

Aufgabe 6

Dennis Blöte, 08.12.2006

Gedichtinterpretation „Worte sind Schatten“

Das Gedicht „Worte sind Schatten“ wurde 1969 von Eugen Gomringer geschrieben und befasst sich auf spielerische Art und Weise mit Worten und ihrer Bedeutung. Gomringer weist Worten verschiedene Bedeutungen zu, auf denen aufbauend er Vergleiche anstellt, welche ihrerseits wiederum zu einer neuen Bedeutungszuweisung führen.

Meine erste Deutung des Gedichts wäre, dass er seine aufgestellte These „Worte sind Spiele“ durch die Form des Gedichts unterstreicht: Die Anordnung seiner Worte in den knappen Sätzen spielt mit der Syntax deutscher Grammatik. Auf der anderen Seite entstehen durch die immer neuen Zuweisungen Verwirrungen – möglicherweise bezeichnet er dies als „Schatten“.

Das Gedicht besteht aus sechs Strophen mit jeweils zwei Versen, in denen immer fünf verschiedene Worte (Worte, sind, Schatten, werden, Spiele) zu jeweils unterschiedlichen Sätzen aneinandergereiht werden. Jeder dieser Verse enthält drei der Worte. Auffällig dabei ist, dass das Wort „Worte“ in jeder Zeile vorkommt und die „Worte“ innerhalb der Strophe immer diagonal angeordnet sind.

Der Satzbau der ersten vier Verse (Strophe 1 und 2) folgt gemäß der Stellung Subjekt, Prädikat, Objekt – die letzten acht Verse (Strophe 3 bis 6) haben den Aufbau Prädikat, Subjekt, Objekt.

Beim Versmaß wechseln sich Jambus und Trochäus ab: Die erste Verszeile einer Strophe fängt immer mit einer unbetonten Silbe an, auf die eine betonte folgt (Jambus) – in der zweiten Verszeile ist es anders herum (Trochäus). Das Gedicht verfügt über keinen Endreim, wohl aber über Stabreime (bspw. „werden Worte“).

Insgesamt hinterlässt das Gedicht einen stimmigen Eindruck. Die Strophen sind in sich schlüssig und klingen sehr harmonisch. Das gesamte Gedicht ist in sich abgeschlossen, jedoch bleibt die Frage offen, wie die Worte „Schatten“ und „Spiele“ zueinander in Bezug stehen, da sie nie direkt aufeinander bezogen werden. Es wird kein Vergleich zwischen ihnen angestellt und es finden auch keine direkten Zuweisungen unter den beiden Worten statt. All das wird allein über die „Worte“ gemacht, welche dadurch zum zentralen Bestandteil des Gedichts werden.

Lässt sich ein Bezug der Gedichte zu unserem Kurs aufstellen?

Meiner Meinung nach lassen sich über mehrere Ansätze Bezüge des Gedichts zu unserem Kurs anstellen: Wir beschäftigen uns mit Programmen und Prozessen - in der Beschreibung „Allgemeines zum Kurs“ heißt es dazu:

„Programm, kann man sagen, ist statischer Text. Prozess ist dann dynamische Operation. [...] In dieser Denkrichtung bedeutet Programme zu schreiben, Texte zu schreiben, die gleichzeitig Maschinen sind. Die wie Maschinen sind, wäre wohl etwas vorsichtiger gesagt. Programme sind exekutierbare Texte.“

Das Gedicht ist statischer Text, welcher sich in Programmform auch ausführbar niederschreiben ließe:

worte sind schatten schatten werden worte		a = b b = a
worte sind spiele spiele werden worte		a = c c = a
sind schatten worte werden worte spiele	→	if b == a a = c
sind spiele worte werden worte schatten		if c == a a = b
sind worte schatten werden spiele worte		if a == b c = a
sind worte spiele werden schatten worte		if a == c b = a

Die Worte sind nicht willkürlich aneinandergereiht, sondern folgen einer bestimmten Syntax, nämlich der deutschen Grammatik. Dennoch wechseln die Worte im Laufe des Gedichts ihre Positionen und werden miteinander vertauscht. Auch dies ist ein bekanntes Muster aus der Informatik, welches als Permutation bezeichnet wird.

Durch die Stellungen nach dem Muster Subjekt, Prädikat, Objekt ergeben sich in diesem Fall durch die Wörter „sind“ und „werden“ Zuweisungen, wie wir sie aus Programmen kennen. Die letzten vier Strophen des Gedichts ließen sich aus dieser Sicht als Abfragen, bzw. Bedingungen deuten.

Definieren wir den Prozess als eine Abfolge von Zuständen eines Systems, so lässt sich auch dies im Gedicht wiederfinden: Die durch Nomen definierten Variablen bekommen Bedeutungen zugewiesen und ändern dadurch ihren Zustand. Das Gedicht könnte somit auch als Programmablauf bezeichnet werden.

Wie ließe sich ein Programm entwerfen, welches solche Texte schreiben kann?

Die Frage ließe sich unterschiedlich beantworten, je nachdem ob man den Schwerpunkt der Frage auf die Form oder den Inhalt des Gedichtes legt.

Ein Programm zu schreiben, welches eigenständig kreative und sinnvolle Gedichte als Ausgabe liefert, scheint mir nicht möglich zu sein. Der Computer ist eine verarbeitende Maschine, die in Zuständen „denkt“, jedoch keine Bedeutungen kennt und logische Zusammenhänge nur auf einer sehr niedrigen Ebene interpretieren kann. Dass sich automatisiert Gedichte schreiben ließen, die sowohl grammatische als auch inhaltliche Ästhetik haben, kann ich mir nicht vorstellen.

Etwas anderes wäre es, wenn es darum ginge, ein Programm zu entwickeln, welches die Form bzw. den Aufbau eines solchen Gedichtes nachempfinden soll und ähnliche Ausgaben machen kann: Im Falle des zweiten Gedichts (silencio) wäre dies sehr einfach zu realisieren - jedoch nur rein formal, denn die Form leitet sich vom Inhalt ab, welchen der Computer nicht verstehen kann. Hier müsste man davon ausgehen, dass das Gedicht mit einem anderen Wort auch einen anderen Aufbau hätte, da die Form nur den Inhalt unterstützt.

Um im Fall von „worte sind schatten“ ein Programm zu schreiben, welches ähnliche Ergebnisse liefert, müsste man eine Struktur erzeugen, welche die Form des Gedichtes - die deutsche Grammatik - nachempfindet.

Als Eingabeparameter würde das Programm fünf verschiedene Worte erwarten: Drei Nomen und zwei Verben, welche sich aufeinander Bezug nehmend verknüpfen lassen (damit es einen Sinn ergibt, müssten bspw. sowohl Nomen als auch Verben in Pluralform sein). Diese Struktur müsste bei der Eingabe als Voraussetzung gegeben sein, da das Programm solche Überprüfungen nicht tätigen kann.

Ich habe auf zweierlei Arten versucht, zum Erfolg zu kommen: Der erste (naive) Ansatz war es, eine Schablone zu verwenden, in welche Worte eingesetzt werden können. Dabei handelte es sich lediglich um eine stumpfe Ersetzung der einzelnen Nomen und Verben.

Der zweite Ansatz war es, die Form durch Programmierung nachzubilden. Das funktioniert für die ersten beiden Strophen noch recht gut, müsste aber für die restlichen vier Strophen um eine Reihe zusätzlicher Regeln erweitert werden, was das Ganze recht starr werden lassen würde.

Letztendlich steckt hinter beiden meiner Ansätze der Versuch, die Struktur möglichst fest vorzugeben - dadurch ließe sich zwar die Form nachbauen, jedoch würde inhaltlich kein neues Meisterwerk entstehen ;)

Naiver Ansatz - Schablone

```
class PoesieMaschine
  # naiver Ansatz
  def PoesieMaschine.gedicht(nomen, verben)
    # Strophe 1
    puts nomen[1] + " " + verben[1] + " " + nomen[2]
    puts nomen[2] + " " + verben[2] + " " + nomen[1] + "\n\n"
    # Strophe 2
    puts nomen[1] + " " + verben[1] + " " + nomen[3]
    puts nomen[3] + " " + verben[2] + " " + nomen[1] + "\n\n"
    # Strophe 3
    puts verben[1] + " " + nomen[2] + " " + nomen[1]
    puts verben[2] + " " + nomen[1] + " " + nomen[3] + "\n\n"
    # Strophe 4
    puts verben[1] + " " + nomen[3] + " " + nomen[1]
    puts verben[2] + " " + nomen[1] + " " + nomen[2] + "\n\n"
    # Strophe 5
    puts verben[1] + " " + nomen[1] + " " + nomen[2]
    puts verben[2] + " " + nomen[3] + " " + nomen[1] + "\n\n"
    # Strophe 6
    puts verben[1] + " " + nomen[1] + " " + nomen[3]
    puts verben[2] + " " + nomen[2] + " " + nomen[1] + "\n\n"
  end
end
```

```
PoesieMaschine.gedicht(["worte", "schatten", "spiele"], ["sind", "werden"])
```

Programmierter Ansatz - Nicht fertig...

```
class PoesieMaschine
  # programmierter Ansatz
  def PoesieMaschine.gedicht(nomen, verben)
    i = 0;
    while i<6
      j = 0;
      k = 0;
      while j<2
        out = "";
        # Strohppe 1 und 2
        if(i<2)
          out += nomen[k] + " "
          out += verben[j] + " "
          if(k == 0)
            k += i + 1
          else
            k -= i + 1
          end
        end
        # Strophe 3 bis 6
        else
          out += verben[j] + " "
          out += "TEST" + " "
        end
        # Ausgabe
        puts out += nomen[k]
        # j inkrementieren
        j += 1
      end
      i+=1
      puts "\n"
    end

  end
end
```

PoesieMaschine.gedicht(["worte", "schatten", "spiele"], ["sind", "werden"])