Informatik für Ingenieure

Wintersemester 2022/23 - Übungsklausur 3 Wintersemester 2022/23

Prof. Dr.-Ing. Görschwin Fey

Auswertungsbericht für Matrikelnummer 552864 erstellt am 17.2.2023 um 08:06:50 Uhr.

Übersicht

Aufgabe	Punkte	Ergebnis
Aufgabe 1	2	2.0
Aufgabe 2	2	2.0
Aufgabe 3	5	5.0
Aufgabe 4	5	5.0
Aufgabe 5	5	5.0
Aufgabe 6	5	5.0
Aufgabe 7	5	5.0
Aufgabe 8	5	3.0
Aufgabe 9	5	3.0
Aufgabe 10	6	6.0
Aufgabe 11	5	5.0
Aufgabe 12	25	21.0
Aufgabe 13	25	25.0
Summe	**100**	**92.0**

Ergebnis

Dies ist die Auswertung eines Testats für die Veranstaltung *Informatik für Ingenieure* im *Wintersemester 2022/23*. Die erreichten Punkte werden anteilig als Bonus auf eine bestandene Klausur angerechnet.

Auswertungsbericht für 552864

Aufgabe 1

2.0 / 2 Punkten

Gegeben sei die positive Dualzahl `01101110`.

Sie erhalten für jede richtig übersetzte Zahl Punkte.

Geben Sie die Zahl im Dezimalsystem an: 110 richtige Lösung ist 110 — **0.67 Punkte**

Wandeln Sie die obige Zahl in ihr negatives Pendant gemäß Einer-Komplement um: 10010001 richtige Lösung ist 10010001 — **0.67 Punkte**

Wandeln Sie die obige Zahl in ihr negatives Pendant gemäß Zweier-Komplement um: 10010010 richtige Lösung ist 10010010 — **0.67 Punkte**

Erstellt am 17.2.2023 um 08:06:50 Uhr

Auswertungsbericht für 552864

Aufgabe 2

2.0 / 2 Punkten

Addieren Sie folgende vorzeichenlose, binäre Zahlen: 10000 + 11111. Die resultierende Binärzahl kann so viele Ziffern enthalten wie nötig, um das Ergebnis exakt zu repräsentieren.

Wählen Sie genau eine Antwortmöglichkeit aus. Sie erhalten nur Punkte für diese Aufgabe, wenn Ihre Auswahl der richtigen Antwort entspricht.

- (A) 100101
- (B) 1100100
- (C) 110110
- (D) 11010
- (E) 101111 **2 Punkte**
- (F) 00100

Auswertungsbericht für 552864

Aufgabe 3

5.0 / 5 Punkten

Füllen Sie die Wahrheitstabelle für den folgenden Booleschen Ausdruck aus:

$$y = [(c \vee c) \wedge \neg(a \vee \neg b)] \bigoplus a$$

Es gilt die Notation aus der Vorlesung:

- $\vee := \mathrm{OR}$
- $\wedge := \mathrm{AND}$
- $\neg := \mathrm{NOT}$
- $\bigoplus := \mathrm{XOR}$

Nicht ausgefüllte Zeilen werden mit 0 Punkten bewertet. Falsch ausgefüllte Zeilen geben Punktabzug. Sie erhalten jedoch niemals weniger als 0 Punkte für diese Aufgabe.

a	b	c	y	y (Lösung)	Bewertung
0	0	0	0	0	0.63 Points
0	0	1	0	0	0.63 Points
0	1	0	0	0	0.63 Points
0	1	1	1	1	0.63 Points
1	0	0	1	1	0.63 Points
1	0	1	1	1	0.63 Points
1	1	0	1	1	0.63 Points
1	1	1	1	1	0.63 Points

Erstellt am 17.2.2023 um 08:06:50 Uhr

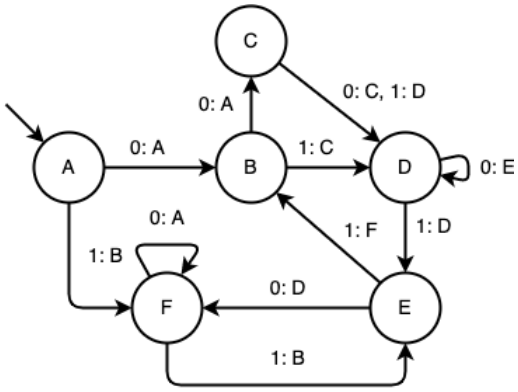
Auswertungsbericht für 552864

Aufgabe 4

5.0 / 5 Punkten

Gegeben seien die folgende Mealy-Maschine und die unten aufgeführen Eingaben. Bitte geben Sie für jede Eingabe die entsprechende Ausgabe der Maschine an.

Nicht oder falsch ausgefüllte Zeilen werden mit 0 Punkten bewertet. Bitte achten Sie bei der Eingabe auf die Groß- und Kleinschreibung.



1. Eingabe 010010 erzeugt Ausgabe : ACEEDD — Korrekte Lösung: ACEEDD --- 1.00 Punkte
2. Eingabe 001000 erzeugt Ausgabe : AADEEE — Korrekte Lösung: AADEEE --- 1.00 Punkte
3. Eingabe 101100 erzeugt Ausgabe : BABFAC — Korrekte Lösung: BABFAC --- 1.00 Punkte
4. Eingabe 001100 erzeugt Ausgabe : AADDDA — Korrekte Lösung: AADDDA --- 1.00 Punkte
5. Eingabe 110100 erzeugt Ausgabe : BBDBDA — Korrekte Lösung: BBDBDA --- 1.00 Punkte

Auswertungsbericht für 552864

Aufgabe 5

5.0 / 5 Punkten

Geben Sie an, ob diese Aussagen über Computerarchitektur korrekt oder nicht korrekt sind.

Wählen Sie für jede Antwortmöglichkeit aus, ob diese richtig oder falsch ist. Für die richige Auswahl erhalten Sie Punkte. Wenn Sie eine falsche Auswahl treffen, dann reduziert dies Ihre Punktzahl. Antwortmöglichkeiten, für die Sie keine Auswahl treffen, werden mit 0 Punkten bewertet.

A ? ✓ ✗ Sowohl der Adress- als auch der Datenbus sind notwendig, um auf den RAM zuzugreifen. **1.25 Punkte**

B ? ✓ ✗ Der Steuerbus wird hauptsächlich für exklusive Buszugriffe verwendet. **1.25 Punkte**

C ? ✓ ✗ Das Bus-System verbindet unterschiedliche Komponenten eines Computers. Daher ist die Breite des Bus-Systems unabhängig von der Prozessorarchitektur. **1.25 Punkte**

D ? ✓ ✗ Je höher das Cache-Level, desto näher ist der Cache am Prozessorkern. **1.25 Punkte**

Auswertungsbericht für 552864

Aufgabe 6

5.0 / 5 Punkten

Welche Aussagen bezüglich Containern der Standard-Library treffen zu?

Wählen Sie für jede Antwortmöglichkeit aus, ob diese richtig oder falsch ist. Für die richige Auswahl erhalten Sie Punkte. Wenn Sie eine falsche Auswahl treffen, dann reduziert dies Ihre Punktzahl. Antwortmöglichkeiten, für die Sie keine Auswahl treffen, werden mit 0 Punkten bewertet.

A ? ✓ ✗ Ein Container der Standard-Library besitzt verschiedene Member-Funktionen, die z.B. die Anzahl enthaltener Elemente zurückgeben oder den Inhalt des Containers löschen können. **1.25 Punkte**

B ? ✓ ✗ Ein `std::array` kann nach der Initialisierung in der Größe verändert werden. **1.25 Punkte**

C ? ✓ ✗ Schlüssel und Werte einer `std::map` müssen vom gleichen Datentyp sein. **1.25 Punkte**

D ? ✓ ✗ Ein `std::vector` kann maximal 10 Elemente enthalten. **1.25 Punkte**

Erstellt am 17.2.2023 um 08:06:50 Uhr

Auswertungsbericht für 552864

Aufgabe 7

5.0 / 5 Punkten

Beim Erzeugen eines Programmes meldet der Compiler einen Syntaxfehler. Was sind mögliche Ursachen?

Wählen Sie für jede Antwortmöglichkeit aus, ob diese richtig oder falsch ist. Für die richige Auswahl erhalten Sie Punkte. Wenn Sie eine falsche Auswahl treffen, dann reduziert dies Ihre Punktzahl. Antwortmöglichkeiten, für die Sie keine Auswahl treffen, werden mit 0 Punkten bewertet.

A ? ✓ ✗ Im C++-Quellcode steht `if (while(a<b))`. **1.25 Punkte**

B ? ✓ ✗ Eine Variable wird benutzt, wurde aber nicht deklariert. **1.25 Punkte**

C ? ✓ ✗ Ein Schlüsselwort ist falsch geschrieben. **1.25 Punkte**

D ? ✓ ✗ Bei einem Zugriff auf ein Feld ist der Index größer als die Größe des Feldes. **1.25 Punkte**

Erstellt am 17.2.2023 um 08:06:50 Uhr

Auswertungsbericht für 552864

Aufgabe 8

3.0 / 5 Punkten

Welche Aussagen bzgl. der dargestellten Funktion sind richtig?

Wählen Sie für jede Antwortmöglichkeit aus, ob diese richtig oder falsch ist. Für die richige Auswahl erhalten Sie Punkte. Wenn Sie eine falsche Auswahl treffen, dann reduziert dies Ihre Punktzahl. Antwortmöglichkeiten, für die Sie keine Auswahl treffen, werden mit 0 Punkten bewertet.

```
int function() {
    int *number = new int(3);

    number = 8;

    return *number;
}
```

A ? ✓ ✗ Der Wert `3` wird dem Zeiger `number` zugewiesen. **-1.00 Punkte**

B ? ✓ ✗ Der Wert `8` überschreibt den gespeicherten Wert `3`. **1.00 Punkte**

C ? ✓ ✗ Der Wert `3`, der für die Initialisierung verwendet wird, wird auf dem Heap gespeichert. **1.00 Punkte**

D ? ✓ ✗ Der Zeiger `number` wird auf dem Heap gespeichert. **1.00 Punkte**

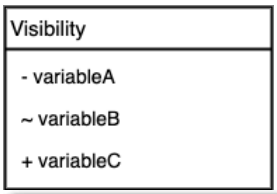
E ? ✓ ✗ Der Speicher, der durch den new-Ausdruck reserviert wird, ist für den restlichen Programmablauf nicht mehr erreichbar. **1.00 Punkte**

Aufgabe 9

3.0 / 5 Punkten

Beantworten Sie die folgenden Fragen unter Berücksichtigung des gegebenen UML-Diagramms.

Wählen Sie für jede Antwortmöglichkeit aus, ob diese richtig oder falsch ist. Für die richige Auswahl erhalten Sie Punkte. Wenn Sie eine falsche Auswahl treffen, dann reduziert dies Ihre Punktzahl. Antwortmöglichkeiten, für die Sie keine Auswahl treffen, werden mit 0 Punkten bewertet.



A ? ✓ ✗ Auf die Klassenvariable `variableB` kann aus der Klasse `Visibility` und aus ihren Unterklassen zugegriffen werden. **1.00 Punkte**

B ? ✓ ✗ Auf die Klassenvariable `variableC` kann von überall zugegriffen werden. **1.00 Punkte**

C ? ✓ ✗ Auf die Klassenvariable `variableA` kann nur aus der Instanz heraus zugegriffen werden. Andere Instanzen der Klasse `Visibility` können nicht auf die Klassenvariable zugreifen. **-1.00 Punkte**

D ? ✓ ✗ Die Klassenvariable `variableA` ist nur vom gleichen Paket (package) aus sichtbar. **1.00 Punkte**

E ? ✓ ✗ Auf die Klassenvariable `variableB` kann nur aus der Klasse `Visibility` zugegriffen werden. **1.00 Punkte**

Auswertungsbericht für 552864

Aufgabe 10

6.0 / 6 Punkten

Welche Aussagen sind beim Löschen eines Datums aus einer doppelt verketteten Liste korrekt?

Wählen Sie für jede Antwortmöglichkeit aus, ob diese richtig oder falsch ist. Für die richige Auswahl erhalten Sie Punkte. Wenn Sie eine falsche Auswahl treffen, dann reduziert dies Ihre Punktzahl. Antwortmöglichkeiten, für die Sie keine Auswahl treffen, werden mit 0 Punkten bewertet.

A ? ✓ ✗ Das Listenelement, das das Datum speichert, muss nicht freigegeben werden. **1.50 Punkte**

B ? ✓ ✗ Es muss das Nachfolgerelement des zu löschenden Datums geändert werden. **1.50 Punkte**

C ? ✓ ✗ Das erste Listenelement kann nicht gelöscht werden, da der Listenanker darauf zeigt. **1.50 Punkte**

D ? ✓ ✗ Es muss das Vorgängerelement des zu löschenden Datums geändert werden. **1.50 Punkte**

Auswertungsbericht für 552864

Aufgabe 11

5.0 / 5 Punkten

Bitte bringen Sie die Schritte der Softwareentwicklung in die korrekte Reihenfolge.

Sie können einige der Bausteine von der rechten Seite auf die linke Seite ziehen und sortieren. Nicht alle Textbausteine müssen benutzt werden. Nutzen Sie nur notwendige Bausteine.

Ablauf der Anforderungsphase

- Start
- Anforderungen
- Entwurf
- Entwicklung
- Test
- Wartung
- Ende

Nicht benutzt

Ihre Antwort	Lösung	Punkte
		5.00
Anforderungen	Anforderungen	0
Entwurf	Entwurf	0
Entwicklung	Entwicklung	0
Test	Test	0
Wartung	Wartung	0
		Total: 5.00

Erstellt am 17.2.2023 um 08:06:50 Uhr

Auswertungsbericht für 552864

Erstellt am 17.2.2023 um 08:06:50 Uhr

Aufgabe 12

21.0 / 25 Punkten

Aufgabenstellung

In der Datei `statistics.hpp` sind Methoden deklariert, die Sie in der Datei `statistics.cpp` implementieren sollen.

Aufgaben

a. `float max(float *data, int size)`

sucht das größte Element des Arrays und gibt den Wert zurück. Ist die Größe des Arrays ungültig, so wirft diese Funktion eine `std::invalid_argument`-Exception.

b. `float min(float *data, int size)`

sucht das kleinste Element des Arrays und gibt den Wert zurück. Ist die Größe des Arrays ungültig, so wirft diese Funktion eine `std::invalid_argument`-Exception.

c. `float average(float *data, int size)`

berechnet den Durchschnitt der gegebenen Werte. Hierfür müssen alle Werte aufaddiert werden und durch die Anzahl der Werte geteilt werden. Ist die Größe des Arrays ungültig, so wirft diese Funktion eine `std::invalid_argument`-Exception.

d. `float median(float *data, int size)`

berechnet den Median der übergebenen Daten. Um den Median zu bestimmen, müssen die Daten erst nach Größe sortiert werden. Der Median ist dann das Element, welches in der Mitte steht. Ist die Größe des Arrays ungültig, so wirft diese Funktion eine `std::invalid_argument`-Exception. Der Algorithmus zur Bestimmung des Medians ist im Folgenden gegeben:

```
Wenn "size" kleiner als 1:
  Wirf die Exception std::invalid_argument.

Lege das Array "copy" mit der Größe "size" an.
Für "i" von 0 bis "size" - 1:
  Weise "copy" an der Stelle "i" den Wert von "data" an der Stelle "i" zu.

Für "i" von "size" - 1 bis 1:
  Für "j" von 0 bis kleiner als "size" - 1:
    Wenn "copy" an der Stelle "j" größer ist als "copy" an der Stelle "j" + 1:
      "tmp" erhält den Wert von "copy" an der Stelle "j".
      "copy" an der Stelle "j" erhält den Wert von "copy" an der Stelle "j" + 1.
      "copy" an der Stelle "j" + 1 erhält den Wert von "tmp".

Gib den Wert von "copy" zurück, der an der Stelle "size" / 2 gespeichert ist.
```

e. `float variance(float *data, int size)`

berechnet die Varianz der gegebenen Daten und wirft bei einem Fehler eine `std::invalid_argument`-Exception. Der Name der Varianz ist s^2 und es ist daher nicht notwendig die Wurzel zu ziehen.
Die Formel ist:

$$s^2(data) = \frac{1}{size} \sum_{i=0}^{size-1} (data_i - average(data))^2$$

Hinweise

- Sie können die Kommentare aus der .hpp auch in die .cpp kopieren, damit sie weniger zwischen den Tabs wechseln müssen.
- Sofern nicht anders angegeben, müssen Sie Parameter nicht auf ihre Gültigkeit prüfen.
- Eine Exception wird durch das Schlüsselwort `throw` erzeugt und es muss dem Konstruktor eine Fehlernachricht übergeben werden: `std::range_error("Hallo Welt, ich bin ein Fehler.")`

Folgende Dateien wurden Ihnen während der Prüfung zur Verfügung gestellt:

main.cpp

```
#include "statistics.hpp"

#include <iostream>
#include <iomanip>

int main() {
    float data[] = { 4.0f, 2.0f, 3.0f, 5.0f, 6.0f, 1.0f };

    // Die folgende Zeile sorgt dafür, dass eine 3,0 mit einer Nachkommastelle ausgegeben wird.
    // The following line makes sure we will print a 3.0 with one decimal.
```

```
    std::cout << std::fixed << std::setprecision(1);

    std::cout << "Das Array wurde mit den Werten [4.0f, 2.0f, 3.0f, 5.0f, 6.0f, 1.0f] initialisiert." << std::endl;
    std::cout << "The Array was initialized using the values [4.0f, 2.0f, 3.0f, 5.0f, 6.0f, 1.0f] initialisiert." << std::endl;

    std::cout << "Das Minimum ist (erwartet 1.0): " << min(data, 6) << std::endl;
    std::cout << "The minimum is (expected 1.0): " << min(data, 6) << std::endl;
    std::cout << std::endl;

    std::cout << "Das Maximum ist (erwartet 6.0): " << max(data, 6) << std::endl;
    std::cout << "The maximum is (expected 6.0): " << max(data, 6) << std::endl;
    std::cout << std::endl;

    std::cout << "Der Durchschnitt ist (erwartet 3.5): " << average(data, 6) << std::endl;
    std::cout << "The average is (expected 3.5): " << average(data, 6) << std::endl;
    std::cout << std::endl;

    std::cout << "Der Median ist (erwartet 4.0): " << median(data, 6) << std::endl;
    std::cout << "The median is (expected 4.0): " << median(data, 6) << std::endl;
    std::cout << std::endl;

    std::cout << "Die Varianz ist (erwartet 2.9): " << variance(data, 6) << std::endl;
    std::cout << "The variance is (expected 2.9): " << variance(data, 6) << std::endl;
    std::cout << std::endl;

    return 0;
}
```

statistics.cpp

```
#include "statistics.hpp"

/*
 * HINWEIS:
 *
 * Details zu den Funktionen gibt es in der Header-Datei. Der zu implementierende
 * Algorithmus der Suche ist in der Aufgabenbeschreibung gegeben.
 *
 * REMARK:
 *
 * The function details are given in the header file and the exercise description.
 *   In the exercise description, you will find the description of the search algorithm
 *   as well.
 */
float max(float *data, int size) {
    return -1;
}

float min(float *data, int size) {
    return -1;
}

float average(float *data, int size) {
    return -1;
}

float median(float *data, int size) {
    return -1;
}

float variance(float *data, int size) {
    return -1;
}
```

statistics.hpp

```
#ifndef STATISTICS_HPP
#define STATISTICS_HPP

#include <stdexcept>

/**
 * @brief Sucht das größte Element im Array und gibt dieses zurück.
 *
 * @param array Das Array, in dem gesucht werden soll.
 * @param size Die Größe des Arrays.
 * @return float Der größte Wert des Arrays.
 * @throws std::invalid_argument if the given size is less than or equal to 0.
 */
/**
 * @brief Searches the largest value in the given array.
 *
 * @param array The array to search in.
 * @param size The size of the array.
 * @return float The largest value.
 * @throws std::invalid_argument if the given size is less than or equal to 0.
 */

float max(float *data, int size);

/**
 * @brief Sucht das kleinste Element im Array und gibt dieses zurück.
 *
 * @param array Das Array, in dem gesucht werden soll.
```

```
 * @param size Die Größe des Arrays.
 * @return float Der kleinsten Wert des Arrays.
 * @throws std::invalid_argument if the given size is less than or equal to 0.
 */
/**
 * @brief Searches the smallest value in the given array.
 *
 * @param array The array to search in.
 * @param size The size of the array.
 * @return float The smallest value.
 * @throws std::invalid_argument if the given size is less than or equal to 0.
 */
float min(float *data, int size);

/**
 * @brief Berechnet den Durchschnitt der übergebenen Daten.
 *
 * @param data Das Datenarray auf dem der Durchschnitt berechnet werden soll.
 * @param size Die Größe des Arrays.
 * @return float Der Durchschnitt der gespeicherten Werte
 * @throws std::invalid_argument Wenn die gegebene Größe kleiner oder gleich 0 ist
 */
/**
 * @brief Calculates the average of the given data.
 *
 * @param data An array of data values.
 * @param size The size of the array.
 * @return float The average value.
 * @throws std::invalid_argument if the given size is less than or equal to 0.
 */
float average(float *data, int size);

/**
 * @brief Berechnet den Median der übergebenen Daten. Für einen detaillierten Algorithmus
 *   schauen Sie bitte in die textuelle Aufgabenbeschreibung.
 *
 * @param data Das Datenarray auf dem der Durchschnitt berechnet werden soll.
 * @param size Die Größe des Arrays.
 * @return float Der Durchschnitt der gespeicherten Werte
 * @throws std::invalid_argument Wenn die gegebene Größe kleiner oder gleich 0 ist
 */
/**
 * @brief Calculates the median of the given data. Please have a look at the
 *  textual task description for the exact algorithm to implement.
 *
 * @param data An array of data values.
 * @param size The size of the array.
 * @return float The average value.
 * @throws std::invalid_argument if the given size is less than or equal to 0.
 */
float median(float *data, int size);

/**
 * @brief Berechnet die Varianz der übergebenen Daten. Die Formel zur Berechnung
 *   finden Sie in der textuellen Aufgabenbeschreibung. Sie können bei der Berechnung die
 *   Funktion <code>average</code> wiederverwenden.
 *
 * @param data Das Datenarray auf dem die Varianz berechnet werden soll.
 * @param size Die Größe des Arrays.
 * @return float Die Standardabweichung der gespeicherten Werte
 * @throws std::invalid_argument Wenn die gegebene Größen kleiner oder gleich 0 ist
 */
/**
 * @brief Calculates the variancen of the given data. Please have a look at the textual task
 *   description for the formula. You are allowed to use the <code>average</code> function
 * you have implemented.
 *
 * @param data An array of data values.
 * @param size The size of the array.
 * @return float The variance of the given data.
 * @throws std::invalid_argument if the given size is less than or equal to 0.
 */
float variance(float *data, int size);

#endif //STATISTICS_HPP
```

Folgende Dateien haben Sie in der Prüfung bearbeitet bzw. erstellt:

main.cpp

```
#include "statistics.hpp"

#include <iostream>
#include <iomanip>

int main() {
    float data[] = { 4.0f, 2.0f, 3.0f, 5.0f, 6.0f, 1.0f };

    // Die folgende Zeile sorgt dafür, dass eine 3,0 mit einer Nachkommastelle ausgegeben wird.
    // The following line makes sure we will print a 3.0 with one decimal.
    std::cout << std::fixed << std::setprecision(1);

    std::cout << "Das Array wurde mit den Werten [4.0f, 2.0f, 3.0f, 5.0f, 6.0f, 1.0f] initialisiert." << std::endl;
    std::cout << "The Array was initialized using the values [4.0f, 2.0f, 3.0f, 5.0f, 6.0f, 1.0f] initialisiert." << std::endl;
```

```
    std::cout << "Das Minimum ist (erwartet 1.0): " << min(data, 6) << std::endl;
    std::cout << "The minimum is (expected 1.0): " << min(data, 6) << std::endl;
    std::cout << std::endl;

    std::cout << "Das Maximum ist (erwartet 6.0): " << max(data, 6) << std::endl;
    std::cout << "The maximum is (expected 6.0): " << max(data, 6) << std::endl;
    std::cout << std::endl;

    std::cout << "Der Durchschnitt ist (erwartet 3.5): " << average(data, 6) << std::endl;
    std::cout << "The average is (expected 3.5): " << average(data, 6) << std::endl;
    std::cout << std::endl;

    std::cout << "Der Median ist (erwartet 4.0): " << median(data, 6) << std::endl;
    std::cout << "The median is (expected 4.0): " << median(data, 6) << std::endl;
    std::cout << std::endl;

    std::cout << "Die Varianz ist (erwartet 2.9): " << variance(data, 6) << std::endl;
    std::cout << "The variance is (expected 2.9): " << variance(data, 6) << std::endl;
    std::cout << std::endl;

    return 0;
}
```

statistics.cpp

```
#include "statistics.hpp"

/*
 * HINWEIS:
 *
 * Details zu den Funktionen gibt es in der Header-Datei. Der zu implementierende
 * Algorithmus der Suche ist in der Aufgabenbeschreibung gegeben.
 *
 * REMARK:
 *
 * The function details are given in the header file and the exercise description.
 *   In the exercise description, you will find the description of the search algorithm
 *   as well.
 */
 /**
 * @brief Sucht das größte Element im Array und gibt dieses zurück.
 *
 * @param array Das Array, in dem gesucht werden soll.
 * @param size Die Größe des Arrays.
 * @return float Der größte Wert des Arrays.
 * @throws std::invalid_argument if the given size is less than or equal to 0.
 */
float max(float *data, int size) {
    if (size <= 0) std::invalid_argument("Size to small");
    float max = data[0];
    for (int i = 0; i < size; i++) {
        if (data[i] > max) max = data[i];
    }
    return max;
}

float min(float *data, int size) {
    if (size <= 0) std::invalid_argument("Size to small");
    float min = data[0];
    for (int i = 0; i < size; i++) {
        if (data[i] < min) min = data[i];
    }
    return min;
}

float average(float *data, int size) {
    if (size <= 0) std::invalid_argument("Size to small");
    double sum = 0;
    for (int i =0; i < size; i++) sum += data[i];
    return sum / size;
}
/*
Wenn "size" kleiner als 1:
  Wirf die Exception std::invalid_argument.

Lege das Array "copy" mit der Größe "size" an.
Für "i" von 0 bis "size" - 1:
  Weise "copy" an der Stelle "i" den Wert von "data" an der Stelle "i" zu.

Für "i" von "size" - 1 bis 1:
  Für "j" von 0 bis kleiner als "size" - 1:
    Wenn "copy" an der Stelle "j" größer ist als "copy" an der Stelle "j" + 1:
      "tmp" erhält den Wert von "copy" an der Stelle "j".
      "copy" an der Stelle "j" erhält den Wert von "copy" an der Stelle "j" + 1.
      "copy" an der Stelle "j" + 1 erhält den Wert von "tmp".

Gib den Wert von "copy" zurück, der an der Stelle "size" / 2 gespeichert ist.
*/
float median(float *data, int size) {
    if (size < 1) throw std::invalid_argument("size to small");
    float * copy = new float[size];
    for (int i = 0; i < size; i++) copy[i] = data[i];
    for (int i = size -1; i > 0; i--) {
        for (int j = 0; j < size -1; j++) {
            if (copy[j] > copy[j+1]) {
                float tmp = copy[j];
```

```
                copy[j] = copy[j+1];
                copy[j+1] = tmp;
            }
        }
    }
    float med = copy[size/2];
    delete copy;
    return med;
}

/**
 * @brief Berechnet die Varianz der übergebenen Daten. Die Formel zur Berechnung
 *   finden Sie in der textuellen Aufgabenbeschreibung. Sie können bei der Berechnung die
 *   Funktion <code>average</code> wiederverwenden.
 *
 * @param data Das Datenarray auf dem die Varianz berechnet werden soll.
 * @param size Die Größe des Arrays.
 * @return float Die Standardabweichung der gespeicherten Werte
 * @throws std::invalid_argument Wenn die gegebene Größen kleiner oder gleich 0 ist
 */
float variance(float *data, int size) {
    if (size < 1) throw std::invalid_argument("size to small");
    float avrg = average(data, size);
    double sum = 0;
    for (int i = 0; i < size; i++) sum += (data[i]-avrg)*(data[i]-avrg);
    return sum / size;
}
```

Bewertung mit Unittests

Testname	Assertions	Bestanden	Punkte
average__array_of_size_one	1	1	1.00
average__array_of_size_ten	1	1	1.00
average__array_of_size_three	1	1	1.00
average__array_out_of_bounds	1	1	1.00
average__negative_size	1	0	0.00
average_zero_size	1	0	0.00
max__max_in_last_slot	1	1	1.00
max__negative_size	1	0	0.00
max__on_array_of_size_1	1	1	1.00
max__on_reverse_sorted_array_of_size_3	1	1	1.00
max__on_shuffled_array_of_size_6	1	1	1.00
max__out_of_bounds_check	1	1	1.00
median__array_of_size_one	1	1	1.00
median__array_of_size_ten	1	1	1.00
median__array_of_size_three	1	1	1.00
median__zero_size	1	1	1.00
min__min_in_last_slot	1	1	1.00
min__on_array_of_size_1	1	1	1.00
min__on_shuffled_array_of_size_10	1	1	1.00
min__on_shuffled_array_of_size_6	1	1	1.00
min__out_of_bounds_check	1	1	1.00
min__size_0	1	0	0.00
variance__array_of_size_ten	1	1	1.00
variance__array_of_size_three	1	1	1.00
variance__zero_size	1	1	1.00

Auswertungsbericht für 552864

Erstellt am 17.2.2023 um 08:06:50 Uhr

Aufgabe 13

25.0 / 25 Punkten

Aufgabenstellung

Ein Präfixbaum (Trie) ist eine assoziative Datenstruktur, die eine Abbildung von Zeichenketten auf Werte speichert. Wie jede Baumstruktur ist der Präfixbaum aus Knoten `trie_node` aufgebaut. Ein Knoten speichert möglicherweise den Wert, der einer Zeichenkette zugeordnet wurde, und hat für jedes Zeichen des ASCII-Zeichencodes einen möglichen Nachfolger. Der Wert ist immer positiv. Die Klassenvariable `data` einer `trie_node` wird immer mit dem Wert `-1` initialisiert, um anzuzeigen, dass dieser Knoten noch keinen Wert speichert. Der Anker der Datenstruktur ist die Klasse `trie`.

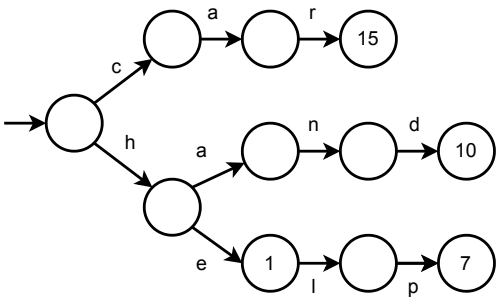
Die Aufgabe stellt eine Implementierung der Knoten (`trie_node.hpp`) und ein Gerüst der Klasse `trie` in den Dateien `trie.cpp` sowie `trie.hpp` bereit. Die Datei `main.cpp` stellt eine beispielhafte Nutzung eines Tries mit dokumentiertem erwartetem Verhalten zur Verfügung. Die Klasse `trie` stellt eine bereits implementierte Funktion `print` zum Ausgeben des Tries bereit.

Bei der Implementierung der Knoten verwenden Sie bitte eine `std::map`, um die Nachfolger eines Knotens in Abhängigkeit des nächstens Zeichens zu speichern. Die notwendigen Funktionen werden am Ende dieser Aufgabenbeschreibung erklärt.

Beispiel

Damit Sie sich die Trie-Datenstruktur leichter vorstellen können, ist hier ein kleiner Code-Abschnitt mit einer grafischen Darstellung des Zustands im Speicher angegeben. In der Darstellung werden die Knoten als Kreise visualisiert (das Ankerelement wird nicht dargestellt). Der Wurzelknoten des Tries wird durch eine eingehende Kante, die keinen Startknoten hat, dargestellt. Die Kanten sind mit den Zeichen beschriftet, durch die sie erreichbar sind und der Knoten ist mit dem zugeordneten Wert beschriftet.

```
trie t;
t.put("car", 15); // Legt für den Schlüssel "car" den Wert 15 in t.
t.put("hand", 10);
t.put("help", 7);
t.put("he", 1);
```



Graphische Darstellung des Ergebnisses.

Aufgaben

Implementieren Sie die folgenden Funktionen - überlegen Sie dabei zunächst, welches die einfachste Funktion ist, und beginnen Sie damit:

- `bool empty() const`

  überprüft, ob der Trie leer ist und gibt in diesem Fall `true` zurück. Ansonsten liefert die Funktion den Wert `false` zurück. Der Baum ist leer, wenn das Ankerelement auf keinen Wurzelknoten zeigt.

- `size_t size() const`

  berechnet rekursiv die Größe des Baums. Ein leerer Baum hat die Größe 0 und für jeden nicht leeren Baum ist die Größe als die Anzahl der Knoten definiert, die einen Wert speichern. Dies ist *nicht* notwendiger Weise die Anzahl der Knoten im Baum. `size_t` ist ein numerischer Datentyp, der nur positive Zahlen oder 0 speichern kann.

- `int put(const std::string& key, const int data)`

  fügt den gegebenen Wert für den Schlüssel im Trie ein. Dafür durchsucht die Funktion den Trie rekursiv bis der Knoten erreicht ist, der dem letzten Zeichen des Schlüssels entspricht. Hier wird der Wert `data` in dem Knoten gespeichert. Sollte dort schon ein Wert vorhanden sein, so wird dieser überschrieben. Wenn einer oder mehrere Knoten fehlen, muss die Funktion diese erstellen. Bei erfolgreichem Einfügen gibt die Funktion `0` zurück, ansonsten den Wert `-1`, falls `key` leer ist.

- `int get(const std::string& key)`

  gibt den Wert zurück, der bereits im Baum für den Schlüssel gespeichert ist. Dafür durchsucht die Funktion den Baum rekursiv nach dem entsprechenden Knoten. Wird der Wert gefunden, so gibt die Funktion das gespeicherte Datum zurück. Ist für den Schlüssel kein Wert gespeichert, so gibt die Funktion `-1` zurück.

Hinweis: Die rekursive Funktion `void print() const` ist in der Datei `trie.hpp` als Beispiel vorgegeben.

Kurzes Tutorial zu std::string

Eine Zeichenfolge kann in C++ dargestellt werden mit einem Feld von `char` Elementen und einem Integer, der die Größe des Feldes angibt. Eine Zeichenfolge kann auch mit der Klasse `std::string` dargestellt werden. Ein `std::string` kann genutzt werden, um Zeichenfolgen zu speichern und zu bearbeiten.

Auf ein Zeichen eines `std::string` kann wie bei einem Feld mit dem Operator `[]` zugegriffen werden und die Länge eines `std::string` kann mit der Funktion `size()` bestimmt werden.

Beispiel

Auswertungsbericht für 552864

Folgende Dateien wurden Ihnen während der Prüfung zur Verfügung gestellt:

main.cpp

```
#include <string>
#include <iostream>

int main()
{
  std::string str = "Hello world!";
    std::cout << str[0] << str[2] << str[6] << std::endl;
    std::cout << "String size: " << str.size();
    return 0;
}
```

Ausgabe:

```
Hlw
String size: 12
```

Kurzes Tutorial zu std::map

Eine `std::map` ist eine assoziative Datenstruktur, die beliebige Schlüssel auf beliebige Werte abbilden kann. Die Klasse `std::map` stellt einen Iterator zur Verfügung, mit dem über die Schlüssel-Wert-Paare iteriert werden kann. Den Iterator erhält man durch den Aufruf der Funktion `map.begin()` und solange dieser nicht `map.end()` entspricht, sind noch Werte in der Map. Über `iterator->first` kann man auf den Schlüssel des Eintrags zugreifen und `iterator->second` speichert den zugehörigen Wert. Durch Aufruf von `iterator++` lässt sich der Iterator auf den nächsten Eintrag weiterschalten.

Außerdem kann man mit der Funktion `find` prüfen, ob ein Schlüssel existiert. Liefert diese Funktion `map.end()` zurück, so gibt es keinen passenden Eintrag für den Schlüssel.

Beispiel

```
#include <map>
#include <iostream>

int main() {
  std::map<int, float> map;

    map[10] = 1.0f;
    map[11] = 2.0f;
  std::cout << std::endl;
  std::cout << std::endl;
    auto iterator = map.begin();
  std::cout << "Bilde 'hand' auf den Wert 10 ab." << std::endl;
    while(iterator != map.end()) {
  std::cout << "Mapping 'hand' to value 10." << std::endl;
  t.put("hand", 10);
      std::cout << "Schlüssel: " << iterator->first << " Wert: " << iterator->second << std::endl;
      iterator++;
    }
  std::cout << "Erwartet/Expected:" << std::endl <<
      R"(└─c
    std::cout << "Direkter Zugriff: " << map[10] << std::endl;
    ├─a
    │  └─r [15]
    std::cout << "Ist der Schlüssel 12 vorhanden? " << (map.find(12) != map.end()) << std::endl;
└─h
    └─a
    return 0;
}       ├─n
        │  └─d [10]
)";
```

Ausgabe:

```
Schlüssel: 10 Wert: 1.0
Schlüssel: 11 Wert: 2.0
Direkter Zugriff: 1.0
Ist der Schlüssel 12 vorhanden? 0
```

```
  std::cout << "Bilde 'help' auf den Wert 7 ab." << std::endl;
  std::cout << "Bilde 'he' auf den Wert 1 ab." << std::endl;
  std::cout << "Mapping 'help' to value 7." << std::endl;
  std::cout << "Mapping 'he' to value 1." << std::endl;
  t.put("help", 7);
  t.put("he", 1);

  std::cout << "Erwartet/Expected:" << std::endl <<
      R"(└─c
    ├─a
    │  └─r [15]
└─h
    └─a
        ├─n
        │  └─d [10]
    └─e [1]
        └─l
            └─p [7]

)";
  std::cout << std::endl << "Aktueller Zustand:" << std::endl << "Current status:" << std::endl;
  t.print();
  std::cout << std::endl;
  std::cout << std::endl;

  std::cout << "Prüfe die Größe des trie (erwartet 4): " << t.size() << std::endl;
  std::cout << "Checking size of trie (expected 4): " << t.size() << std::endl;

  std::cout << std::endl;
  std::cout << std::endl;

  std::cout << "Suche nach Einträgen:" << std::endl;
  std::cout << "Searching for entries:" << std::endl;
```

```
    int value = t.get("help");
    std::cout << "Suche nach 'help' (erwartet 7): " <<  value << std::endl;
    std::cout << "Searching for 'help' (expected 7): " << value << std::endl;

    std::cout << std::endl;

    value = t.get("car");
    std::cout << "Suche nach 'car' (erwartet 15): " << value << std::endl;
    std::cout << "Searching for 'car' (expected 15): " << value << std::endl;

    return 0;
  } catch(std::logic_error e) {
    std::cerr << "Ein Fehler ist bei der Ausführung aufgetreten: " << e.what() << std::endl;
    std::cerr << "An error occured during execution: " << e.what() << std::endl;
  }
}
```

trie.cpp

```
#include <iostream>
#include <map>

#include "trie.hpp"

void put_recursive(trie_node* node, const std::string& key, const int current_char, const int data) {
  throw std::logic_error("Not yet implemented");
}

int trie::put(const std::string& key, const int data) {
  throw std::logic_error("Not yet implemented");
}

int get_recursive(const trie_node* node, const std::string& key, const int current_char) {
  throw std::logic_error("Not yet implemented");
}

int trie::get(const std::string& key) const {
  throw std::logic_error("Not yet implemented");
}

bool trie::empty() const {
  throw std::logic_error("Not yet implemented");
}

size_t size_recursive(const trie_node *node) {
  throw std::logic_error("Not yet implemented");
}

size_t trie::size() const {
  throw std::logic_error("Not yet implemented");
}
```

trie.hpp

```
#ifndef TRIE_HPP
#define TRIE_HPP

#include "trie_node.hpp"

/**
 * @brief Die Datenstruktur trie bildet Strings auf nicht negative Zahlen ab.
 */
/**
 * @brief The data structure trie maps strings to non-negative integers.
 **/
class trie {
  public:
    /**
     * @brief Prüft, ob der trie leer ist.
     *
     * @return true Wenn der trie leer ist.
     * @return false Wenn der trie Elemente enthält.
     */
    /**
     * @brief Checks if the trie is empty.
     *
     * @return true if the trie is empty.
     * @return false if the trie contains elements.
     */
    bool empty() const;

    /**
     * @brief Berechnet die Größe des trie. Die Größe ist die Anzahl der
     *   gesetzten Werte (nicht die Anzahl der Knoten im Baum).
     *
     * @return size_t Die Größe des tries
     */
    /**
     * @brief Calculates the size of the trie. The size is defined as the
     * number of bound values (and not as the number of nodes in the trie).
     *
     * @return The size of the trie.
     */
    size_t size() const;
```

```
    /**
     * @brief Fügt einen Wert in den Trie ein. Diese Operation überschreibt bereits
     *   gespeicherte Werte.
     *
     * @param key Der Schlüssel (dieser darf nicht leer sein), unter dem der Wert
     *   gespeichert werden soll.
     * @param data Der zu speichernde Wert. Nur Werte, die größer oder gleich 0 sind, sind erlaubt.
     * @return -1, wenn kein Wert eingefügt werden konnte und 0 bei Erfolg.
     */
    /**
     * @brief Inserts a value into the trie. The operation overwrites already existing
     *   bindings.
     *
     * @param key The key (which is not allowed to be empty) that should be used.
     * @param data The data to store. Only values >= 0 are allowed.
     * @return -1 if the value cannot be inserted and 0 on success.
     */
    int put(const std::string& key, const int data);

    /**
     * @brief Gibt einen Zeiger auf einen Wert zurück, der im Trie gespeichert ist.
     *
     * @param key Der Schlüssel, dessen Wert gesucht wird.
     * @return Wenn ein Wert gespeichert wurde. Wenn der Schlüssel nicht verwendet
     *   wird, wird eine negative Zahl zurück gegeben.
     */
    /**
     * @brief Returns a pointer to a value stored in the trie.
     *
     * @param key The key of the value.
     * @return The function retuns the value if the key exists.
     *   Otherwise, it returns a negative value.
     */
    int get(const std::string& key) const;

  ////////////////////////////////////////////////////////////////////////
  // Vorgegebene Methoden zum Debuggen und Testen
  // Given methods for debugging and testing
    trie() : head(nullptr) {
    }

    trie_node* get_head() {
      return head;
    }

    void set_head(trie_node* h) {
      head = h;
    }

    void print() const{
      if(empty()) {
        std::cout << "  empty trie" << std::endl;
        return;
      }

      print_recursive("", head, false);
    }

  private:
    void print_recursive(const std::string& prefix, const trie_node* node, bool isLeft) const {
      if (node == nullptr) {
        return;
      }

      auto iterator = node->children.begin();

      while (iterator != node->children.end()) {
        std::cout << prefix;

        std::cout << (isLeft ? "├──" : "└──" );

        // print the value of the node
        if (iterator->second->data >= 0) {
          std::cout << iterator->first << " [" << iterator->second->data << "]" << std::endl;
        }
        else {
          std::cout << iterator->first << std::endl;
        }

        auto value = iterator->second;
        iterator++;

        print_recursive(prefix + (isLeft ? "│   " : "    "), value, iterator != node->children.end());
      }
    }

  public:
    trie_node *head;
};

#endif //TRIE_HPP
```

trie_node.hpp

```
#ifndef TRIE_NODE_HPP
#define TRIE_NODE_HPP
```

```
/**
 * @brief Darstellung eines Trie-Knotens.
 *
 */
class trie_node {
  public:
    /**
     * @brief Der Wert, der diesem Knoten zugeordnet wurde. Es sind nur positive
     *   Werte oder 0 erlaubt. Ein negativer Wert zeigt an, dass dieser Knoten
     *   nicht verwendet wird.
     */
    /**
     * @brief The value that is bound to this node. Only positive values and 0
     *   are allowed. A negative value indicates that the node is not used.
     */
    int data;

    /**
     * @brief Nachfolgeknoten und die zugehörigen Schlüssel, mit denen man sie erreicht.
     */
    /**
     * @brief Successor nodes of this <tt>trie_node</tt>.
     */
    std::map<unsigned char, trie_node*> children;

  public:
    trie_node() : data(-1) {
    }

};
#endif // TRIE_NODE_HPP
```

Folgende Dateien haben Sie in der Prüfung bearbeitet bzw. erstellt:

main.cpp

```
#include <iostream>
#include <map>

#include "trie.hpp"

int main() {
  try {
    trie t;

    std::cout << "Starte mit einem leeren Trie (erwartet 'empty trie')." << std::endl;
    std::cout << "Starting with an empty trie (expected 'empty trie')." << std::endl;
    t.print();

    std::cout << std::endl;

    std::cout << "Prüfe, ob trie leer ist (erwartet 1): " << t.empty() << std::endl;
    std::cout << "Checking if trie is empty (expected 1): " << t.empty() << std::endl;

    std::cout << std::endl;

    std::cout << "Bilde 'car' auf den Wert 15 ab." << std::endl;
    std::cout << "Mapping 'car' to value 15." << std::endl;
    t.put("car", 15);

    std::cout << "Erwartet/Expected:" << std::endl <<
        R"(└──c
      ├──a
      │   └──r [15]
  )";

    std::cout << std::endl << "Aktueller Zustand:" << std::endl << "Current status:" << std::endl;
    t.print();



    std::cout << std::endl;
    std::cout << std::endl;
    std::cout << "Bilde 'hand' auf den Wert 10 ab." << std::endl;
    std::cout << "Mapping 'hand' to value 10." << std::endl;
    t.put("hand", 10);

  std::cout << "Erwartet/Expected:" << std::endl <<
        R"(└──c
      ├──a
      │   └──r [15]
  └──h
      └──a
          ├──n
          │   └──d [10]
  )";

    std::cout << std::endl << "Aktueller Zustand:" << std::endl << "Current status:" << std::endl;
    t.print();

    std::cout << std::endl;
    std::cout << std::endl;
    std::cout << "Bilde 'help' auf den Wert 7 ab." << std::endl;
```

```
    std::cout << "Bilde 'he' auf den Wert 1 ab." << std::endl;
    std::cout << "Mapping 'help' to value 7." << std::endl;
    std::cout << "Mapping 'he' to value 1." << std::endl;
    t.put("help", 7);
    t.put("he", 1);

    std::cout << "Erwartet/Expected:" << std::endl <<
        R"(└─c
      ├─a
      │  └─r [15]
  └─h
      └─a
         ├─n
         │  └─d [10]
      └─e [1]
         └─l
            └─p [7]

  )";
    std::cout << std::endl << "Aktueller Zustand:" << std::endl << "Current status:" << std::endl;
    t.print();
    std::cout << std::endl;
    std::cout << std::endl;

    std::cout << "Prüfe die Größe des trie (erwartet 4): " << t.size() << std::endl;
    std::cout << "Checking size of trie (expected 4): " << t.size() << std::endl;

    std::cout << std::endl;
    std::cout << std::endl;

    std::cout << "Suche nach Einträgen:" << std::endl;
    std::cout << "Searching for entries:" << std::endl;

    int value = t.get("help");
    std::cout << "Suche nach 'help' (erwartet 7): " <<  value << std::endl;
    std::cout << "Searching for 'help' (expected 7): " << value << std::endl;

    std::cout << std::endl;

    value = t.get("car");
    std::cout << "Suche nach 'car' (erwartet 15): " << value << std::endl;
    std::cout << "Searching for 'car' (expected 15): " << value << std::endl;

    return 0;
  } catch(std::logic_error e) {
    std::cerr << "Ein Fehler ist bei der Ausführung aufgetreten: " << e.what() << std::endl;
    std::cerr << "An error occured during execution: " << e.what() << std::endl;
  }
}
```

trie.cpp

```
#include <iostream>
#include <map>

#include "trie.hpp"

void put_recursive(trie_node* &node, const std::string& key, const int current_char, const int data) {
    if (!key.size()) return;
    //std::cout << "Check2"<<std::endl;
    if (node == nullptr) {
        node = new trie_node();
        //std::cout << "Check7"<<std::endl;
    }
    //std::cout << "Check3"<<std::endl;
    if (current_char+1 >= key.size()) {
        node->data = data;
        //std::cout << "Check4"<<std::endl;
        return;
    }
    if (!(node->children.find(key[current_char+1]) != node->children.end())) {
        node->children[key[current_char+1]] = new trie_node;
        *node->children[key[current_char+1]] = trie_node();
        //std::cout << "Check5"<<std::endl;
    }
    //std::cout << "Check6"<<std::endl;
    put_recursive(node->children[key[current_char+1]], key, current_char+1, data);
}
// check if exists (map.find(12) != map.end())
int trie::put(const std::string& key, const int data) {
    if (!key.size()) return -1;
    //std::cout << "Check"<<std::endl;
    //head = new trie_node();
    put_recursive(head, key, -1, data);
    return 0;
  throw std::logic_error("Not yet implemented");
}

int get_recursive(trie_node* node, const std::string& key, const int current_char) {
    if (node == nullptr) return -1;
    if (current_char == key.size()) return node->data;
    if (node->children.find(key[current_char]) == node->children.end()) return -1;
    return get_recursive(node->children[key[current_char]], key, current_char+1);
}
/*
gibt den Wert zurück, der bereits im Baum für den Schlüssel
gespeichert ist. Dafür durchsucht die Funktion den Baum
```

```
rekursiv nach dem entsprechenden Knoten. Wird der Wert gefunden,
so gibt die Funktion das gespeicherte Datum zurück.
Ist für den Schlüssel kein Wert gespeichert, so gibt die
Funktion -1 zurück.
*/
int trie::get(const std::string& key) const {
    return get_recursive(head, key, 0);
  throw std::logic_error("Not yet implemented");
}

bool trie::empty() const {
    if (head == nullptr) return 1;
    return 0;
  throw std::logic_error("Not yet implemented");
}

size_t size_recursive(const trie_node *node) {
    if (node == nullptr) return 0;
    int sum = 0;
    auto iterator = node->children.begin();
    while (iterator != node->children.end()) {
        sum += size_recursive(iterator->second);
        iterator++;
    }
    if (node->data != -1) sum++;
    return sum;
}

size_t trie::size() const {
    return size_recursive(head);
  throw std::logic_error("Not yet implemented");
}
```

Bewertung mit Unittests

Testname	Assertions	Bestanden	Punkte
empty__empty_trie	1	1	1.00
empty__trie_with_1_binding	1	1	1.00
empty__trie_with_1_binding_one_char.	1	1	1.00
empty__trie_with_3_bindings.	1	1	1.00
get__empty_trie	1	1	1.00
get__inner_on_trie_with_3_bindings	1	1	1.00
get__leaf_on_trie_with_3_bindings	1	1	1.00
get__leaf_on_trie_with_single_binding	1	1	1.00
get__non_existing_binding	1	1	1.00
get__non_existing_binding_2	1	1	1.00
get__non_existing_binding__single.	1	1	1.00
get__simple_case	1	1	1.00
put__2_bindings	2	2	1.00
put__bindings_to_trie	3	3	1.00
put__check_inner_nodes_are_empty	6	6	1.00
put__first_binding_on_empty_trie	1	1	1.00
put__inner_binding	3	3	1.00
put__overwrite_inner_binding	3	3	1.00
put__overwrite_leaf_binding	2	2	1.00
put__sequence_of_as	4	4	1.00
put__single_letter_binding	1	1	1.00
size__empty_trie	1	1	1.00
size__trie_with_1_binding	1	1	1.00
size__trie_with_2_bindings	1	1	1.00
size__trie_with_3_bindings_with_inner.	1	1	1.00