



Betreutes Programmieren 7

Aufgabe 7 ** (*Mathematische Funktionen*)

Diesmal sollen Sie einige mathematischen Bibliotheksfunktionen aus `math.h` mittels eigener Funktionen oder Makros (teilweise) nachimplementieren, in einer eigenen Übersetzungseinheit zur Verfügung stellen und in einem Hauptprogramm testen. Bei den Implementierungen dürfen dabei natürlich die entsprechenden Bibliotheksfunktionen nicht verwendet werden. Gehen Sie dabei Schritt für Schritt gemäß der nachfolgenden Beschreibung vor.

a) Implementieren Sie eine eigene Header-Datei `xmath.h` mit folgendem Inhalt:

- Definition eines Makros `xceil`, das dieselbe Funktionalität hat wie die Bibliotheksfunktion `ceil`.

Hinweise:

- Benutzen Sie geeignet den Cast-Operator.
- Die Berechnung muss nur im darstellbaren Zahlenbereich von `long int` funktionieren.
- Definition eines Makros `xfabs`, das dieselbe Funktionalität hat wie die Bibliotheksfunktion `fabs`.
- Deklaration einer Funktion `double xpow(double x, int y)`, die bzgl. der erlaubten Argumente dieselbe Funktionalität hat wie die Bibliotheksfunktion `pow` (diese sieht für `y` sogar den Datentyp `double` vor).
- Einbinden von für Fehlerbehandlungen notwendigen Bibliotheks-Kopfdateien (siehe unten).

b) Implementieren Sie eine C-Datei `xmath.c` nach folgendem Schema:

- Einbindung der Header-Datei aus der ersten Teilaufgabe.
- Implementierung der Funktion `xpow`.

Hinweise:

- Beim Aufruf `xpow(a, b)` mit `b > 0` kann dazu in einer Schleife eine Hilfsvariable, beginnend bei `1.0`, `b`-mal mit `a` multipliziert werden.
- Beim Aufruf `xpow(a, b)` mit `b == 0` ist der Rückgabewert `1.0`.
- Beim Aufruf `xpow(a, b)` mit `b < 0` ist der Rückgabewert gleich `1 / xpow(a, -b)`.

- Berücksichtigen Sie für alle Fehlerfälle die folgenden an die Dokumentation von `math.h` unter den <http://www2.hs-fulda.de/~klingebiel/c-stdlib/math.htm> angelegten Vorgaben (tatsächlich ist die Dokumentation in `math.h` allgemeiner und weniger exakt, lässt also mehr Raum für undefiniertes Verhalten - Sie sollen es hier aber etwas einfacher haben): Fehlerbehandlungen mit der globalen Variable `errno`: Die Makros `EDOM` und `ERANGE` (die man in `<errno.h>` findet), sind von Null verschiedene ganzzahlige Konstanten, mit denen Fehler im Argument- und Resultatbereich von `xpow` angezeigt werden; Ein Argumentfehler (domain error) liegt vor, wenn ein Argument nicht in dem Bereich liegt, für den `xpow` definiert ist. Bei einem Argumentfehler erhält `errno` den Wert `EDOM`; der Resultatwert ist 0. Ein Resultatfehler (range error) liegt vor, wenn das Resultat von `xpow` nicht als `double` dargestellt werden kann (Stichwort Unter- bzw. Überlauf). Bei einem Resultatfehler liefert `xpow` den Wert 0 und `errno` erhält den Wert `ERANGE`.

c) Test mit vorgegebenem Hauptprogramm:

Testen Sie nun Ihre Implementierungen mit dem vorgegebenen Hauptprogramm `bp7.c`, das jede der obigen Funktionen angewendet auf geeignete Benutzereingaben testet. Für die Verarbeitung der Benutzereingaben wird eine Quelldatei `eingabe.c` inklusive passender Header-Datei `eingabe.h` mit vorgegebenen Funktionen zur Verfügung gestellt, die dann im Hauptprogramm aufgerufen werden.

Schließlich kompilieren und verbinden Sie den in dieser Aufgabe vorgegebenen und den von Ihnen implementierten Code zu einem Programm.

Dokumentation der vorgegebenen Funktionen:

- `int leeren(void)`: Leert den Eingabestrom. Gibt bei einem Pufferfehler 0, sonst 1 zurück.
- `double einlesen(void)`: Wandelt, falls möglich, eine Benutzereingabe in einen `double`-Wert aus dem Bereich `]DBL_MIN;DBL_MAX[` um und im Erfolgsfall den umgewandelten Wert zurück. Ist komplette Umwandlung der Benutzereingabe nicht möglich, wird der Wert `DBL_MAX` zurückgeliefert. Tritt beim Einlesen ein Pufferfehler auf, wird der Wert `DBL_MIN` zurückgeliefert. Die Funktion `einlesen` sorgt bereits dafür, dass nach Beendigung immer ein leerer Puffer vorliegt.
- `void xpow_tests(void)`: Führt verschiedene Tests mit der implementierten `xpow`-Funktion durch und gibt zu jedem Test jeweils das Ergebnis und danach den aktuellen Wert von `errno` aus.