Answers to the exercises for chapter: Parsing

1. Given two transitions one an input symbol, replace that by a single transition to a new state, and ϵ-transitions to the original two destination states.
   If an NFA has an ϵ-transition from state 1 to state 2, we can replace that by a jump from state 0 to state 2, triggered by the symbol that made the transition from 0 to 1.
2. *no answer given*
3. Given a mapping $I \times S \to O$, and denote $o_{i,s}$ the output symbol for input i and state s. For each state s, introduce a new state s_i for any input symbol that has a defined transition out of s. Let that state output the symbol $o_{i,s}$. Now define an ϵ-transition from s_i to the state that was the original transition from s under i. Give that state no output.
   The reverse equivalence is trivial: if a state maps to an output, then it maps to that output with every input. Formally, if $s \mapsto o$ in the second definition, then $\{i, s\} \mapsto o$ for all i in the first.
4. *no answer given*
5. *no answer given*
6. *no answer given*
7. This question derives from `http://www.gnu.org/software/bison/manual/html_node/Shift-Reduce.html` Proof of shifting, which implies right associativity:

```
%{

#include "sr.h"
extern int yylval;

%}

%%

[0-9]+ {yylval = atoi(yytext); return N;}
[+\n-] {return *yytext;}

%token N

%%

P : E '\n' {printf("Result: %d\n",$$);}
```

```
E : E '+' E {$$ = $1+$3;}
| E '-' E {$$ = $1-$3;}
  | N

%%

  int main(void)
{
  yyparse();
}
```

Ouput

```
%% sr
1-2+5
Result: -6
```