Programmieren 2 **Übungszettel 2** **SS2010**

Aufgabe 1) Schnelle Division durch Bit-Operationen

Gegeben sei eine beliebige nicht negative Ganzzahl. Wir wollen jetzt wissen, ob diese Zahl nach einer Ganzzahl-Division durch 16 noch zusätzlich durch 4 teilbar ist. Aus Laufzeitgründen soll die Division durch 16 aber nicht ausgeführt werden!
Realisieren Sie dies mittels einer Funktion `div4nach16`. Überprüfen Sie im Hauptprogramm durch Nachrechnen, ob diese Funktion auch korrekt arbeitet.
Verallgemeinern Sie die Funktion zu `divNnach16(zahl,n)`. Es wird überprüft, ob die Zahl nach einer Ganzzahl-Division durch 16 noch durch 2^n teilbar (1<=n<=10) ist.

Aufgabe 2) Datentypen für Bit-Ketten

Entwerfen Sie einen Datentypen BitVektor (struct bitvector), der zur Aufnahme beliebig langer Bit-Ketten dient. Stellen Sie für diese Struktur die folgenden Funktionen zur Verfügung:

- `struct bitvector *createBitVector(unsigned bits);`
  erzeugt einen `bits`-breiten BitVektor
- `void setBit(struct bitvector *bv, unsigned bitNr);`
  setzt das in `bitNr` angegebene Bit auf 1
- `void resetBit(struct bitvector *bv, unsigned bitNr);`
  setzt das in `bitNr` angegebene Bit auf 0
- `struct bitvector *getSlice(struct bitvector *bv, unsigned pos, unsigned n);`
  liefert einen neuen BitVektor der Breite `n`, der `n` Bits des übergebenen Vektors von der Position `pos` an abwärts enthält (vgl. Funktion aus der Vorlesung)
- `struct bitvector *addBitVector(struct bitvector *s1, struct bitvector *s2);`
  führt eine Addition der beiden Bitvektoren durch und erzeugt einen neuen Bitvektor mit dem Ergebnis. Achten Sie insbesondere auf durch Überläufe erzeugte Verbreiterungen des Vektors
- `char *printBitVector(struct bitvector *bv);`
  druckt den Inhalt des Bitvektors in ein Feld von `char`, so dass sie dies anschließend mit `cout` darstellen können. Achten Sie insbesondere auf die Reihenfolge der Bits. Zwecks besserer Übersicht bietet es sich an, zwischen Nibbles (4 Bit=eine Hex-Ziffer) ein Leerzeichen einzuschieben.
- Wenn Sie möchten, dürfen sie auch noch Shift-Operationen implementieren, äquivalent zu << und >>. Achten Sie dabei insbesondere auf die ungenutzten Bits im Speicher im Falle von Bitbreiten, die nicht durch 8 teilbar sind.

Bedenken Sie beim Entwurf die Problematiken der Größen von elementaren Datentypen und der Byte-Order auf Ihrer Maschine.