4 Zahlendarstellungen

4.1 Verschiedene Zahlensysteme

Lesen Sie eine natürliche Zahl ein und geben Sie diese als Dezimal-, Oktal- und Hexadezimal-Wert aus. Hinweis: Die eingelesene Zahl muss hierfür nicht verändert werden.

4.2 Dezimal-Zahl einlesen und als Binär-Zahl ausgeben

Schreiben Sie eine Funktion namens print_bit(?), der Sie einen Wert übergeben und die dann den Zahlenwert in binärer Form auf dem Konsolenfenster darstellt. Fügen Sie nach jedem 4. Bit ein Leerzeichen ein.

4.3 Binär-Zahl einlesen und als Dezimal-Zahl ausgeben

Lesen Sie eine Zahl in binärer Form ein und wandeln Sie diese in eine dezimale Integer-Zahl um. Berücksichtigen Sie eine Speicherbreite von 16 Bit. Wie behandeln Sie das Vorzeichen?

4.4 Anzahl auf 1 gesetzter Bits ermitteln

Schreiben Sie eine Funktion namens bits(?), der Sie einen Wert übergeben und die dann die Anzahl der auf „1“ gesetzten Bits zurückliefert. Die Zahl 5 besitzt 2 auf „1“ gesetzte Bits, denn sie wird binär als 0101 dargestellt.

4.5 Zahl auf gerade / ungerade überprüfen

Schreiben Sie eine Funktion is_even(?), die einen ihr übergebenen Zahlenwert überprüft, ob dieser gerade oder ungerade ist. Die Funktion liefert einen wahren Wert („1“) zurück, wenn die Zahl gerade ist, und einen falschen Wert („0“), wenn die Zahl ungerade ist. Lösen Sie die Aufgabe zunächst mit dem Modulo-Operator. Überlegen Sie sich einen zweiten Lösungsweg unter Einsatz logischer Operationen. Gibt es weitere Möglichkeiten, den Wert zu überprüfen?

4.6 Zahl auf Teilbarkeit prüfen

Schreiben Sie eine Funktion, die überprüft, ob eine ihr übergebene Zahl durch 6 teilbar ist. Die Funktion liefert einen wahren Wert („1“) zurück, wenn die Zahl durch 6 teilbar ist, und einen falschen Wert („0“), wenn die Zahl nicht durch 6 teilbar ist. Benutzen Sie keinen Modulo-Operator.