Programmierung 2 VU 051020

Übungseinheit 1

WS 2020

universität
wien

Vector – Überblick

Bis zur nächsten Woche....

Klasse Vector Kann beliebig viele Werte speichern.

Dynamisches Wachstum Speicherplatz vergrößern, wenn der Vector voll ist.

Datentyp Für die erste Version genügen double - Werte. Templates werden in Woche 3 eingebaut.

Aufbau Den Code in ein Header File schreiben, **nicht** auf zwei (Cpp und Header) Files aufteilen.

Die folgenden Anforderungen sind mindestens zu erfüllen.

universität wien

Vector – Instanzvariablen

size_t sz	Enthält die Anzahl der Elemente im Vector.
size_t max_sz	Enthält die maximale Anzahl an Elementen die möglich sind.
double* values	Zeigt auf ein Feld, welches die Elemente des Vectors beinhaltet.

Tipp: Sie können eine zusätzliche Variable **static constexpr size_t min_sz**, die eine Mindestgröße (5) für max_sz festlegt einführen. Dies erleichtert den Umgang mit einigen Sonderfällen.

universität wien

Vector – Konstruktoren/Destruktor

Default	Liefert einen leeren Vector.
size_t n	Liefert einen Vector mit Platz für n Elemente.
std::initializer_list<double>	Liefert einen Vector mit spezifiziertem Inhalt.
Kopierkonstruktor	Liefert einen Vector mit demselben Inhalt.
Destruktor	Gibt den Speicherplatz wieder frei.

Beachten Sie Speicherlecks und Sonderfälle wie Vector(0) und Vector({}).

universität wien

Vector – Methoden (Teil 1)

`size_t size() const`	Retourniert Anzahl der gespeicherten Elemente.
`bool empty() const`	Retourniert `true` falls der Vector leer ist, ansonsten `false`.
`void clear()`	Löscht alle Elemente aus dem Vector.
`void reserve(size_t n)`	Kapazität des Vectors wird auf n vergrößert, falls nötig.
`void shrink_to_fit()`	Kapazität wird auf Anzahl der Elemente reduziert.
`void push_back(double x)`	Fügt eine Kopie von x am Ende des Vectors hinzu.
`void pop_back()`	Entfernt das letzte Element im Vector. Wirft eine Exception, falls der Vector leer war.

universität wien

Vector – Methoden (Teil 2)

`Kopierzuweisungsoperator(operator=)` Das this-Objekt übernimmt die Werte vom Parameter.

`double& operator[](size_t index)` Abgesichert, retourniert das Element an der gegebenen Position (index).

`const double& operator[](size_t index) const` Abgesichert, retourniert das Element an gegebener Position (index).

`size_t capacity() const` Retourniert aktuelle Kapazität des Vectors.

universität wien

Vector – Ausgabeoperator

ostream& operator<<(ostream&, const Vector&) Ausgabe in der Form: [Element1, Element2, Element3].

Beispiel Vector x({1,2,3,4}) → [1, 2, 3, 4]

Friend für operator<< ist erlaubt.

universität wien

Vector – Testfiles

Zum Testen Ihrer Implementierung sollten Sie ein eigenes main-File schreiben.
Als Ergänzung werden jede Woche Testfiles bereitgestellt.
Beachten Sie, dass diese nur als Hilfestellung gedacht sind und **keine Garantie** darstellen, dass Ihr Vector korrekt funktioniert.

Vector – Erweiterungen

Sie können zu Übungszwecken noch weitere Methoden und globale Funktionen implementieren (siehe Vorlesungsfolien PR1).
Folgende (verpflichtende) Erweiterungen werden in den nächsten beiden Übungsstunden vorgestellt:

- Iteratoren, ermöglichen es range-based-for-loops zu verwenden sowie Algorithmen der STL. (Woche 2)
- Templates, ermöglichen es "beliebige" Werte zu speichern, nicht nur double-Werte. (Woche 3)

Machen Sie sich bereits vor den Übungsstunden mit den Konzepten vertraut (bereitgestellte Videos).
Zum Test benötigen Sie einen Vector mit der beschriebenen Basisfunktionalität, Iteratoren und Templates.

universität wien