PROGRAMMIERUNG MIT C++ 1

Aufgabenblatt 3

Aufgabe 1. *ASCII Tabelle* 70 %

Schreiben Sie ein C-Programm, das eine vierspaltige ASCII-Tabelle ausgibt, die jeweils den oktalen, dezimalen und hexadezimalen Zahlenwert eines Zeichens (dreistellig mit führenden Nullen) sowie das zugehörige abdruckbare Zeichen enthält. Anstelle der nicht darstellbaren Steuerzeichen (0-31, 127) sollen drei Punkte ausgegeben werden.
Die Ausgabe soll wie folgt aussehen:

```
*** ASCII Tabelle ***
Okt Dez Hex Zch | Okt Dez Hex Zch | Okt Dez Hex Zch | Okt Dez Hex Zch
-----------------------------------------------------------------
000 000 000 ... | 040 032 020      | 100 064 040  @  | 140 096 060  `
001 001 001 ... | 041 033 021  !   | 101 065 041  A  | 141 097 061  a
002 002 002 ... | 042 034 022  "   | 102 066 042  B  | 142 098 062  b
003 003 003 ... | 043 035 023  #   | 103 067 043  C  | 143 099 063  c
004 004 004 ... | 044 036 024  $   | 104 068 044  D  | 144 100 064  d
...
035 029 01d ... | 075 061 03d  =   | 135 093 05d  ]  | 175 125 07d  }
036 030 01e ... | 076 062 03e  >   | 136 094 05e  ^  | 176 126 07e  ~
037 031 01f ... | 077 063 03f  ?   | 137 095 05f  _  | 177 127 07f ...
```

Hinweise:

a) Wie Sie in der Ausgabe erkennen, werden die ASCII-Codes in 4 Spalten zu je 32 Werten ausgegeben. Versuchen Sie zunächst die Zahlen von 0 bis 127 in folgender Form auszugeben:
```
000 032 064 096
001 033 065 097
...
```
b) Ergänzen Sie dann die Oktal- und Hexadezimaldarstellung der Zahlen

c) Ergänzen Sie abschließend die ASCII-Zeichen der Codes. Zur Überprüfung, ob sie ein Zeichenwert drucken können (32-126) oder nicht (0-31, 127) benötigen Sie eventuell eine sogenannte Fallunterscheidung. Dazu können Sie die `if`-Anweisung benutzen. Im unten stehenden Beispiel sehen Sie eine `if`-Anweisung mit zugehörigem Alternativteil (`else`-Teil). Wird die `BedingungA` als *wahr* (d.h. ungleich Null) ausgewertet, so werden `Anweisung1` und `Anweisung2` ausgeführt. Andernfalls werden `Anweisung3` und `Anweisung4` ausgeführt. Der `else`-Teil ist dabei optional, d.h. er kann auch weggelassen werden.

```
if (BedingungA) {
    Anweisung1;
    Anweisung2;
} else {
    Anweisung3;
    Anweisung4;
}
```

Aufgabe 2. *Genauigkeit von float und double* 30 %

Entwickeln Sie ein C-Programm, das die Wertentwicklung eines Sparkontos mittels einer Schleife berechnet. Auf dem Konto soll ein Startbetrag von €10000.00 über einen Zeitraum von 50 Jahren mit einem Zinssatz von 3.13% angelegt werden. Der Geldbetrag erhöht sich also jedes Jahr um das 1.0313-fache. Als Resultat soll das Programm den Kontostand mit mehreren Nachkommastellen ausgeben. Schreiben Sie 2 Versionen Ihres Programms. In der ersten benutzen Sie für alle Fließkomma-Variablen und -Konstanten `double` Typen, in der zweiten Version ausschließlich `float`. Was fällt Ihnen am Ergebnis auf?
Welches der beiden Ergebnisse ist *korrekt*?

Hinweis: Eine *Kommazahl*-Konstante in der Form `1.23` hat immer den Typ `double`. Wenn Sie in einem Ausdruck eine solche Konstante verwenden, wird der gesamte Ausdruck als `double` berechnet, also etwas genauer. Wenn Sie eine *Kommazahl* explizit als `float` angeben wollen, hängen Sie ein `f` an die Zahl an, also etwa so: `1.23f`. Sind alle Werte in einem Ausdruck vom Typ `float`, so wird der Ausdruck auch nur als `float` berechnet.

Freiwillige Erweiterung: Erweitern Sie das Programm so, dass Startbetrag, Zinssatz und Zeitraum über die Kommandozeile vom Benutzer eingegeben werden können.