Aufgabe 27

Unter Moodle finden Sie die Eingabefiles ‚uebung27.l' für den Scannergenerator lex und ‚uebung27.y' für den Parsergenerator bison, die noch verändert werden sollen. Achten Sie darauf, dass das im Vorspann mit uebungs27_tab.h bezeichnete Header-file vom Parsergenerator bison geschrieben wird. Die erforderlichen binaries für die Anwendung des Parsergenerators bison können Sie für Windows unter https://sourceforge.net/projects/winflexbison/
finden.
Das Generieren des Scanners aus uebung27.l funktioniert so wie in den Aufgaben 7 und 8. Für das Generieren des Parsers mit win_bison rufen Sie einfach

```
win_bison -dtv uebung27.y
```

auf.
Anschließend bitte sowohl die entstandenen c-Dateien:
uebung27.tab.c
lex.yy.c
als auch das von bison generierte header-file:
uebung27.tab.h
in das Projekt einbinden und übersetzen.
In der von win_bison erstellten Datei uebung27.output sehen Sie die Parsertabelle für den LR-Parser nebst einigen anderen Informationen.
Nun zur Aufgabe:
Folgende Grammatik ist gegeben:

```
S: A B C;
A: a;
B: b B;
B: b;
C: c;
```

Die entstehende Sprache is $L(G)=\{ab^nc|n>=1\}$. Ihr Parser soll nach der Eingabe eines Worts, die Anzahl der auftretenden b's bestimmen.
Geben Sie die Grammatik an der geeigneten Stelle in uebung27.y ein und führen Sie Aktionen aus, die die Anzahl der b's im Eingabewort bestimmen.

Aufgabe 28

Ein Roboter kann mit Hilfe folgender Befehle gesteuert werden:
s – Start
v – 1 Schritt voraus
z – 1 Schritt zurück
r – Drehung um 90° nach rechts
l – Drehung um 90° nach links
e – Ende der Eingabe

Schreiben Sie mit flex und bison ein Programm, das eine beliebige Befehlsequenz einliest und die Koordinaten ausgibt, die der Roboter nach dieser Schrittfolge einnimmt. Der Roboter startet im Koordinatenursprung (0,0) und ist in x-Richtung ausgerichtet.

Beispieleingaben:
`s v v r v l v e` ergibt die Endkoordinaten (3/-1)
`s l v v r v l v e` ergibt die Endkoordinaten (1/3)

Aufgabe 29

Erstellen Sie mittels der tools flex und bison, die auch in Aufgabenblatt 11 zum Einsatz kamen, einen „Taschenrechner“, der geklammerte Ausdrücke mit +,-,* und / verarbeiten kann und das Ergebnis eines eingegebenen Ausdruckes auf dem Bildschirm ausgibt.
Eingabebeispiel: 3.4+5*7
Ausgabe:38.4

Aufgabe 30

Erweitern Sie den Taschenrechner aus Aufgabe so, dass Variablenzuweisungen verarbeitet werden können. Hierzu müssen Sie eine Symboltabelle erstellen (z.B. in Form eines arrays), die Namen und Wert der Variablen enthält.

Aufgabe 31

Wandeln Sie den Taschenrechner so ab, dass der eingegebene Ausdruck in Postfix-Notation ausgegeben wird. Dies kann beispielsweise bereits als Zwischencode für eine Stack-Maschine verwendet werden.
Eingabebeispiel: 3.4+5*7
Ausgabebeispiel Zwischencode Stackmaschine:

```
PUSH 3.4
PUSH 5
PUSH 7
MULT
ADD
```

Aufgabe 32

Schreiben Sie einen Taschenrechner mit flex+bison, der unär codierte ganze Zahlen erkennt, in Dezimalzahlen übersetzt und die Summe dieser Zahlen als Dezimalzahl ausgibt:
Eingabe: 11111+111
Ausgabe: 5+3=8