Informatik II

Oliver Jack

Fachhochschule Jena
Fachbereich Elektrotechnik und Informationstechnik

Sommersemester 2010

Inhalt

Vorlesung 13. Zusammenfassung

C++ in der Zusammenfassung

- Anwendungsgebiete und Spracheigenschaften
- Sprachkonzepte
- Die Standardbibliothek

C++ in der Welt der Programmiersprachen

Anwendungsgebiete von C++...

Systemprogrammierung

- Betriebssysteme
- eingebettete Systeme
- virtuelle Maschinen
- Treiber
- Signalprozessoren

. . .Anwendungsgebiete von C++

Anwendungsprogrammierung

- Textverarbeitung
- Tabellenkalkulation
- Informationssysteme
- Computer Aided Design

Spracheigenschaften von C++

- C++ basiert auf der Programmiersprache C, ISO/IEC 9899:1990
- C++ ist eine Multi-Paradigmen-Sprache

Multi-Paradigmen-Sprache

- Prozedurale Programmierung
- Modulare Programmierung
- Strukturierte Programmierung
- Programmierung mit selbst-definierten Datentypen
- Objektorientierte Programmierung
- Generische Programmierung mittels Templates.

Über C hinausgehende Eigenschaften

- weitere Datentypen
- Klassen
- Vererbung
- virtuellen Funktionen
- Ausnahmebehandlung
- Templates
- Namensräume
- Inline-Funktionen
- Überladen von Operatoren und Funktionsnamen
- Referenzen
- Operatoren zur Freispeicherverwaltung
- C++-Standardbibliothek inklusive Standard-Template-Library (STL)

Klassen...

Grundidee

- Gleichartige Objekte werden gleichartig verwaltet
- Vorlagen, aus denen zur Laufzeit konkrete Objekte erzeugt werden (Instanzen)
- Im Programm werden nur die Vorlagen definiert
- Datentypen und zugehörige Algorithmen (Methoden)
- Kapselung der Daten und Methoden

...Klassen

Realisierung in C++

```
class Klasse {
private:        // gekapselt
  int zahl;
  int funktion ();  // Methode
public:         // Schnittstelle nach aussen
  int id;
  Klasse ();    // Konstruktor
  ~Klasse ();   // Destruktor
};
```

Vererbung...

Grundidee

- Definition von neuen Objekten aus bereits vorhandenen Objekten: abgeleitete Klassen (Wiederverwendung)
- Ergänzung vorhandener Objekte um weitere Eigenschaften
- Modifikation vorhandener Objekteigenschaften (Überschreiben)
- hierarchische Strukturierung von Objekten

...Vererbung

Realisierung in C++

```
class Basis {
public:
  int allgemein;
  int basisfuntion();
};

class Abgeleitet : public Basis {
public:
  int spezial; // zusaetzliche Variable
  int basisfunktion(); // ueberschriebene Funktion
};
```

Virtuelle Funktionen...

Grundidee

- Definition abstrakter Schnittstellen, die erst in abgeleiteten Klassen implementiert werden
- Dynamische Bindung, erst zur Laufzeit wird die Einsprungadresse der Funktion ermittelt
- Methode wird durch das tatsächlich zur Laufzeit übergebene Objekt bestimmt
- Überschreiben von Funktionen

...Virtuelle Funktionen

Realisierung in C++

```
class basis {
public:
  int i;
  virtual void print_i()
  {
    cout << i << "␣in␣basis" << endl;
  }
};
class abgeleitet: public basis {
public:
  void print_i()
  {
    cout << i << "␣in␣abgeleitet" << endl;
  }
};
```

Ausnahmebehandlung...

Grundidee

- Behandlung von Fehlerzuständen im Programm
- Abarbeitung definierter Algorithmen im Fehlerfall
- Trennung von normaler Algorithmus-Abarbeitung und Fehler-Abarbeitung
- Strukturierte Bearbeitung von Fehlerzuständen

...Ausnahmebehandlung

Realisierung in C++

```
try {
  // Anweisungen
  throw (ausnahme1);
  // ...
}

catch (Ausnahmetyp1) {
  // Ausnahmebehandlung1
}

catch (Ausnahmetyp2) {
  // Ausnahmebehandlung2
  throw;  // Weiterreichen der Ausnahme
}
```

Templates...

Grundidee

- Generische Programmierung
- Objekte und Algorithmen werden unabhängig von den darin enthaltenen Datentypen definiert
- Wiederverwendung von Programm-Code in typsicherer Weise
- Parametrische Polymorphie (Datentypen sind Parameter des Programm-Codes)
- Templates für Funktionen und für Klassen

...Templates...

Realisierung in C++

```
// Template-Funktion
template <class TYP>
TYP max(TYP d1, TYP d2)
{
  if (d1 > d2) {
    return(d1);
  } else {
    return(d2);
}
```

...Templates

Realisierung in C++

```
// Template-Klasse
template <class TYPE>
class stack {
private:
   enum { EMPTY = -1};
   TYPE *s;
   int max_len, top;
public:
   stack(): max_len(1000)
   { s = new TYPE[1000]; top = EMPTY; }
   // ...
};
```

Überladen von Operatoren...

Grundidee

- Benutzung gleicher Syntax für gleiche Operationen auf verschieden Objekten
- Polymorhphie
- Bindung der Bedeutung des Operators an die verbundenen Objekte

...Überladen von Operatoren

Realisierung in C++

```
struct complex {
  float re;
  float im;
};
// Operatorfunktion +, Addition komplexer Zahlen
complex operator +(complex &zahl1, complex &zahl2)
{
  complex erg;
  erg.re = zahl1.re + zahl2.re;
  erg.im = zahl1.im + zahl2.im;
  return erg;
}
```

Die C++ Standardbibliothek...

Grundidee

- Algorithmen, die fest zur Programmiersprache gehören
- Stream-orientierte Ein- und Ausgabe (mittels Operatoren)
- Zeichenkettenbearbeitung (String-Klasse)
- Container und Iteratoren (vector, stack, queue, etc.)

...Die C++ Standardbibliothek...

```
cout << variable << endl;

string s1 = "Hallo"; // Initialisierung
string s2 = "Welt";  // dito
string s3 = s1 + s2; // Konkatenation
s3 += '!'; // Ausrufzeichen anhaengen
string name = "Hans␣Maier";
string nachname = name.substr(5,5); // "Maier"
name.replace(0,3,"Herbert"); // "Herbert Maier"
```

...Die C++ Standardbibliothek

```
typename Container::iterator it;
  for(it = c.begin(); it != c.end(); it++)
    cout << *it << " ";
  cout << endl;
  cout << "size() "          << c.size()
       << " max_size() "     << c.max_size()
       << " front() "        << c.front()
       << " back() "         << c.back()
       << endl;
}
```

Evaluation

Evaluationsergebnisse und Diskussion

Ende

Ende der Veranstaltung