichen Häufigkeiten der Zeichen F, W, S und N in Bewegungsmuster-Zeichenketten wie z. B. **route** sind bekannt und in Abbildung 6 angegeben.

k) Untersuchen Sie, wie viel Speicherplatz bei der Verwendung einer Huffman-Codierung der Zeichen F, W, S und N für Zeichenketten der Länge 100 im Vergleich zur in Abbildung 6 gegebenen 2-Bit Codierung durchschnittlich eingespart wird.  
**[6 BE]**

#### Gesamtergebnis

|  |  |  |
| --- | --- | --- |
| **Aufgabe A** | **Mögliche Punkte** | **Erreichte Punkte** |
| **a)** | **4 BE** |  |
| **b)** | **4 BE** |  |
| **c)** | **4 BE** |  |
| **d)** | **6 BE** |  |
| **e)** | **6 BE** |  |
| **f)** | **2 BE** |  |
| **g)** | **4 BE** |  |
| **h)** | **4 BE** |  |
| **i)** | **6 BE** |  |
| **j)** | **4 BE** |  |
| **k)** | **6 BE** |  |