Task 1.1 Eivind Siqveland Larsen (eivinlar)

E --> S | S r S
S --> T | I T | S a T
T --> F | T m F
F --> i | n
I --> p | m

First and follow sets for nonterminals:

NT	First(NT)	Follow(NT)
E	{i,n,p,m}	{$}
S	{i,n,p,m}	{$, a, r}
T	{i,n}	{$, a, r, m}
F	{i,n}	{$, r, a, m}
I	{p,m}	{i,n}

Task 1.2

1. A top-down parser cannot simply parse a left recursion grammar, where as a bottom-up can.
2. GNU Bison generates a LALR parser.
3. This is a lot of work and the course would not be very efficient if we would spend so much time handcoding everything.

Task 1.3

1. A shift-reduce parser parses input by pushing tokens onto a stack and tries to generate the parse tree bottom up by identifying terminals first (the LHS), then reducing the stack by removing the LHS and replacing it with a RHS expression, until only the grammars starting symbol is left on the stack.

2. A shift-reduce conflict occurs when a action in the action table calls for both a shift and a reduce action.
A reduce-reduce conflict occurs when a action in the action table calls for two different reduce actions.