Kapitel 2: Programmfluss (Lösungen)

LÖSUNG ZU AUFGABE 04

```
void act() {
  // Wiederholen, solange nicht auf einem Blatt.
  while (!onLeaf()) {

    if (treeFront()) {
      goAroundTree();
    } else {
      move();
    }
  }

  // Am Schluss noch das Blatt auflesen.
  removeLeaf();
}

void goAroundTree() {
  turnLeft();
  move();
  turnRight();
  move();
  move();
  turnRight();
  move();
  turnLeft();
}
```

LÖSUNG ZU AUFGABE 05

```
void act() {
  // Wiederholen, solange nicht links und rechts ein Baum steht.
  while (!(treeLeft() && treeRight())) {
    move();
  }

  putLeaf();
}
```

LÖSUNG ZU AUFGABE 06

```
void act() {
  // Wiederholen, solange nicht auf einem Blatt.
  while (!onLeaf()) {

    if (treeLeft() || treeRight()) {
      putLeaf();
    }

    move();
  }
}
```

LÖSUNG ZU AUFGABE 07

```
void act() {
  // Wiederholen, solange nicht vor einem Baum.
  while (!treeFront()) {
    if (!onLeaf()) {
      putLeaf();
    }
    move();
  }

  // Das letzte Blatt nicht vergessen.
  if (!onLeaf()) {
    putLeaf();
  }
}
```

LÖSUNG ZU AUFGABE 08

```
void act() {
  for (int i = 0; i < 1000; i++) {
    makeOneStep();
  }
}

void makeOneStep() {
  if (!treeRight()) {
    // no tree right --> go right
    turnRight();
    move();
  } else {
    // there is a tree right
    if (!treeFront()) {
      // no tree in front --> move
      move();
    } else {
      // trees right and front
      if (!treeLeft()) {
        // no tree left --> go left
        turnLeft();
        move();
      } else {
        // trees right, front and left: dead end
        turnLeft();
        turnLeft();
        move();
      }
    }
  }
}
```