Prof. Dr. Franz Schweiggert   Abteilung Angewandte Informationsverarbeitung   3. Dezember 2003
Dr. Andreas Borchert, Michael Wiedemann   Blatt 8

Allgemeine Informatik I (WS 2003/2004)

Abgabetermin: 10. Dezember 2003

18 Verschlüsselung leicht gemacht – 3 Punkte

Ihr habt Euren (Nicht-AI1-) Freunden von Euren neu erworbenen tollen Programmierkenntnissen berichtet. Üblicherweise ist die Welle der Begeisterung Eurer Freunde überschaubar.
So, denen wollen wir es jetzt mal zeigen. Mal sehen, ob sie immer noch lachen, wenn Ihr Ihnen einen verschüsselten Text schickt. Dabei machen wir uns folgenden Umstand zu Nutze:

Jeder Buchstabe, ob groß oder klein, wird durch einen numerischen Wert repräsentiert, festgelegt durch die Norm ISO-8859-1. Diese Norm stellt eine Erweiterung von ASCII dar. Zum Beispiel wird das *A* durch den Wert *65* beschrieben. Durch den Aufruf von *ORD* bzw. *CHR* innerhalb eines Oberon-Programms kann der Wert eines Buchstabens bzw. der zu einem numerischen Wert zugehörige Buchstabe ermittelt werden.

Schreibt also ein Programm, dass die Buchstaben um eine vorher festgelegte Konstante nach "vorne" verschiebt (die übrigen Zeichen, wie zum Beispiel Nummern oder Sonderzeichen, bleiben unberührt). Dies sollte solange durchgeführt werden, bis "nichts" mehr kommt. Am einfachsten benutzt Ihr die Beispieltexte zu diesem Übungsblatt zum Testen. Mit der Dateiumlenkung sollte dies keine Probleme bereiten. Sollte man durch die Verschiebung über die Grenze der Buchstaben hinauskommen, sollte wieder vorne begonnen werden.

Kleines Beispiel hierzu:
Verschiebung der Buchstaben um 10:
Aus a wird k.
Aus x wird h.
Aus Q wird A.
Aus Z wird J.
usw.

19 Verschlüsselung schwer gemacht – 4 Punkte

Nun ist die Verschlüsselung der vorherigen Aufgabe nicht besonders berauschend. Da zum Beispiel das "e" in Texten sehr häufig vorkommt, ist es für Eure zunächst verdutzen Freunde nicht allzu schwierig, auf den verwendeten Verschlüsselungsalgorithmus zu kommen. Deswegen sollte nun etwas trickreicher verschlüsselt werden:

Der erste eingelesene Buchstabe wird um eine Position verschoben, der zweite um zwei, usw. Wiederum sollte solange eingelesen werden, bis nichts mehr kommt. Somit wird zum Beispiel aus "Hallo" "Icopt" – schon besser ...

20 Und wieder zurück – 3 Punkte

Was nützt die beste Verschlüsselung, wenn man das ganze nicht mehr lesen kann? So oder so ähnlich könnte der Hilfeschrei desjenigen ausfallen, dem Ihr einen verschlüsselten Text geschickt habt. Also Kommando zurück. Schreibt für Eure Verschlüsselungsprogramme entsprechende Entschlüsselungsprogramme, die den ursprünglichen Text wiederherstellen. Die ewige Bewunderung Eures Freundes ist Euch nun gewiss.

Generelle Tipps:

- *Eine brauchbare Einleseschleife könnte in etwa so aussehen:*

  ```
  Read.Char(ch);
  (* solange nicht das Ende kommt *)
  WHILE ~Streams.stdin.eof DO
     *hier kommt das Verarbeiten*
     Read.Char(ch); (* das naechste Zeichen einlesen *)
  END;
  ```

  Alternativ kann auch folgendes verwendet werden:

  ```
  WHILE Streams.ReadByte(ch) DO
     (* ch verarbeiten *)
  END;
  ```

- *Es macht Sinn, die Ver- und Entschlüsselungsprogramme im gleichen Ordner zu halten, allerdings muss man dabei beachten, dass das entsprechende Makefile die "richtigen" Dateien verarbeitet. Danach muss man "make" das passende Makefile mitgeben, das geschieht durch "make -f Makefilename". Alternativ könnt Ihr den Befehl "mmo -c makefile" verwenden. Das resultierende makefile baut dann alle Programme bzw. kann natuerlich auch jedes einzelne bauen.*

Viel Erfolg!