Programmieren für Physiker

Interfakultatives Institut für Anwendungen der Informatik
Institut für Theoretische Teilchenphysik

Prof. Dr. M. Steinhauser, Dr. A. Mildenberger
http://comp.physik.kit.edu

SS 2011 – Blatt 10
Bearbeitungszeitraum: bis 22. Juni 2011

Aufgabe 25: Schach, Reis und Superlong **Pflichtaufgabe**

Eine Legende zur Erfindung des Schachspiels: Als der indische Erfinder seinem König das Spiel vorstellte, fand dieser so großen Gefallen daran, dass er dem Erfinder einen beliebigen Wunsch erfüllen wollte. Der Erfinder dachte nach und erbat sich für das erste Feld ein Reiskorn, für das zweite Feld zwei, für das dritte Feld vier, für das vierte Feld acht, usw. Zunächst war der König verärgert über den vermeintlich bescheidenen Wunsch, aber schließlich musste er lernen, dass er den Wunsch nicht erfüllen konnte.
Wie viele Reiskörner hätte der Erfinder erhalten sollen? Bestimmen Sie die *genaue* Anzahl der Reiskörner.
Weil selbst der Datentyp `long` auf den meisten Rechnern nicht die erforderliche Stellenanzahl besitzt, besteht die Aufgabe darin, eine Klasse `superlong` zu entwickeln, die (einige) exakte Rechenoperationen mit 50-stelligen positiven ganzen Zahlen ermöglicht. Diese Größe ist ausreichend zur Lösung der Aufgabe.
Verwenden Sie innerhalb der Klasse ein `int`-Feld fester Länge, in dem Sie die 50 Ziffern abspeichern können. Die Klasse soll drei öffentliche Mitgliedsfunktionen haben. (i) `set( int)`: eine 50-stellige Zahl mit einem Wert zu initialisieren, (ii) `add( superlong)`: eine zweite `superlong`-Zahlen zur aktuellen addieren und (iii) `print()`: Ausgabe der Zahl. Tipp zur Addition: In analoger Weise zur Rechnung „von Hand“ vorgehen und die Stellenüberträge berücksichtigen.

Folgender Hauptprogrammteil soll die Berechnung der Anzahl der Reiskörner vornehmen:

```
int main()
{   superlong sum ;  // Gesamtsumme der Reiskoerner
    sum.set(0) ;     // ... diese 0 setzen
    superlong k ;    // Reiskoerner auf dem aktuellen Feld
    k.set(1) ;       // ... startet mit 1

    for (int i=1; i<=64; ++i)
    {   cout << "Koernerzahl auf  Feld " << i << ": " ;
        k.print() ;
        sum.add(k) ; // Addiere aktuelle Kornzahl zur Summe
        k.add(k) ;   // Verdopple Kornzahl auf aktuellem Feld
    }
    cout << "Summe aller Felder: " ;
    sum.print() ;
}
```

Dieser Codeteil ist per Webseite oder im Pool unter `/home/ck11/daten/a25-sl-fragment.cc` erhältlich.

Zusatz (freiwillig): Programmieren Sie weitere Rechenoperationen mit `superlong`, z.B. eine Multiplikation.

Aufgabe 26: Acht-Damen-Problem **Pflichtaufgabe**

Auf einem Schachbrett sollen acht Damen so platziert werden, dass keine Dame eine andere Dame schlagen kann. Im Schachspiel bewegen sich Damen waagrecht, senkrecht und diagonal beliebig weit.
Das Programm soll einen sogenannten *Backtracking*-Algorithmus verwenden. Beginnen Sie mit einem leeren Spielfeld und besetzen Sie es zeilenweise nach folgendem Schema:
In der aktuellen Zeile werden nacheinander alle acht Spaltenpositionen für die neue Dame ausprobiert. Bei jedem Einzelnen dieser acht Versuche wird nun getestet, ob die neue Dame von einer bisher schon vorhandenen Dame geschlagen werden kann. Falls ja, ist diese Konstellation nicht weiter zu verfolgen. Falls nein, wird die Dame an dieser Position (temporär) vermerkt und das gleiche Verfahren zur Positionsfindung in der nächsten Zeile begonnen.
Zusätzlich ist zu berücksichtigen: Wenn eine Dame in der achten Zeile regelgerecht gesetzt werden konnte, ist eine Lösung gefunden und diese soll ausgegeben werden.

Diese Art der Lösungssuche kann am besten *rekursiv* durch eine Funktion programmiert werden, die jeweils in einer Schleife alle Spalten der aktuellen Zeile ausprobiert und sofort bei einer gültigen Positionierung sich selbst aufruft. Die zu bearbeitende Zeile soll als Funktionsargument übergeben werden. Um die Konfiguration der Damen während der Lösungssuche aufzubewahren, ist ein eindimensionales Feld zweckmäßig, in welchem zu jeder Zeile die Spaltenposition der Dame vermerkt wird.

Geben Sie die Lösungen auf dem Bildschirm aus.

Zusatzfragen (freiwillig): (i) Wieviele Lösungen gibt es insgesamt? (ii) Modifizieren Sie Ihr Programm so, dass die Anzahl der Lösungen für n Damen auf einem $n \times n$ Spielfeld berechnet wird.

Zum Knobeln: Zahlen in Summanden zerlegen **freiwillig**

Die Zahl 5 kann auf genau 7 verschiedene Arten als Summe von positiven ganzen Zahlen geschrieben werden: $(1+1+1+1+1)$, $(1+1+1+2)$, $(1+2+2)$, $(1+1+3)$, $(2+3)$, $(1+4)$, (5).
Allgemein bezeichne $p(n)$, $n \in \mathbb{N}$ die Anzahl der verschiedenen Möglichkeiten, die Zahl n als Summe von positiven ganzen Zahlen zu zerlegen. Es ist also $p(5) = 7$.
Entwickeln und programmieren Sie ein Verfahren, um $p(n)$ für ein gegebenes n zu berechnen.
Einige weitere Werte zum Testen: $p(12) = 77$, $p(34) = 12310$, $p(123) = 2552338241$.
