

Vertiefende Zusatzaufgaben

Kaufmännisches Runden

Beim kaufmännischen Runden werden Zahlen mit Nachkommastellen in ganze Zahlen umgewandelt. Dabei werden Zahlen, deren Nachkommaanteile echt kleiner als 0.5 sind, auf die nächstniedrigere ganze Zahl abgerundet; Zahlen mit Nachkommaanteilen größer oder gleich 0.5 werden auf die nächsthöhere ganze Zahl aufgerundet.

Beispiele: 1.2 und 1.49999 werden auf 1 abgerundet; 1.5 und 1.7 werden auf 2 aufgerundet.

Schreiben Sie ein Java-Programm, das von der Tastatur eine `double`-Zahl `a` einliest und die entsprechende gerundete Zahl ausgibt.

Hinweis: Den gerundeten Wert können Sie ganz einfach dadurch berechnen, indem Sie auf `a` den Wert 0.5 aufaddieren und vom Ergebnis die Nachkommastellen abschneiden. Das Abschneiden der Nachkommastellen erledigt dabei die Methode `Math.floor()`.

Umwandlung einer Binär- in eine Dezimaldarstellung

Erarbeiten Sie ein Verfahren, mit dem eine Binärdarstellung in eine Dezimaldarstellung umgewandelt werden kann. Schreiben Sie hierfür ein Java-Programm, das folgendermaßen vorgeht:

- Einlesen der Binärdarstellung als Zeichenkette (String). Die eingegebene Zeichenkette soll eine nichtnegative Binärzahl mit 16 Ziffern darstellen, also genau 16 Zeichen umfassen, die zur Menge $\{0, 1\}$ gehören. Diese Bedingungen sollen vom Programm überprüft werden. Im Fehlerfall soll die Eingabe wiederholt werden – so oft, bis die Eingabe korrekt ist.
- Umwandlung des Strings in ein `char`-Feld (die Klasse `String` bietet dazu eine passende Methode).
- Durchlauf des Feldes, wobei aus den gespeicherten Ziffern der Wert der Zahl berechnet wird; Abspeicherung des Werts in einer Variablen vom Typ `short`.
- Ausgabe des Variablenwerts auf den Bildschirm in dezimaler Schreibweise.
- Der Benutzer soll, mit Hilfe einer Schleife, das Programm beliebig oft durchlaufen und schließlich auf Wunsch beenden können.

Hinweise:

- Beachten Sie bei der Umwandlung, dass ganz links die höchstwertige und ganz rechts die geringstwertige Ziffer steht. Benutzen Sie die Summenformel, die in der Vorlesung im Zusammenhang mit Zahlendarstellungen besprochen wurde.
- Benutzen Sie keine Funktionen aus der mathematischen Bibliothek von Java!

Primfaktorzerlegung

Schreiben Sie ein Programm, das eine `long`-Zahl einliest und sie in ihre Primfaktoren zerlegt. Die Ausgabe soll wie im folgenden Beispiel gezeigt erfolgen:

$3920 = 2^4 * 5 * 7^2$ (das Zeichen \wedge steht also für „hoch“)

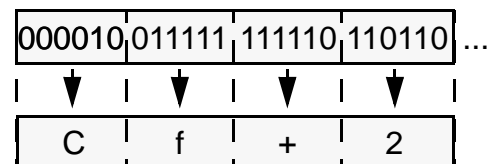
Bei einer Primzahl soll der Text „Primzahl“ ausgegeben werden.

Erweiterung: Das Programm soll zwei `long`-Zahlen `z1` und `z2` einlesen und dann die Primfaktorzerlegung für sämtliche Zahlen zwischen `z1` und `z2` bestimmen.

Base64-Codierung

Die Base64-Codierung stellt beliebige Bitfolgen durch ASCII-Zeichen dar. Sie wird beispielsweise verwendet, um Dateien mit binärem Inhalt (z.B. Bilddateien) als Mails zu verschicken, denn eine Mail nach Internet-Standard darf nur ASCII-Zeichen enthalten.

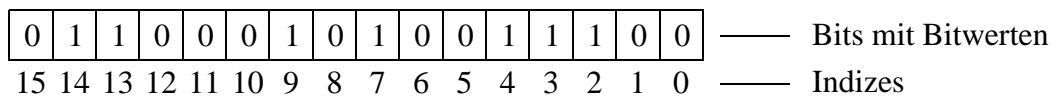
Die Codierung verläuft wie folgt: Die Bitfolge wird in Gruppen zu je 24 Bit aufgeteilt. Jede Gruppe wird dann durch vier ASCII-Zeichen dargestellt, d.h. jeweils sechs Bit werden durch ein ASCII-Zeichen codiert. Welches Bitmuster durch welches Zeichen dargestellt wird, ist durch eine Tabelle spezifiziert (siehe dazu beispielsweise Wikipedia, Stichwort „Base64“).



Schreiben Sie ein Programm, das mit Hilfe von Base64 Bitfolgen in entsprechende ASCII-Folgen umwandelt und umgekehrt.

Eine Klasse zur Darstellung von Prozessorregistern

Computerprozessoren (CPUs) besitzen Register zur Speicherung von Werten. Ein Register besteht aus einer bestimmten Anzahl von Bits, die man über einen Index einzeln adressieren kann:



Schreiben Sie ein Java-Programm, in dem eine Klasse für solche Register definiert und benutzt wird. Objekte der Klasse sollen folgende Eigenschaften haben:

- Ein `private` Attribut: Array mit `short`-Einträgen zur Aufnahme der Bitwerte.
- Zwei öffentliche Methoden:
 - Setzen eines bestimmten Bitwerts:
Zwei Parameter: Index des Bits, das gesetzt werden soll; neuer Wert für dieses Bit.
Rückgabewert: 0 für ok, -1 im Fehlerfall (wenn der Indexparameter außerhalb des zulässigen Bereichs liegt oder wenn der Bitwertparameter nicht 0 oder 1 ist).
 - Ausgabe des Arrayinhalts auf den Bildschirm (kein Parameter, kein Rückgabewert).

- Ein Konstruktor: Der Konstruktor soll einen short-Parameter übergeben bekommen, der die gewünschte Länge des Registers angibt. Er soll den Array erzeugen und mit Nullen vorbe-setzen.

Das Hauptprogramm soll folgendes tun:

- Erzeugung eines Registerobjekts. Dabei soll die Arraylänge über die Tastatur eingelesen und dann an den Konstruktor übergeben werden.
- Ausgabe des Registerinhalts auf den Bildschirm.
- Einlesen eines Index- und eines Bitwerts von der Tastatur, anschließend Zuweisung dieses Bitwerts an das entsprechende Registerbit.
- Erneute Ausgabe des Registerinhalts auf den Bildschirm.

Beachten Sie, dass alle Tastatureingaben im Hauptprogramm, also nicht in den Objektmethoden selbst vorgenommen werden sollen!

Änderungen an einer Bilddatei

Eine Bitmap-Bilddatei kann die Daten eines Bilds entweder komprimiert oder unkomprimiert speichern. Eine unkomprimierte Datei enthält die Daten byteweise wie folgt:

- Die ersten 54 Byte sind der Header (Kopf) der Datei, der allgemeine Informationen enthält.
- Anschließend folgen die Daten der letzten Bildzeile: Jeder Bildpunkt („Pixel“) wird durch genau drei Byte codiert, die jeweils den Rot-, Grün- und Blau-Anteil seiner Farbe angeben. Die Pixel der Zeile sind dabei von links nach rechts angeordnet, wobei jeweils als erstes der Blauwert, dann der Grünwert und zuletzt der Rotwert des Pixels gespeichert ist. Die Daten der Zeile werden durch zwei Byte abgeschlossen, die den Wert 0 enthalten.
- Danach folgen die Daten der vorletzten Bildzeile, die auf dieselbe Weise strukturiert sind, danach die Daten der drittletzten Zeile usw.

Schreiben Sie ein Programm, das die Rot-, Grün- und/oder Blau-Anteile aller Pixel eines Bilds auf 0 setzt. Das Programm soll wie folgt vorgehen:

- Einlesen der Daten einer Bitmap-Datei in ein Array des Komponententyps „byte“.
- Frage an den Benutzer, wie viele Zeilen die Bilddatei enthält und wie viele Pixel pro Zeile vorhanden sind.
(Wenn Sie möchten, können Sie Ihr Programm auch so gestalten, dass es diese Informationen aus dem Header der Datei bezieht. Details zum Aufbau des Headers finden Sie beispielsweise unter www.wikipedia.de, Stichwort „Windows bitmap“).
- Frage an den Benutzer, welcher Farbanteil oder welche Farbanteile auf 0 gesetzt werden sollen.
- Durchführung der Farbänderungen auf dem Array.
- Übertragung der Daten aus dem Array in eine neue Datei.

Testen Sie Ihr Programm an den Bitmap-Dateien, die Sie mit einem Browser von <http://www.nt.fh-koeln.de/vogt/dv/file1.bmp> und [../file2.bmp](http://www.nt.fh-koeln.de/vogt/dv/file2.bmp) herunterladen können (nicht wundern: Das erste Bild ist tatsächlich rein weiß!). Beide Dateien enthalten 584 Zeilen mit je 822 Pixeln. Schauen Sie sich dazu insbesondere das Ausgangsbild und das resultierende Bild mit einem Bildbetrachtungsprogramm (z.B. Irfanview) an.

Literaturhinweis

Eine Fülle weiterer Aufgaben findet man in:

Reinhard Schiedermeier, Klaus Köhler

Das Java-Praktikum – Aufgaben und Lösungen zum Programmierenlernen
dpunkt-Verlag, 2008