Programmieren für Physiker

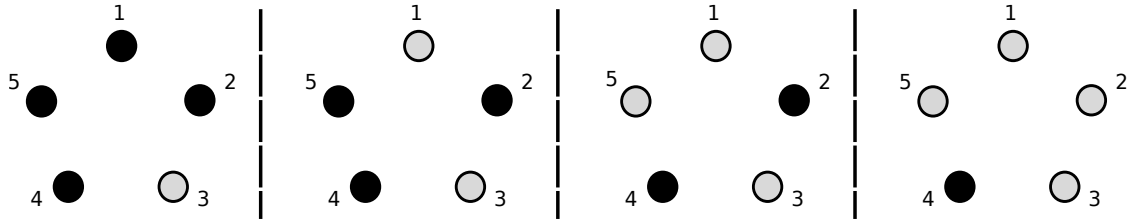
Interfakultatives Institut für Anwendungen der Informatik
Institut für Theoretische Teilchenphysik

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

SS 2018 – Blatt 06
Bearbeitungszeitraum: bis 30. Mai 2018

Aufgabe 15: Abzählreim – wer geht, wer bleibt? **Pflichtaufgabe**

Kinder ermitteln oft einen Kandidaten für ein Spiel, in dem sie sich in einem Kreis aufstellen und durch wiederholte Anwendung eines Abzählreims den Kreis so oft verkleinern, bis nur noch ein Kind übrig bleibt.
Eine formalere Beschreibung dieses Verfahrens ist: Es seien n Teilnehmer vorhanden, von 1 bis n durchnummeriert. Zur Elimination werde eine Abzählvers mit k Silben verwendet. Zu Beginn startet dieser bei Person Nummer 1. Die Person, bei der der Vers endet, scheidet aus. Bei der nächsten noch teilnehmenden Person wird der Abzählvers von vorne begonnen und wieder die k Silben unter den verbliebenen Personen durchgezählt, um die nächste auszuscheidende Person zu ermitteln. Das Verfahren wird insgesamt so lange durchgeführt, bis nur noch eine Person übrig bleibt.
Die folgende Abbildung zeigt nacheinander die Eliminationen im Fall $n = 5,\ k = 3$:



Im Beispiel der Abbildung ist demnach Nummer 4 als letzte verbleibende Person der Gewinner. Schreiben Sie ein Programm, das für einzugebendes n und k ermittelt und ausgibt, wer nacheinander ausscheidet und welche Person am Ende übrig bleibt.

Aufgabe 16: Polynom-Interpolation **Pflichtaufgabe**

Bei numerischen Berechnungen tritt gelegentlich der Wunsch nach Funktionswerten von Funktionen auf, von denen nur einige Punkte bekannt sind. Dann müssen, mittels einer Modellannahme über die Funktion, die neu zu berechnenden Werte inter- oder extrapoliert werden. Ein möglicher Weg besteht darin, für n gegebene Punkte $(x_i,\ f(x_i))$, $i = 1, \ldots, n$ ein Polynom vom Grad $n - 1$ zu bestimmen, und dieses Polynom für Vorhersagen zu verwenden.
Dies kann z.B. mittels der Interpolationsformel von Lagrange geschehen. Die Polyonome

$$L_i(x) := \prod_{\substack{k=1 \\ k \neq i}}^{n} \frac{x - x_k}{x_i - x_k}, \qquad i = 1, \ldots, n$$

sind alle vom Grad $n - 1$ und haben die Eigenschaft

$$L_i(x_j) = \delta_{ij} = \begin{cases} 1 \text{ für } i = j \\ 0 \text{ für } i \neq j \ . \end{cases}$$

Damit ergibt sich für das Interpolationspolynom der Ausdruck

$$f(x) = \sum_{i=1}^{n} f(x_i) \cdot L_i(x) = \sum_{i=1}^{n} f(x_i) \prod_{\substack{k=1 \\ k \neq i}}^{n} \frac{x - x_k}{x_i - x_k}.$$

Die Funktion $f(x)$ ist vom Grad $n-1$ und nimmt an den Stellen x_i jeweils den Wert $f(x_i)$ an.

Scheiben Sie ein C++-Programm, das diesen Interpolationsausdruck für einzugebende Stützstellen x numerisch auswertet. In der Datei `/home/ck18/daten/a16-interpol.dat` oder auf der WWW-Seite finden Sie einen Datensatz. Verwenden Sie eine `while (datei >> ...)` Schleife, um aus der Datei Wertepaare $(x_i,\ f(x_i))$ in zwei Felder einzulesen, bis das Ende der Datei erreicht ist. Der Benutzer soll nun die Möglichkeit haben, x-Werte einzugeben, für die der interpolierte Funktionswert berechnet wird. Dies soll so lange geschehen, bis der Benutzer den Wert $x = 0$ eingibt. Geben Sie dabei jeden interpolierten Wert auf dem Bildschirm aus und schreiben Sie die Werte x und $f(x)$ zeilenweise in eine Ergebnisdatei `a16-interpol-res.dat`.
Die ursprüngliche Punktedatei und die von Ihnen erzeugte Ergebnisdatei schauen Sie sich bitte in einem geeigneten Plot-Werkzeug (z.B. `python/matplotlib`, `gnuplot` oder `xmgrace`) an. Stellen Sie bitte die Ausgabe so ein, dass die gegebenen und die neu berechneten Funktionswerte durch Punkte und verschiedene Farben erkennbar sind.

Zum Knobeln: Nim-Spiel **freiwillig**

Beim Nim-Spiel in der hier betrachteten Variante gibt es zwei Spieler und zwei Häufen mit Streichhölzern auf dem Tisch. Die Spieler sind abwechselnd am Zug und haben dabei jeweils folgende zwei Möglichkeiten: entweder von genau einem Haufen ihrer Wahl eine beliebige Zahl (≥ 1) an Hölzchen zu entfernen oder aber von beiden Häufen genau die gleiche Zahl an Hölzchen zu entfernen. In jedem Zug muss also immer mindestens ein Streichholz weggenommen werden. Wer mit seinem Zug das letzte Streichholz (oder die letzten Streichhölzer) wegnehmen kann, hat gewonnen.

Schreiben Sie zunächst ein C++-Programm, welches Sie dieses Spiel mit zwei Spielern spielen lässt. In diesem Fall soll das Programm also die Streichholzhäufen verwalten und dabei von beiden Spielern abwechselnd gültige Züge erfragen, bis ein Spieler gewonnen hat.

Interessant ist es, sich zu überlegen, welche Strategie in diesem Spiel zum Gewinn führt. Gibt es Haufengrößen, von denen aus der Sieg erzwungen werden kann?
Ersetzen Sie in Ihrem Programm einen Mitspieler durch den Computer und bringen Sie ihm Ihre Gewinnstrategie bei.

Hinweis: Es ist zweckmäßig, mit zwei zufälligen Stapeln von je etwa 5 bis 25 Streichhölzern zu beginnen. Um dies zu erreichen, können Sie vom Rechner ermittelte Zufallszahlen verwenden. Nach der Anweisung `#include <cstdlib>` und `#include <ctime>` im Kopf des Programms können Sie mit `rand()%a` ganzzahlige Zufallszahlen im Intervall $[0, a[$ erzeugen. Um den Generator auf einen nicht vorherbestimmten Anfangswert zu setzen, benötigen Sie noch ein einmaliges `srand((unsigned int)time(0))` am Anfang des Programmcodes.
