10 STL Funktionen

10.1 Übungsaufgabe

Erweitern Sie die Klasse `cis_wstring` der letzten Aufgabe, verwenden Sie allerdings zum Speichern der Buchstaben nicht ein Array von `wchar_t` sondern den Container der STL `vector`. Nutzen Sie so gut es geht STL-Funktionen.

```
class Cis_wstring {
public:
    \\liefert die Anzahl der Buchstaben der Cis_wstring Daten
    int get_anzahl()

private:
    \\alle Buchstaben der Cis_wstring Daten
    vector <wchar_t> data

    \\Anzahl der Buchstaben in data
    int anzahl

    \\liefert wahr/falsch, ob der erste Buchstaben des cis_
    \\ wstrings ein Großbuchstabe ist (Verwenden Sie
    \\ Libraryfunktionen zum Testen)
    bool first_is_upper()

    \\ alle Großbuchstaben in Kleinbuchstaben und umgekehrt
    \\ konvertieren und ausgeben (Verwenden Sie Libraryfunktionen).
    void translate()

    \\ testet, ob Cis_wstring ein Palindrom ist. (Groß und
    \\ Kleinbuchstaben sollen nicht unterschieden werden)
    bool is_palindrom()
}
```

`str++`: Konvertiert die Buchstaben eines `Cis_wstrings` in Großbuchstaben

`--str`: Konvertiert die Buchstaben eines `Cis_wstrings` in Kleinbuchstaben

Schreiben Sie eine eigene Vergleichsfunktion `str1 < str2`, die beim alphabetischen Vergleich der beiden `Cis_wstrings` `str1 < str2 true` liefert, falls `str1` alphabetisch kleiner als `str2` ist. Ihr Stringvergleich soll eventuelle Groß- und Kleinbuchstaben nicht unterscheiden.

Nun soll folgende `main()` Funktion möglich sein:

```
int main() {
```

```
// Setzen des locale auf utf-8
setlocale(LC_ALL, "de_DE.utf-8");
wstring eingabe;
wstring ausgabe;
Cis_wstring text;

wcout << L"Bitte geben sie einen String ein :";
getline(wcin, eingabe);
text.set_cis_wstring_data(eingabe);
// eingabe in den Cis_wstring text eintragen

wcout << L"Die Anzahl der Buchstaben ist : " <<
text.get_anzahl_chars() << endl;

// testet ob der erste Buchstabe groß ist:
if (text.first_is_upper()) {
    wcout << L"Der erste Buchstabe des Strings ist ein
    großer Buchstabe" << endl;
}

// Konvertieren auf Groß/Kleinschreibung:
text.translate();
text.get_cis_wstring_data(ausgabe);
wcout << L"TRANSLATE: Cis_wstring ist : " << ausgabe << endl;

// Konvertieren auf upper:
text++;
text.get_cis_wstring_data(ausgabe);
wcout << L"TOUPPER: Cis_wstring ist : " << ausgabe << endl;

// Konvertieren auf lower:
--text;
text.get_cis_wstring_data(ausgabe);
wcout << L"TOLOWER: Cis_wstring ist : " << ausgabe << endl;

// is palindrom (Funktion aus der letzten Aufgabe)
if (text.is_palindrom()) {
    text.get_cis_wstring_data(ausgabe);
    wcout << L"Ich bin ein Palindrom " << ausgabe << endl;
} else {
    wcout << L"Ich bin kein Palindrom " << ausgabe << endl;
}

//Test der < Funktion
wstring eingabe2;
wstring ausgabe2;
wcout << L"Bitte geben sie einen String ein :";
getline(wcin, eingabe2);
Cis_wstring text2(eingabe2);
text.get_cis_wstring_data(ausgabe);
text2.get_cis_wstring_data(ausgabe2);
```

```
    if (text < text2) {
        wcout << ausgabe << L" ist kleiner als " << ausgabe2 << endl;
    } else {
        wcout << ausgabe2 << L" ist nicht kleiner als " << ausgabe << endl;
    }
}
```

10.2 Wiederholung und zusätzliche Informationen

10.2.1 Container der STL

In der Standard Template Library (STL) gibt es verschiedene Container, die Objekte vom gleichen Datentyp speichern.

Container, die Sequenzen von Objekten speichern:

- Vector: eindimensionaler Vektor aus der Mathematik, Einfügen/ Löschen am Ende effizient
- Deque: eindimensionaler Vektor aus der Mathematik, Einfügen/Löschen am Ende und am Anfang effizient
- List: eindimensionaler Vektor aus der Mathematik, Elemente sind direkt miteinander verkettet, Einfügen/Löschen daher überall effizient

Container, die Mengen von Objekten speichern:

- Set: Geordnete Menge von Objekten
- Map: Geordnete Menge von Tupeln von Objekten
- Multiset: Menge von Objekten, auch identische Objekte möglich
- Multimap: Menge von Tupeln von Objekten, auch identische Objekte möglich
- Unordered set: Ungeordnete Menge von Objekten (Hash)
- Unordered map: Ungeordnete Menge von Tupeln von Objekten (Hash)

10.2.2 Templates

Templates sind Schablonen für konkrete Objekte mit Platzhaltern. Statt einem Datentyp wird X verwendet. Der Objekttyp wird erst bei der Deklaration festgelegt. Templates sind für Funktionen und Klassen verwendbar.

Das folgende Beispiel zeigt eine solche generische Funktion:

```

#include <iostream>
using namespace std;

//Generische Funktion wird deklariert für Datentyp X
template<class X> void my_swap(X &a, X &b) {
    X temp;
    temp = a;
    a = b;
    b = temp;
}

int main() {
    wcout << L"Swap mit Integers" << endl;
    int firstInt = 1;
    int secondInt = 2;

    //Funktion wird mit Integerwerten aufgerufen
    my_swap(firstInt, secondInt);
    wcout << L"First Int: " << firstInt << endl;
    wcout << L"Second Int: " << secondInt << endl;
    wcout << endl;
    wcout << L"Swap mit Wstrings" << endl;
    wstring firstWstring = L"Erster";
    wstring secondWstring = L"Zweiter";

    //Funktion wird mit Wstringwerten aufgerufen
    my_swap(firstWstring, secondWstring);
    wcout << L"First Wstring: " << firstWstring << endl;
    wcout << L"Second Wstring: " << secondWstring << endl;
}

```

10.3 Lösungsvorschlag

10.3.1 Aufteilung in CPP und H

CPP:

Enthält die Klassenimplementierung und die Hauptmethode

```

/*Autor: Nicola Greth
 * Uebung 8
 * Klasse Cis_wstring
 */
#include <iostream>
#include <vector>
#include <locale>
#include "Cis_wstring.h"
using namespace std;

//Konstruktor, der alle Daten auf ein Space setzt
Cis_wstring::Cis_wstring() {

```

```
    anzahl = 0;

    for (int i = 0; i < max_anzahl; i++) {
        data.push_back(L' ');
    }
}
;

//Konstruktor, der ein Argument in den Buchstabenspeicher
//überträgt
Cis_wstring::Cis_wstring(wstring eingabe) {
    //data wird geleert bevor eingabe eingetragen wird
    data.clear();

    if (eingabe.size() <= max_anzahl) {
        anzahl = eingabe.size();

        for (int i = 0; i < anzahl; i++) {
            data.push_back(eingabe.at(i));
        }
    }
}
;

//liefert die Anzahl der Buchstaben der Cis_wstring Daten
int Cis_wstring::get_anzahl_chars() {
    return anzahl;
}
;

//liefert als wstring den Inhalt wchar_t Daten aus data[132]
void Cis_wstring::get_cis_wstring_data(wstring &text) {
    text.clear();

    for (int i = 0; i < anzahl; i++) {
        text = text + data[i];
    }
}
;

//trägt den String text in die Cis_wstring Daten ein =
//Konstruktor mit Argument
void Cis_wstring::set_cis_wstring_data(wstring text) {
    //data wird geleert bevor eingabe eingetragen wird
    data.clear();

    if (text.size() <= max_anzahl) {
        anzahl = text.size();

        for (int i = 0; i < anzahl; i++) {
            data.push_back(text.at(i));
        }
    }
}
```

```

    }
  }
}
;

//testet ob der erste Buchstabe ein Großbuchstabe ist
bool Cis_wstring::first_is_upper() {
  if (anzahl > 0) {
    if (iswupper(data.at(0))) {
      return true;
    }
  }
  return false;
}
;

//übersetzt Groß-/ und Kleinbuchstaben
void Cis_wstring::translate() {
  for (int i = 0; i < anzahl; i++) {

    //wenn es ein Großbuchstabe ist, wird er in einen
    //Kleinbuchstaben konvertiert
    if (iswupper(data.at(i))) {
      data.at(i) = towlower(data.at(i));
    }

    //wenn es ein Kleinbuchstabe ist, wird er in einen
    //Kleinbuchstaben konvertiert
    else if (iswlower(data.at(i))) {
      data.at(i) = towupper(data.at(i));
    }
  }
}
;

//testet ob der Cis_wstring ein Palindrom ist
bool Cis_wstring::is_palindrom() {
  for (int i = 0; i < anzahl / 2; i++) {
    if (data.at(i) != data.at(anzahl - 1 - i)) {
      return false;
    }
  }
  return true;
}
;

//Überladen des Operators ++ als Konvertieren in
Großbuchstaben
void Cis_wstring::operator++(int notused) {
  for (int i = 0; i < anzahl; i++) {
    data.at(i) = towupper(data.at(i));
  }
}
;

```

```
    }
}
;

//Überladen des Operators -- als Konvertieren in
Kleinbuchstaben
void Cis_wstring::operator--() {
    for (int i = 0; i < anzahl; i++) {
        data.at(i) = tolower(data.at(i));
    }
}
;

//Überladen des Operators < als alphabetischen Vergleich
bool Cis_wstring::operator<(Cis_wstring& other) {
    //other wird in wstring konvertiert
    wstring otherOutput;
    other.get_cis_wstring_data(otherOutput);

    //die Mindestanzahl der zu vergleichenden chars wird
    //berechnet
    int min_anzahl = min(anzahl, other.get_anzahl_chars());

    //für jeden Buchstaben wird verglichen
    for (int i = 0; i < min_anzahl; i++) {

        //wenn der Buchstabe von Data < dem Buchstaben von
        otherOutput ist wird true zurückgegeben

        if (tolower(data.at(i)) <
            tolower(otherOutput.at(i))) {
            return true;
        }

        //wenn der Buchstabe von Data > dem Buchstaben von
        otherOutput ist wird false zurückgegeben

        if (tolower(data.at(i)) >
            tolower(otherOutput.at(i))) {
            return false;
        }
    }

    //wenn unser Cis_wstring komplett durchlaufen wurde und
    //kein größerer oder kleinerer Buchstabe gefunden wurde
    //und der andere Cis_wstring aber noch Buchstaben hat,
    //dann ist unser Cis_wstring kleiner
    if (anzahl < other.get_anzahl_chars()) {
        return true;
    }
}
```

```
//wenn das komplette Wort durchlaufen wurde und kein
//größerer oder kleinerer Buchstabe gefunden wurde und
//der andere Cis_wstring kleiner als unser Cis_wstring ist,
// wird false zurückgegeben
return false;
}
;

int main() {
    // Setzen des locale auf utf-8
    setlocale(LC_ALL, "de_DE.utf-8");
    wstring eingabe;
    wstring ausgabe;
    Cis_wstring text;

    wcout << L"Bitte geben sie einen String ein :";
    getline(wcin, eingabe);
    text.set_cis_wstring_data(eingabe); // eingabe in den
    //Cis_wstring text eintragen

    wcout << L"Die Anzahl der Buchstaben ist : " <<
    text.get_anzahl_chars() << endl;

    // test ob der erste Buchstabe groß ist:
    if (text.first_is_upper()) {
        wcout << L"Der erste Buchstabe des Strings ist ein
        großer Buchstabe" << endl;
    }

    // Konvertieren auf Groß/Kleinschreibung:
    text.translate();
    text.get_cis_wstring_data(ausgabe);
    wcout << L"TRANSLATE: Cis_wstring ist : " << ausgabe <<
    endl;

    // Konvertieren auf upper:
    text++;
    text.get_cis_wstring_data(ausgabe);
    wcout << L"TOUPPER: Cis_wstring ist : " << ausgabe << endl;

    // Konvertieren auf lower:
    --text;
    text.get_cis_wstring_data(ausgabe);
    wcout << L"TOLOWER: Cis_wstring ist : " << ausgabe << endl;

    // is palindrom, verwenden Sie Ihre Funktion aus der
    //letzten Hausaufgabe:
    if (text.is_palindrom()) {
        text.get_cis_wstring_data(ausgabe);
        wcout << L"Ich bin ein Palindrom " << ausgabe << endl;
    } else {
```

```
        wcout << L"Ich bin kein Palindrom " << ausgabe << endl;
    }

    //neu eingefügt zum Testen der < Funktion
    wstring eingabe2;
    wstring ausgabe2;
    wcout << L"Bitte geben sie einen String ein :";
    getline(wcin, eingabe2);

    Cis_wstring text2(eingabe2);
    text.get_cis_wstring_data(ausgabe);
    text2.get_cis_wstring_data(ausgabe2);

    if (text < text2) {
        wcout << ausgabe << L" ist kleiner als " << ausgabe2 << endl;
    } else {
        wcout << ausgabe << L" ist nicht kleiner als " << ausgabe2 << endl;
    }
    return 0;
}
```

H-Datei:

Enthält die Klassendeklaration

```
/*Autor: Nicola Greth
 * Uebung 8
 * Klasse Cis_wstring
 */
#ifndef CISWSTRING_H
#define CISWSTRING_H
#include <vector>
using namespace std;

//Klassendeklaration
class Cis_wstring {
    //public Komponenten
public:
    Cis_wstring();
    Cis_wstring(wstring eingabe);
    int get_anzahl_chars();
    void get_cis_wstring_data(wstring &text);
    void set_cis_wstring_data(wstring text);
    void cis_wstring_to_upper();
    bool first_is_upper();
    void translate();
    bool is_palindrom();
    void operator++(int notused);
    void operator--();
    bool operator<(Cis_wstring& other);

    //private Komponenten
private:
```

```

        vector<wchar_t> data;
        int anzahl;
        static const int max_anzahl = 132;
};
#endif

```

10.3.2 Aufteilung in CPP und HPP

CPP:

Enthält die Main Methode

```

/*Autor: Nicola Greth
 * Uebung 8
 * Klasse Cis_wstring
 */
#include <iostream>
#include <vector>
#include <locale>
#include "Cis_wstring.hpp"
using namespace std;

int main() {
    // Setzen des locale auf utf-8
    setlocale(LC_ALL, "de_DE.utf-8");
    wstring eingabe;
    wstring ausgabe;
    Cis_wstring text;
    wcout << L"Bitte geben sie einen String ein :";
    getline(wcin, eingabe);

    text.set_cis_wstring_data(eingabe); // eingabe in den
    //Cis_wstring text eintragen
    wcout << L"Die Anzahl der Buchstaben ist : " <<
    text.get_anzahl_chars() << endl;

    // test ob der erste Buchstabe groß ist:
    if (text.first_is_upper()) {
        wcout << L"Der erste Buchstabe des Strings ist ein
        großer Buchstabe" << endl;
    }

    // Konvertieren auf Groß/Kleinschreibung:
    text.translate();
    text.get_cis_wstring_data(ausgabe);
    wcout << L"TRANSLATE: Cis_wstring ist : " << ausgabe << endl;

    // Konvertieren auf upper:
    text++;
    text.get_cis_wstring_data(ausgabe);
    wcout << L"TOUPPER: Cis_wstring ist : " << ausgabe << endl;
}

```

```
// Konvertieren auf lower:
--text;
text.get_cis_wstring_data(ausgabe);
wcout << L"TOLOWER: Cis_wstring ist : " << ausgabe << endl;

// is palindrom, verwenden Sie Ihre Funktion aus der
// letzten Hausaufgabe:
if (text.is_palindrom()) {
    text.get_cis_wstring_data(ausgabe);
    wcout << L"Ich bin ein Palindrom " << ausgabe << endl;
} else {
    wcout << L"Ich bin kein Palindrom " << ausgabe << endl;
}

//neu eingefügt zum Testen der < Funktion
wstring eingabe2;
wstring ausgabe2;
wcout << L"Bitte geben sie einen String ein :";
getline(wcin, eingabe2);

Cis_wstring text2(eingabe2);
text.get_cis_wstring_data(ausgabe);
text2.get_cis_wstring_data(ausgabe2);

if (text < text2) {
    wcout << ausgabe << L" ist kleiner als " << ausgabe2
        << endl;
} else {
    wcout << ausgabe << L" ist nicht kleiner als " <<
        ausgabe2 << endl;
}
return 0;
}
```

HPP:

Enthält Klassendeklaration und Implementation

```
/*Autor: Nicola Greth
* Uebung 8
* Klasse Cis_wstring
*/
#ifndef CISWSTRING_HPP
#define CISWSTRING_HPP
#include <iostream>
#include <vector>
#include <locale>
using namespace std;

//Klassendeklaration
class Cis_wstring {
    //public Komponenten
public:
```

```

        Cis_wstring();
        Cis_wstring(wstring eingabe);
        int get_anzahl_chars();
        void get_cis_wstring_data(wstring &text);
        void set_cis_wstring_data(wstring text);
        void cis_wstring_to_upper();
        bool first_is_upper();
        void translate();
        bool is_palindrom();
        void operator++(int notused);
        void operator--();
        bool operator<(Cis_wstring& other);

//private Komponenten
private:
    vector<wchar_t> data;
    int anzahl;
    static const int max_anzahl = 2;
};

//Konstruktor, der alle Daten auf ein Space setzt
Cis_wstring::Cis_wstring() {
    anzahl = 0;

    for (int i = 0; i < max_anzahl; i++) {
        data.push_back(L' ');
    }
}
;

//Konstruktor, der ein Argument in den Buchstabenspeicher
überträgt
Cis_wstring::Cis_wstring(wstring eingabe) {
    //data wird geleert bevor eingabe eingetragen wird
    data.clear();

    if (eingabe.size() <= max_anzahl) {
        anzahl = eingabe.size();

        for (int i = 0; i < anzahl; i++) {
            data.push_back(eingabe.at(i));
        }
    }
}
;

//liefert die Anzahl der Buchstaben der Cis_wstring Daten
int Cis_wstring::get_anzahl_chars() {
    return anzahl;
}
;

```

```
//liefert als wstring den Inhalt wchar_t Daten aus data[132]
void Cis_wstring::get_cis_wstring_data(wstring &text) {
    text.clear();

    for (int i = 0; i < anzahl; i++) {
        text = text + data[i];
    }
}
;

//trägt den String text in die Cis_wstring Daten ein =
//Konstruktor mit Argument
void Cis_wstring::set_cis_wstring_data(wstring text) {
    //data wird geleert bevor eingabe eingetragen wird
    data.clear();

    if (text.size() <= max_anzahl) {
        anzahl = text.size();

        for (int i = 0; i < anzahl; i++) {
            data.push_back(text.at(i));
        }
    }
}
;

//testet ob der erste Buchstabe ein Großbuchstabe ist
bool Cis_wstring::first_is_upper() {
    if (anzahl > 0) {
        if (iswupper(data.at(0))) {
            return true;
        }
    }
    return false;
}
;

//übersetzt Groß-/ und Kleinbuchstaben
void Cis_wstring::translate() {
    for (int i = 0; i < anzahl; i++) {
        //wenn es ein Großbuchstabe ist, wird er in einen
        //Kleinbuchstaben konvertiert

        if (iswupper(data.at(i))) {
            data.at(i) = towlower(data.at(i));
        }

        //wenn es ein Kleinbuchstabe ist, wird er in einen
        //Kleinbuchstaben konvertiert
        else if (iswlower(data.at(i))) {

```

```

        data.at(i) = towupper(data.at(i));
    }
}
;

//testet ob der Cis_wstring ein Palindrom ist
bool Cis_wstring::is_palindrom() {
    for (int i = 0; i < anzahl / 2; i++) {
        if (data.at(i) != data.at(anzahl - 1 - i)) {
            return false;
        }
    }
    return true;
}
;

//Überladen des Operators ++ als Konvertieren in
Großbuchstaben
void Cis_wstring::operator++(int notused) {
    for (int i = 0; i < anzahl; i++) {
        data.at(i) = towupper(data.at(i));
    }
}
;

//Überladen des Operators -- als Konvertieren in
//Kleinbuchstaben
void Cis_wstring::operator--() {
    for (int i = 0; i < anzahl; i++) {
        data.at(i) = tolower(data.at(i));
    }
}
;

//Überladen des Operators < als alphabetischen Vergleich
bool Cis_wstring::operator<(Cis_wstring& other) {
    //other wird in wstring konvertiert
    wstring otherOutput;
    other.get_cis_wstring_data(otherOutput);

    //die Mindestanzahl der zu vergleichenden chars wird
    //berechnet
    int min_anzahl = min(anzahl, other.get_anzahl_chars());

    //für jeden Buchstaben wird verglichen
    for (int i = 0; i < min_anzahl; i++) {
        //wenn der Buchstabe von Data < dem Buchstaben von
        otherOupt ist, wird true zurückgegeben

        if (tolower(data.at(i)) <

```

```
        tolower(otherOutput.at(i))) {
            return true;
        }

        //wenn der Buchstabe von Data > dem Buchstaben von
        //otherOutput ist, wird false zurückgegeben
        if (tolower(data.at(i)) >
            tolower(otherOutput.at(i))) {
            return false;
        }
    }

    //wenn unser Cis_wstring komplett durchlaufen wurde und
    //kein größerer oder kleinerer Buchstabe gefunden wurde und
    //der andere Cis_wstring aber noch Buchstaben hat,
    //dann ist unser Cis_wstring kleiner

    if (anzahl < other.get_anzahl_chars()) {
        return true;
    }

    //wenn das komplette Wort durchlaufen wurde und kein
    //größerer oder kleinerer Buchstabe gefunden wurde und der
    //andere Cis_wstring kleiner als unser Cis_wstring ist,
    // wird false zurückgegeben
    return false;
}
;
//Ende der Klassendeklaration
#endif
```