Programmieren für Physiker

Interfakultatives Institut für Anwendungen der Informatik
Institut für Theoretische Teilchenphysik

Prof. Dr. M. Steinhauser, Dr. A. Mildenberger
http://comp.physik.uni-karlsruhe.de

SS 2011 – Blatt 02
Bearbeitungszeitraum: bis 27. April 2011

Aufgabe 4: Schleifen: Produkte, Summen **Pflichtaufgabe**

a) Schreiben Sie ein C++-Programm zur Berechnung des ganzzahligen Produkts

$$P_N = \prod_{i=0}^{N-1} (3i+1) \ .$$

Verwenden Sie in diesem Falle zu Testzwecken nur ganzzahlige Datentypen. Lassen Sie den Benutzer N eingeben und berechnen Sie dann das Produkt, wobei Zwischenergebnisse auszugeben sind. Was passiert für große N?
b) Nun ist die folgende (reelle) Doppelsumme zu berechnen:

$$S_N = \sum_{i=1}^{N} \sum_{j=1}^{i} \frac{1}{i \cdot j} \ .$$

Nach Eingabe von N durch den Benutzer soll das Programm S_N ermitteln und ausgeben.

Aufgabe 5: Wann ist eigentlich Ostern? **freiwillig**

Die Kirche legt den Ostertermin als ersten Sonntag nach dem Frühlingsvollmond fest, wobei Frühlingsvollmond als erster Vollmond ab dem 21. März definiert ist. Der frühest mögliche Ostersonntag ist also der 22. März. Wie ermittelt man aber nun konkret das Datum? Anstelle langer astronomischer Berechnungen des Mond- und Sonnenzyklus gelang es Carl-Friedrich Gauß, die komplexe Periodizität in einer *Osterformel* zusammenzufassen:

$$\begin{aligned}
a &= \text{Jahr} \bmod 19 \\
b &= \text{Jahr} \bmod 4 \\
c &= \text{Jahr} \bmod 7 \\
k &= \text{Jahr} \operatorname{div} 100 \\
p &= (8k+13) \operatorname{div} 25 \\
q &= k \operatorname{div} 4 \\
M &= (15+k-p-q) \bmod 30 \\
N &= (4+k-q) \bmod 7 \\
d &= (19a+M) \bmod 30 \\
e &= (2b+4c+6d+N) \bmod 7 \\
\text{Ostertag} &= 1+(21+d+e) \bmod 31 \\
\text{Ostermonat} &= 3+(21+d+e) \operatorname{div} 31
\end{aligned}$$

Schreiben Sie ein C++-Programm, das zu einer einzugebenden Jahreszahl das Osterdatum berechnet und ausgibt. Bei ganzzahligen Operanden erhalten Sie die benötigten Rechenoperationen div und mod durch `/` und `%`.
Es sei noch erwähnt, dass es zusätzlich zwei selten zutreffende Ausnahmeregeln gibt, die sehr

späte Ostertermine etwas vorverlegen. Diese Regeln sind in obiger Osterformel nicht enthalten. Weitere Infos: z.B. deutsche Wikipedia unter „Osterdatum“.

Aufgabe 6: Genauigkeitstest **Pflichtaufgabe**

Mithilfe eines Programmes soll untersucht werden, bis zu wie kleinen $\epsilon > 0$ der Rechner die Zahlen 1 und $1 + \epsilon$ unterscheiden kann. Für sehr kleine ϵ wird eine Unterscheidung nicht mehr funktionieren, da rationale Zahlen im Rechner nicht beliebig genau dargestellt sind.

Gehen Sie in Ihrem Programm folgendermaßen vor: Starten Sie mit $\epsilon = 1$. Schreiben Sie nun eine Schleife, die testet, ob 1 und $1 + \epsilon$ verschieden sind. Solange dies der Fall ist, soll innerhalb der Schleife ϵ halbiert werden. Nach der Schleife geben Sie bitte 2ϵ aus, dies ist die Größenordnung des gesuchten Ergebnisses.

Ermitteln Sie das Ergebnis für die Datentypen `float` und `double`.

Hierarchieliste der C++ Operatoren

Operator	Prio	Asso	Bedeutung
`::`	17		Auswahl des Bezugsrahmens (binär und unär)
`[]`	16	L	Indexoperator (Feldzugriff)
`()`		L	Funktionsaufruf und Konstruktion eines Wertes
`++ --`		L	Inkrement, Dekrement (Postfix: `x++, x--`)
`.`		L	Mitgliedsauswahl eines Objekts (*object.member*)
`->`		L	Mitgliedsauswahl bei Pointer auf ein Objekt (*ptr->member*)
`!`	15	R	logische Negation
`~`		R	bitweises Komplement
`++ --`		R	Inkrement, Dekrement (Prefix: `++x, --x`)
`+ -`		R	Vorzeichen
`& *`		R	Adressoperator (*&lvalue*), Inhaltsoperator (**ptr*)
`()`		R	Typkonversion, z.B. `(long) x`
`sizeof`		R	Größe eines Objekts oder Datentyps
`new delete`		R	dynamische Speicherverwaltung
`->* .*`	14	L	Dereferenzierung eines Objektmitglieds
`* / %`	13	L	Multiplikation, Division, Modulus
`+ -`	12	L	Addition, Subtraktion (binär)
`<< >>`	11	L	bitweise Schiebe-Operatoren, auch Ein- und Ausgabe
`< <= >= >`	10	L	Vergleichsoperatoren kleiner, kleiner-gleich, größer-gleich, größer
`== !=`	9	L	Vergleich auf Gleichheit, Ungleichheit
`&`	8	L	bitweises Und
`^`	7	L	bitweises Exklusives-Oder
`|`	6	L	bitweises Oder
`&&`	5	L	logisches Und
`||`	4	L	logisches Oder
`? :`	3	R	bedingter Ausdruck (ternär)
`=`	2	R	Zuweisungsoperatoren
`+= -= *= /= %=`		R	Zuweisung mit Rechenoperation
`>>= <<= &= ^= |=`		R	
`,`	1	L	Hintereinander-Ausführung

Erklärung der Assoziativität: `a-b-c` wird `(a-b)-c` ausgewertet, weil '`-`' zunächst links bindet. Andererseits wird z.B. `a=b=c` so ausgewertet: `a=(b=c)`. Im Zweifelsfall helfen Klammern.
