The GAME Engineers

#3 – Komplemente und Kontrollstrukturen

Inhalt

- Komplemente
  - Einskomplement
  - Zweikomplement

- Kontrollstrukturen in Java
  - Verzweigungen
  - Schleifen

Komplemente – negative Zahlen?

- Wie stellt man negative Binärzahlen dar?

  - mathematisch nach Stellenwertsystem?
    → für uns uninteressant (ist ja nur Vorzeichen – davorsetzen)

  - im Computer?
    → für uns sehr interessant (Weil wie verarbeiten Rechner das?)

Negative Zahlen im Computer

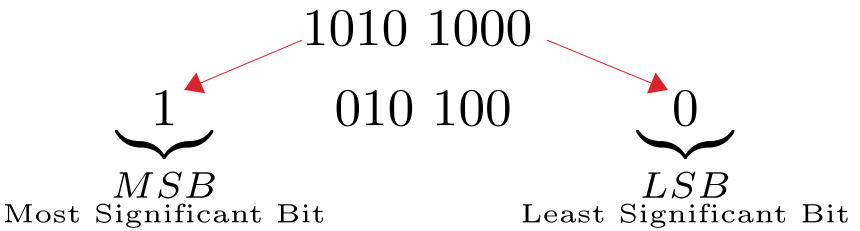
Ideen?

Komplemente – Einskomplement

- Einskomplementoperator
  - Auch Bitweise-Negation genannt: ~

$$\sim 1001 = 0110$$
$$\sim 0111 = 1000$$
$$\sim 1011\ 0010 = 0100\ 1101$$

- Wie interpretiere ich das jetzt allerdings für negative Zahlen?
  - MSB 0 → positiv
  - MSB 1 → negativ



Komplemente – Einskomplement

- MSB ist also Vorzeichenbit
- Außerdem wird definiert:

$$0000_2 = 0_{10}$$
$$1000_2 = -7_{10}$$

→ Allgemein: Stellenwertigkeit des höchstwertige Bit minus 1

$$1000\ 0000_2 = -2^7 - 1 = -127$$

- Die nachfolgenden Einsen werden dazu addiert:

$$1010_2 = -5_{10}$$
$$\underset{1000_2}{-7} + \underset{010_2}{2} = -5$$

$$0101_2 = 0 + 4 + 1 = +5_{10}$$
$$0111_2 = 0 + 4 + 2 + 1 = +7_{10}$$

$$1101_2 = -7 + 4 + 1 = -2_{10}$$
$$1111_2 = -7 + 4 + 2 + 1 = 0_{10}$$

Bzw. -0_{10} da führende 1 negatives Vorzeichen

$$0000_2 = +0_{10}$$
$$1111_2 = -0_{10}$$

Wertebereich (4 Bit):
-7 bis +7

Komplemente – Zweikomplement

- MSB ist auch Vorzeichenbit
- Außerdem wird definiert:

$$0000_2 = 0_{10}$$
$$1000_2 = -8_{10}$$

- Die nachfolgenden Einsen werden dazu addiert:

$$1010_2 = -6_{10}$$
$$\underset{1000_2}{-8} + \underset{0010_2}{2} = -6$$

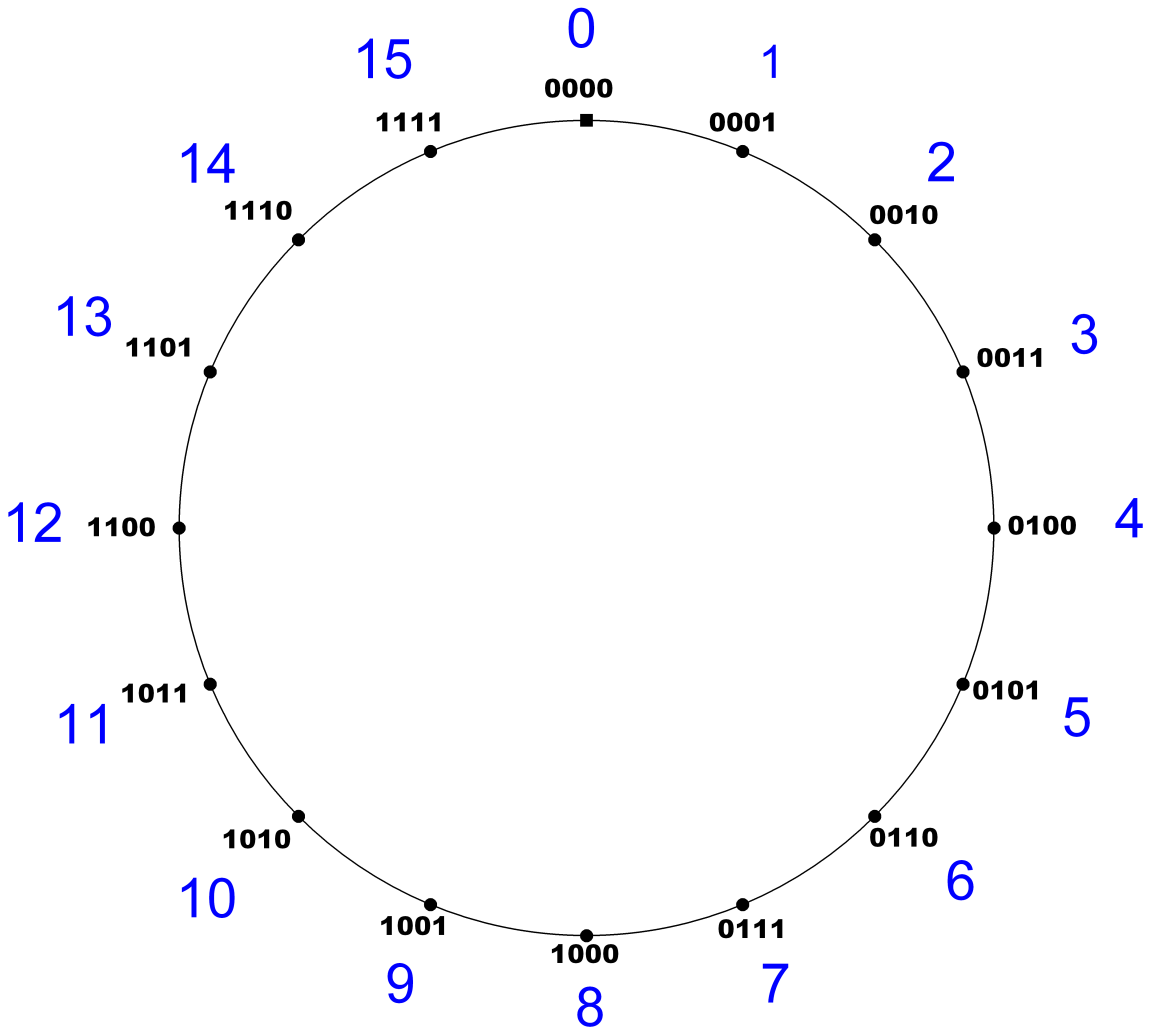
$$0101_2 = 0 + 4 + 1 = +5_{10}$$
$$0111_2 = 0 + 4 + 2 + 1 = +7_{10}$$

$$1101_2 = -8 + 4 + 1 = -3_{10}$$
$$1111_2 = -8 + 4 + 2 + 1 = -1_{10}$$

$$0000_2 = 0_{10}$$
$$1111_2 = -1_{10}$$

Wertebereich (4 Bit):
-8 bis 7

Komplemente – Zweikomplement



Komplemente – Zweikomplement

- Allgemeiner Wertebereich
  für n Bit: -2^{n-1} bis $2^{n-1}-1$

32 Bit:

$$-2^{31} = 2\ 147\ 483\ 648$$
$$2^{31}-1 = 2\ 147\ 483\ 647$$

Datentyp	Größe	Wertebereich	Beschreibung
boolean	1 bit	True/False	Boolescher Wahrheitswert
char	16 bit / 2 byte	0 bis 65.535	Unicode-Zeichen (UTF-16) z.B.: A,B,C
byte	8 bit / 1 byte	- 128 bis 127	Ganzzahliger Wert
short	16 bit / 2 byte	-32.768 bis 32.767	Ganzzahliger Wert
int	32 bit / 4 byte	- 2.147.483.648 bis 2.147.483.647	Ganzzahliger Wert
long	64 bit / 8 byte	-2^{63} bis $2^{63}-1$, ab Java 8 auch 0 bis $2^{64}-1$	Ganzzahliger Wert

Komplemente – Zweikomplement

- Lass mal rechnen (5 Bit)!
  - Geg.: 15 und -12
  - z.R.: 15 + (-12) = 3

$$15_{10} = 0\ 1111_2 \qquad -12_{10} = 1\ 0100_2$$
$$3_{10} = 0\ 0011_2$$

$$\begin{array}{r} 01111 \\ +\ 10100 \\ \hline = \ 00011 \end{array}^{111}$$

Aber was ist, wenn wir +15 und +12 gegeben haben und 15 - 12 rechnen wollen?

Komplemente – Zweikomplement

- Wie bilde ich Zweikomplement einer Zahl?
  - Einskomplement bilden
  - 1 addieren

$$12_{10} = 0\ 1100_2$$

$$\sim 0\ 1100_2 = 1\ 0011_2$$

$$\begin{array}{r} 10011 \\ +\ 00001 \\ \hline = \ 10100 \end{array}$$

$$\begin{array}{r} {\scriptstyle 1\quad 11} \\ \end{array}$$

$$10100_2 = -12_{10}$$

Kontrollstrukturen

Kontrollstrukturen sind Anweisungen in Programmiersprachen, die verwendet werden um den Ablauf eines Computerprogramms zu steuern.

Verzweigungen

Kontrollstrukturen - Verzweigungen

If-Statement (Einfache Verzweigung):

```
if (boolscherAusdruck)
{
  // Anweisung(en) 1
}
```

```
int x = 3;
if (x > 5) {
  System.out.println("Zahl ist größer als 5");
}
```

Kontrollstrukturen - Verzweigungen

If-Else-Statement (Einfache Verzweigung):

```
if (boolscher Ausdruck) {
  // Anweisung(en) 1
} else {
  // Anweisung(en) 2
}
```

```
int x = 3;
if (x > 5) {
  System.out.println("Zahl ist größer als 5");
} else {
  System.out.println("Zahl ist nicht größer als 5");
}
```

Kontrollstrukturen - Verzweigungen

„Vereinfachtes“ If-Else-Statement (Tenärer Operator):

```
(boolscher Ausdruck) ? AusgaewertTrue : AusgaewertFalse;
```

```
int x = 3;
if (x > 5) {
  System.out.println(“Zahl ist größer als 5“);
} else {
  System.out.println(“Zahl ist nicht größer als 5“);
}
```

```
int x = 3;
(x > 5)? System.out.println(“Zahl ist größer als 5“) : System.out.println(“Zahl ist nicht größer als 5“);
```

Kontrollstrukturen - Verzweigungen

Mehrere Wege ?:

```
if (boolscher Ausdruck) {
  // Anweisung(en) 1
}
if (boolscher Ausdruck) {
  // Anweisung(en) 2
}
if (boolscher Ausdruck) {
    // Anweisung(en) 3
} else {
  // Anweisung(en) 4
}
```

```
if (boolscher Ausdruck) {
  // Anweisung(en) 1
} else {
  if (boolscher Ausdruck) {
    // Anweisung(en) 2
  } else {
    if (boolscher Ausdruck) {
      // Anweisung(en) 3
    } else {
      // Anweisung(en) 4
    }
  }
}
```

Praktisch?

Kontrollstrukturen - Verzweigungen

Mehrfachverzweigung (Switch-Case):

```
switch (variable) {
  case prüfwert1 :
          // Anweisung(en) 1
          break;
  case prüfwert2 :
          // Anweisung(en) 2
          break;
  case prüfwert3 :
          // Anweisung(en) 3
          break;
  default :
          // Standardanweisungen
}
```

Kontrollstrukturen - Verzweigungen

Mehrfachverzweigung (Switch-Case):

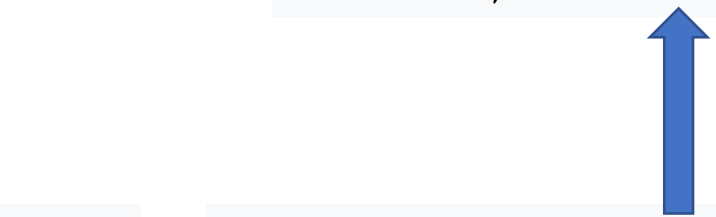
```
int x = 3;
switch (x) {
  case 0 :
          System.out.println("Mein Wert ist 0.");
          break;
  case 1 :
          System.out.println("Mein Wert ist 1.");
          break;
  case 2 :
          System.out.println("Mein Wert ist 1 oder 2.");
          break;
  default :
          System.out.println("Ich habe einen anderen Wert.");
}
```

Schleifen

Kontrollstrukturen - Schleifen

For-Schleife (Zählschleife)

```
i ++;          i --;
i = i + 1;     i = i - 1
```



```
for(Variable; boolscher Ausdruck; Rechenoperation){
  System.out.println( i );
}
```

```
for( int i=0; i < 3; i++ ){
  System.out.println( i );
}
```

Ausgabe:

```
0
1
2
```

Kontrollstrukturen - Schleifen

While-Schleife (Vorbedingung)

```
while (boolscher Ausdruck) {
   // Anweisung(en) 1
}
```

1. **Überprüfung**
2. **Anweisung(en)**
3. …


```
int x = 2;
while (x < 5) {

System.out.println(x);
   x++;
}
```

Ausgabe:

```
2
3
4
```

```
int x = 2;
while (x < 5) {
   x++;

System.out.println(x);
}
```

Ausgabe:

```
3
4
5
```

Kontrollstrukturen - Schleifen

Do-While-Schleife (Nachbedingung)

```
do {
  // Anweisung(en) 1
} while (boolscher Ausdruck);
```

```
1. Anweisung(en)
2. Überprüfung
3. …
```

```
int x = 2;
do {
  System.out.println(x);
  x++;
} while (x < 5);
```

Ausgabe:

```
2
3
4
```

Kontrollstrukturen - Schleifen

While vs Do-While

```
int x = 10;
while (x < 10) {

System.out.println(x);
  x++;
}
```

Ausgabe:

```
int x = 10;
do {
  System.out.println(x);
  x++;
} while (x < 10);
```

Führt die Anweisungen mindestens 1 mal aus!

Ausgabe:

```
10
```

Ende

Fragen?