Projektaufgabe COBOL
Texteingabe Handy

Team:

Problemstellung

Da Mobiltelefone nur über eine numerische Tastatur verfügen, gestaltet sich die Eingabe von Texten (z.B. für Kurzmitteilungen bzw. SMS Messages) i. Allg. etwas schwierig. Die numerischen Tasten sind jeweils mit mehreren Buchstaben belegt, so dass mehrere Tastebdrücke pro Buchstabe erforderlich sind.
Eine einfache Software, die dieses Problem elegant löst, ist unter dem Namen T9 bekannt. („T9 Text Input“ ist ein eingetragenes Warenzeichen und eine patentierte Technologie von Tegic Communication, Inc.)

Programmsystem

Schreiben Sie ein Programm, das eine vereinfachte Texteingabe auf einer numerischen Tastatur ermöglicht. Die Tastatur soll nur über die folgenden Tasten verfügen: `1234567890*#`.
Die Zuordnung der Buchstaben zu den numerischen Tasten sei wie folgt festgelegt:

1	␣	(Leerzeichen)
2	ABC	
3	DEF	
4	GHI	
5	JKL	
6	MNO	
7	PQRS	
8	TUV	
9	WXYZ	
0	.	(Punkt)

Die Tasten * und # werden zum Abwickeln eines einfachen Benutzerdialoges verwendet:

*	bedeutet Ja bzw. Vorschlag angenommen,
#	bedeutet Nein bzw. Vorschlag abgelehnt.

Für die vereinfachte Texteingabe gelten folgende Bedingungen:

- Es wird nicht zwischen Groß– und Kleinschreibung unterschieden (nur Großbuchstaben).
- Bei der Texteingabe soll pro Buchstabe jede Zifferntaste nur einmal gedrückt werden.
- Ein Wort wird bei der Eingabe stets durch ein Leerzeichen abgeschlossen (Taste 1). Sonderfall: der Punkt (Taste 0) beendet Wort und Satz (Näheres s.u.).

Modularisieren Sie Ihr Programmsystem nach dem EVA-Prinzip sowie nach sinnvollen Funktionseinheiten.

Beispiel

Das Wort "SOFTWARE" kann mit den obigen Zuordnungen durch die Tastenfolge „763892731" definiert werden. Dabei gehört die Ziffer 1 nicht zum Wort selbst, sondern dient der Worttrennung.

Zum Erkennen von Wörtern führt das Programm ein Wörterbuch mit, das zu einer Ziffernkombination die Menge der bekannten Wörter enthält. Das Wörterbuch muss dynamisch erweiterbar sein. Zu Beginn des ersten Aufrufs des Programms ist es leer!
Die Interaktion mit dem Benutzer geschieht so, dass das Programm nach dem Starten auf Texteingabe des Benutzers wartet. Wird eine Ziffernfolge, die mit einem Leerzeichen endet (Taste 1), erkannt, so wird der eingegebene Zifferncode mit dem Wörterbuch verglichen und ggf. das passende Wort ausgegeben. (Anmerkung: aus technischen Gründen kann es notwendig sein, die Eingabe des Wortes zusätzlich zum Leerzeichen durch Drücken der ENTER–Taste zu beenden. Dies ist im Sinne der Aufgabenstellung zulässig!)
Beim Erkennen von Wörtern sind drei Fälle zu unterscheiden:

Fall 1: Zum eingegebenen Zifferncode ist kein passendes Wort im Wörterbuch gespeichert. In diesem Fall muss das Wort im sogenannten „Explizitmodus" eingegeben werden. Der Explizitmodus wird weiter unten erläutert. Wird das im Explizitmodus eingegebene Wort vom Benutzer als korrekt akzeptiert, so wird es zusammen mit dem Zifferncode in das Wörterbuch aufgenommen. Außerdem wird es an den zu bildenden Satz angehängt. Das Bilden von Sätzen wird weiter unten erläutert.

Fall 2: Zum eingegebenen Zifferncode ist genau ein Wort im Wörterbuch gespeichert. In diesem Fall wird das gespeicherte Wort als Wortvorschlag ausgegeben und der Benutzer erhält Gelegenheit, das Wort als korrekt zu akzeptieren (Taste *) oder es zu verwerfen (Taste #). Akzeptiert er das Wort, so wird es Bestandteil des zu bildenden Satzes, verwirft er es, so fährt das Programm mit der Eingabe im Explizitmodus fort. (Wie in Fall 1 erhält der Benutzer wieder Gelegenheit, das im Explizitmodus eingegebene Wort als korrekt zu akzeptieren (Taste *) oder es zu verwerfen (Taste #). Das akzeptierte Wort wird wiederum in das Wörterbuch und in den Satz aufgenommen.)

Fall 3: Es sind mehrere Wörter zum eingegebenen Zifferncode gespeichert. Der Reihe nach werden nun alle im Wörterbuch zu diesem Code verzeichneten Wörter als Wortvorschlag angeboten. Dabei soll nach der Häufigkeit des Auftretens vorgegangen werden. Das am häufigsten aufgetretene Wort zuerst, dann das zweithäufigste usw. (Anmerkung: Jedesmal, wenn ein bereits im Wörterbuch gespeichertes Wort erneut ausgewählt wird, wird seine Häufigkeit um eins erhöht.)

Wie in Fall 2 wird ein akzeptiertes Wort in den Satz übernommen und die Ausgabe von Wortvorschlägen für diesen Code endet. Akzeptiert der Benutzer keinen der vom Programm gemachten Wortvorschläge, so folgt wieder die Dateneingabe im Explizitmodus wie im Fall 2.

Dateneingabe im Explizitmodus

Im Explizitmodus ist die Texterkennung über das Wörterbuch ausgeschaltet. Jeder Buchstabe wird jetzt durch zwei Ziffern codiert:

1. Die erste Ziffer bezeichnet den Buchstabenblock (z.B. `4` für `GHI`).
2. Die zweite Ziffer gibt an, der wievielte Buchstabe im Block gewählt wird (z.B. `1` für `G`, `3` für `I` usw.).

Auch im Explizitmodus wird das Wort durch ein Leerzeichen abgeschlossen (Taste `1`), wobei diese Taste jedoch nur **einmal** gedrückt wird. (Wie vorher ist es zulässig, die Eingabe durch die ENTER–Taste abzuschließen.)

Beispiel

Das Wort „`WORT`“ hat im Explizitmodus die Darstellung „`916373811`“.

Bilden von Sätzen

Im Laufe der Interaktion mit dem Benutzer soll jedes der akzeptierten Wörter an den aktuellen Satz angehängt werden. Wörter sollen durch ein Leerzeichen voneinander getrennt werden. Die Eingabe eines Punktes (Taste `0`) beendet Wort und Satz. Der so beendete Satz soll nun als Zeichenkette ausgegeben werden. Ein neuer Satz beginnt. Ein Satz, der nur aus dem Punkt besteht (Taste `0`), beendet das Programm.

Zum Programmablauf

Damit das Wörterbuch wiederverwendet werden kann, soll es vor dem Beenden des Programms abgespeichert werden (auf Festplatte oder Floppy etc.) und ein bestehendes Wörterbuch beim Starten des Programms wieder eingelesen werden. Für die Speicherung ist die folgende Spezifikation zu verwenden:

- Alle Daten werden als lesbarer ASCII–Text gespeichert.
- Pro Eintrag des Wörterbuches wird eine Zeile gespeichert (durch Zeilenvorschub getrennt).
- Jede Zeile hat den Aufbau: *¡Code¿¡Wort¿¡Häufigkeit¿*. Dabei werden die Komponenten einer Zeile durch genau ein Leerzeichen getrennt. Der Zifferncode und das Wort des Wörterbuches werden ohne das abschließende Leerzeichen bzw. den abschließenden Punkt gespeichert.
- Das gespeicherte Wörterbuch muss mit einem normalen Texteditor bearbeitbar sein.

Das Wort „`WORT`“ hat im Explizitmodus die Darstellung „`916373811`“.

Bilden von Sätzen

Im Laufe der Interaktion mit dem Benutzer soll jedes der akzeptierten Wörter an den aktuellen Satz angehängt werden. Wörter sollen durch ein Leerzeichen voneinander getrennt werden. Die Eingabe eines Punktes (Taste `0`) beendet Wort und Satz. Der so beendete Satz soll nun als Zeichenkette ausgegeben werden. Ein neuer Satz beginnt. Ein Satz, der nur aus dem Punkt besteht (Taste `0`), beendet das Programm.

Zum Programmablauf

Damit das Wörterbuch wiederverwendet werden kann, soll es vor dem Beenden des Programms abgespeichert werden (auf Festplatte oder Floppy etc.) und ein bestehendes Wörterbuch beim Starten des Programms wieder eingelesen werden. Für die Speicherung ist die folgende Spezifikation zu verwenden:

- Alle Daten werden als lesbarer ASCII–Text gespeichert.
- Pro Eintrag des Wörterbuches wird eine Zeile gespeichert (durch Zeilenvorschub getrennt).
- Jede Zeile hat den Aufbau: *¡Code¿¡Wort¿¡Häufigkeit¿*. Dabei werden die Komponenten einer Zeile durch genau ein Leerzeichen getrennt. Der Zifferncode und das Wort des Wörterbuches werden ohne das abschließende Leerzeichen bzw. den abschließenden Punkt gespeichert.
- Das gespeicherte Wörterbuch muss mit einem normalen Texteditor bearbeitbar sein.

Beispiel:

```
76389273 SOFTWARE 1
9678 WORT 5
337 DER 7
337 FES 1
...
```

Hinweise und Beschränkungen

- Die maximale Wortlänge beträgt 15 Zeichen. Die maximale Satzlänge beträgt 200 Zeichen.
- Die Größe des Wörterbuches darf begrenzt sein. Die in einer konkreten Implementierung verwendete Maximalgröße soll jedoch durch eine einfache Änderung im Quelltext und Neucompilation leicht veränderbar sein.
- Da das Wörterbuch häufig durchsucht wird, sollte eine möglichst geeignete Datenstruktur für die interne Speicherung gefunden werden. Sie soll auch bei einer sehr großen Anzahl von Einträgen im Wörterbuch effizient bearbeitet werden können (Suchen, Einfügen). Beim Speichern des Wörterbiches auf Festplatte ist die interne Datenstruktur in die geforderte externe Form umzuwandeln. Beim Einlesen des Wörterbuches ist der Vorgang entsprechend umzukehren. Bei beiden Vorgängen ist keine Sortierung vorzunehmen.

Zusatzfunktion (da Dreiergruppe)

Es soll die Möglichkeit bestehen, ein bereits vorhandenes (externes) Wörterbuch für die Bearbeitung einzulesen und dann gemäß obiger Aufgabenstellung zu erweitern.
Nach Beendigung eines Dialogs soll das verwendete Wörterbuch alphabetisch aufsteigend sortiert werden.

Beispiel eines Programmdialogs – hier ein Bsp. ohne obige Zusatzfunktion

Wir beginnen mit einem leeren Wörterbuch.

Benutzereingabe	Antwort des Programms
	Kein Woerterbuch geladen
3371	
	Kein passendes Wort gespeichert.
	Eingabe im Explizitmodus:
3132731	
	DER
	Wort ok?
*	
	Wort gespeichert
4881	
	Kein passendes Wort gespeichert.
	Eingabe im Explizitmodus:
4282811	
	HUT
	Wort ok?
*	
	Wort gespeichert
4781	
	Kein passendes Wort gespeichert.
	Eingabe im Explizitmodus:
4374811	
	IST
	Wort ok?
*	
	Wort gespeichert
3461	
	Kein passendes Wort gespeichert.
	Eingabe im Explizitmodus:
3243621	
	EIN
	Wort ok?
*	
	Wort gespeichert

```
3370
        DER
        Wort ok?
#
        Eingabe im Explizitmodus:
3332740
        FES
        Wort ok?
*
        Wort gespeichert
        Der Satz lautet:
        DER HUT IST EIN FES.
0
        Programmende.
```

Das gespeicherte Wörterbuch könnte bei Programmende das folgende Aussehen haben:

```
337 FES 1
337 DER 1
346 EIN 1
478 IST 1
488 HUT 1
```

Testbeispiele

Es sollen Testbeispiele für alle wichtigen Standard– und Sonderfälle untersucht und diskutiert werden. Dabei sollen auch Fehlersituationen und Grenzfälle betrachtet werden.
Das erste Beispiel (Beispiel aus der Aufgabenstellung) ist mit einem leeren Wörterbuch zu beginnen. Nach dem ersten Beispiel sollen die beiden folgenden Sätze:

```
DER SATZ IST KURZ.
EIN FES IST EIN HUT.
```

in der beschriebenen Form eingegeben werden. Bei jedem Beispiel soll auf dem im vorigen Beispiel erstellten Wörterbuch aufgesetzt werden.
Nach jedem Beispiel ist das Wörterbuch in Textform in der Testdokumentation anzugeben (vgl. externes Wörterbuchformat).

Entwicklerdokumentation

Die ganzheitliche Entwicklerdokumentation sollte ein Inhaltsverzeichnis aufweisen und zu jedem Kapitel eine Autorenangabe erhalten.
Dokumentieren Sie jeweils zu Ihrem Teil (s. u.), Beschreibung der Algorithmen anhand eines selbstgewählten Beispiels sowie als Struktogramm und Programmeinheiten.
Begründen Sie die Auswahl Ihrer Testfälle und diskutieren Sie sie.

Teil A

- Einlesen des Wörterbuchs am Programmanfang
- Speichern des Wörterbuchs bei Programmende
- Behandlung von Fehlersituationen

Das Programm soll in der Lage sein, mehrere Testfälle nacheinander automatisch zu verarbeiten.

Teil B

- Durchführung des Benutzerdialogs

Teil C

- Einlesen und Erweitern des Wörterbuchs gemäß oben beschriebener Zusatzfunktion (inkl. Sortierung)

Abzugeben sind:

Aufgabenstellung, Programmcodes als Listing und ausführbares Programmsystem, E-/A-Dateien der Tests, die oben beschriebene ganzheitliche Entwicklerdokumentation in schriftlicher Form (Word-, oder pdf-Datei).
Unterschriebene Eigenständigkeitserkärung mit Nennung der eigenen erstellten Programmteile.