Christoph Ott Institut für Angewandte Informationsverarbeitung 27. August 2008
Blatt 3

WiMa-Praktikum/ Programmier-Praktikum (SS 2008) Präsenz-Übungsaufgaben

Tutoriumstermin: 27. August 2008 11-13 Uhr

1 Mathematische Funktionen

Es gibt in C++ natürlich auch mathematische Funktionen und Konstanten, die man nach Einbinden der Bibliothek <*cmath*> ohne Weiteres verwenden kann. Schreibt ein Programm, das über den Argumentvektor sowohl einen Buchstaben als auch einen reellen Wert entgegennimmt und je nach dem übergebenen Buchstaben eine entsprechende Funktion von <*cmath*> mit dem übergebenen Wert aufruft. Die übergebenen Buchstaben haben folgende Bedeutung:

- **e**: Die Exponentialfunktion (*exp()*) soll mit dem übergebenen Wert aufgerufen werden.
- **l**: Die Logarithmusfunktion (*log()*) soll mit dem übergebenen Wert aufgerufen werden.
- **p**: Die Zahl π (*M_PI*) soll mit dem übergebenen Wert multipliziert werden.
- **s**: Die Wurzelfunktion (*sqrt()*) soll mit dem übergebenen Wert aufgerufen werden.
- **t**: Die Tangensfunktion (*tan()*) soll mit dem übergebenen Wert aufgerufen werden.
- anderer Buchstabe: Eine Fehlermeldung soll auf *stderr* ausgegeben werden und das Programm soll mit Exit-Status 1 beendet werden. Das Programm soll ebenso mit Fehlermeldung und Exit-Status 1 terminieren, wenn die Anzahl der Argumente (des Argumentvektors) von der Vorgabe abweicht.

Das berechnete Ergebnis soll mit 6 signifikanten Stellen auf dem Bildschirm ausgegeben werden.

2 Zahlenraten

Programmiert das folgende kleine Spiel „Zahlenraten“:
Der Computer erzeugt eine zufällige Zahl zwischen 1 und 100. Der Spieler versucht diese Zahl per Tastatureingabe zu erraten. Nach jedem Fehlversuch gibt der Computer an, ob die gesuchte Zahl größer oder kleiner der gesuchten Zahl ist. Ist die gesuchte Zahl gefunden, so gibt das Programm die Anzahl der benötigten Versuche am Bildschirm aus und terminiert anschließend.

Eine Zufallszahl kann wie in Beispiel *randomNumber.cpp* gefunden werden.

3 Lagerhaltung

Stellt Euch vor, Ihr sollt für Eine Firma die Lagerhaltung programmieren. Stellt Euch also vor ein Lager sei dadurch gekennzeichnet, dass dort sowohl Schrauben als auch Muttern gelagert würden. Natürlich werden immer wieder Schrauben und Muttern im Lager geholt und auch immer wieder welche eingelagert. Somit ist ein Lager geradezu prädestiniert um als Klasse implementiert zu werden.

3.1 Einfache Lagerhaltung

Implementiert eine Klasse *lager*, in der die Anzahl an Schrauben und Muttern, die in einem bestimmten Lager vorhanden sind, verwaltet werden. Hierzu bedarf es natürlich einiger Variablen wie *anzahlSchrauben* oder *anzahlMuttern* und einiger Elementfunktionen wie bspw. *schraubenAbholen* oder *mutternEinlagern* (analog zur Klasse *konto* aus der Vorlesung). Außerdem soll eine Elementfunktion lagerwert geschrieben werden, die den aktuellen Wert eines Lagers ausgibt (Schrauben sollen hierfür einen Wert von 2 Cent besitzen, Muttern von 1 Cent). Erstellt nun in Eurem Hauptprogramm zwei Elemente der Klasse *lager*, initialisert für diese Objekte den Anfangsbestand an Schrauben und Muttern willkürlich und gebt den jeweiligen Lagerwert aus. Verändert nun den Lagerbestand über die Elementfunktionen und gebt den Lagerwert erneut aus.

3.2 Verbesserte Lagerhaltung

Momentan erlaubt Euer System, dass Schrauben aus Lagern geholt werden, in denen gar keine (oder zu wenige) gelagert sind. Dies soll in dieser Teilaufgabe geändert werden. Außerdem sollen Lager auch nicht beliebig aufgefüllt werden können; die maximale Kapazität an Schrauben soll in unserem Fall 100000, die an Muttern 250000 betragan. Ändert nun Eure Elementfunktionen derart, dass sie stets eine Variable vom Typ *bool* zurückliefern, der besagt, ob die gewünschte Aktion durchgeführt werden konnte oder nicht.
Viel Erfolg!