Aufgabe 1

Ausgabedatum: 15.03.2007
Abgabetermin: 22.03.2007

Transformation eines Textes, sodass die Buchstaben jedes 2. Wortes in umgekehrter Reihenfolge erscheinen.

Die in der Vorlesung besprochene Aufgabenstellung der Texttransformation soll in drei Varianten in Java implementiert werden:

1. Implementierung des in der Vorlesung entwickelten (nicht-rekursiven) Algorithmus.
2. Implementierung eines eigenen OO-Designs (z.B. eine eigene Klasse für ein Wort, etc.)
3. Implementierung des in der Vorlesung vorgestellten rekursiven Algorithmus.

Zur Erinnerung nochmals die Aufgabenstellung (in Anlehnung an [EWD302] bzw. [Dijkstra89]):

Gegeben sei ein Zeichensatz bestehend aus:

- Buchstaben
- Leerzeichen
- Punkt

Wörter bestehen aus max. 20 Buchstaben und werden durch beliebig viele Leerzeichen getrennt. Die Eingabe besteht aus beliebig vielen Wörtern und wird durch einen Punkt abgeschlossen.

Die Ausgabe soll wie folgt aussehen:

1. Wörter sollen durch ein einziges Leerzeichen getrennt sein.
2. Das letzte Wort soll durch einen Punkt abgeschlossen werden.
3. Die Buchstaben jedes 2. Wortes sollen in umgekehrter Reihenfolge sein.
   Genauer: Gegeben sei eine Ordnung der Wörter in Leserichtung (d.h. von links nach rechts), sodass jedem Wort eine eindeutige Zahl 0,1,2,... zugeordnet werden kann. Wörter denen eine gerade Zahl zugeordnet ist sind exakt wiederzugeben. Bei Wörtern denen eine ungerade Zahl zugeordnet ist sind die Buchstaben in umgekehrter Reihenfolge wiederzugeben.

Beispiel: "???this?is?a???silly???text???." wird transformiert in "this?si?a?yllis?text." (? entspricht den Leerzeichen)

Nichtrekursiver Algorithmus:

Von dem in der Vorlesung entwickelten Algorithmus gibt es ein Tafelabbild.

Rekursiver Algorithmus (in Anlehnung an [Dijkstra89]):

```
main:
```

```
    repeat
        PrintWord
        if (not end_of_text)
             PrintWordInReverseOrder
    until end_of_text

PrintWord:
    repeat
        SkipAndPrintBlanks;
            ch := RNC;      // read next  character of text
input
            Print(ch);
        until ch= space or ch = point

PrintWordInReverseOrder:
    SkipAndPrintBlanks;
        ch := RNC
        if (end_of_word)
            Print(ch)
        else
            PrintWordInReverseOrder
            Print(ch)
        endif
```

Referenzen:

[EWD302] Dijkstra, E. W: Design considerations in more detail. Circulated privately. Available online: http://www.cs.utexas.edu/users/EWD/ewd03xx/EWD302.PDF
[Dijkstra89] Dijkstra, E. W, Feijen,W. H. J., 1988: A Method of Programming. Addison-Wesley.