Banner number: Name:

Midterm Exam

CSCI 3136: Principles of Programming Languages

March 2, 2018

Question 1.1		Question 2.1		Question 3.1		$\sum$
Question 1.2		Question 2.2		Question 3.2		
$\sum$		$\sum$		$\sum$		

Instructions:

- Provide your answer in the box after each question. If you absolutely need extra space, use the backs of the pages; but try to avoid it. Keep your answers short and to the point.
- You are not allowed to use a cheat sheet.
- Make sure your answers are clear and legible. If I can't decipher an answer or follow your train of thought with reasonable effort, you'll receive 0 marks for your answer.
- Read every question carefully before answering.
- Do not forget to write your banner number and name on the top of this page.
- This exam has 7 pages, including this title page. Notify me immediately if your copy has fewer than 7 pages.

1 Programming Languages

Question 1.1 — 10 marks

(a) Why doesn't a purely functional language have any loop constructs?

Variables in a purely functional language are immutable. A loop requires mutation of variables. Without mutation of variables, any condition for exiting the loop is either true or false in every iteration. In the former case, the loop has at most one iteration. In the latter case, it runs forever.

(b) Without loops, how do you express iterative computations in a purely functional language? How do you ensure that this uses no more space than the corresponding loop would use.

Any iterative computation can be expressed using recursion. If the computation is expressed using a tail-recursive function (the recursive call is the last thing the function does and the return value of the function is whatever this recursive call returns), the compiler can translate the recursive calls into a loop, so no stack frames are built up for the recursive calls.

Question 1.2 — 10 marks

Consider the following Prolog database:

```
a(1).  a(2).  a(3).  b(1,1).  b(1,2).  b(2,3).  b(2,4).  b(5,5).
```

and the following two implementations of a Prolog predicate `f`:

(a)
```
f(X,Y) :- g(X,Y).
f(3,3).
g(X,Y) :- a(X), b(X,Y).
g(4,4).
```

(b)
```
f(X,Y) :- g(X,Y).
f(3,3).
g(X,Y) :- a(X), !, b(X,Y).
g(4,4).
```

Provide the output the query `?- f(X,Y).` produces for each definition of `f`:

(a)
```
X = 1, Y = 1 ;
X = 1, Y = 2 ;
X = 2, Y = 3 ;
X = 2, Y = 4 ;
X = 4, Y = 4 ;
X = 3, Y = 3.
```

(b)
```
X = 1, Y = 1 ;
X = 1, Y = 2 ;
X = 3, Y = 3.
```

2 Regular Languages

Question 2.1 — 10 marks

(a) Formally define what a DFA is.

A DFA is a tuple $D = (S, \Sigma, \delta, s_0, F)$ where

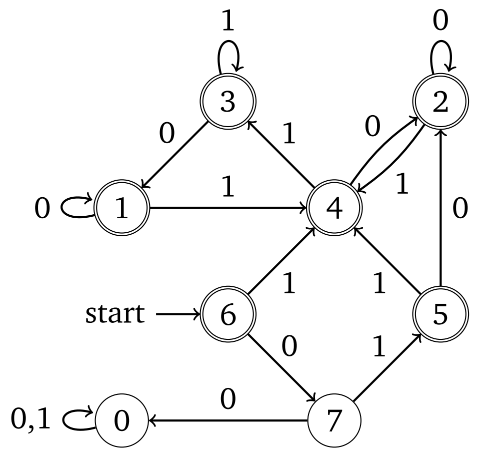
- *S is a set of* states,
- *Σ is a finite* alphabet,
- *$\delta : S \times \Sigma \rightarrow S$ is a* transition function,
- *$s_0 \in S$ is the* start state, *and*
- *$F \subseteq S$ is a set of* accepting states.

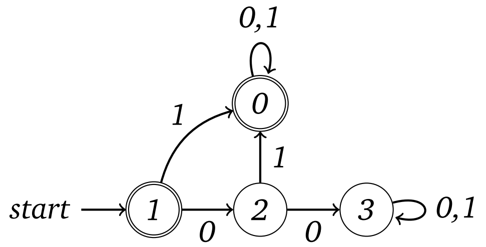
(b) Formally define the language decided by a DFA.

Let $\delta^*(s, \sigma)$ be the state reached by D after starting in the state s and reading the string σ. Then the language decided by D is $\mathcal{L}(D) = \{\sigma \in \Sigma^* \mid \delta^*(s_0, \sigma) \in F\}$. Formally, $\delta^*(s, \varepsilon) = s$ for all $s \in S$ and $\delta^*(s, x\sigma) = \delta^*(\delta(s, x), \sigma)$ for all $s \in S$, $x \in \Sigma$, and $\sigma \in \Sigma^*$.

Question 2.2 **10 marks**

Consider the following DFA. Construct another DFA that decides the same language and has the minimum possible number of states.





3 Context-Free Languages

Question 3.1 **10 marks**

(a) Formally define what a context-free grammar is.

A context-free grammar is a tuples $G = (V, \Sigma, P, S)$ *where*

- V *is a set of* non-terminals*,*
- Σ *is a set of* terminals*,*
- P *is a set of* productions $X \to \sigma$ *with* $X \in V$ *and* $\sigma \in (V \cup \Sigma)^*$*, and*
- $S \in V$ *is a* start symbol.

(b) Formally define the language defined by a context-free grammar.

G defines the language $\mathcal{L}(G) = \{\sigma \in \Sigma^* \mid S \Rightarrow_G^* \sigma\}$ *where*

- $\alpha X \beta \Rightarrow_G \alpha\sigma\beta$ *for all* $\alpha, \beta \in (V \cup \Sigma)^*$ *and* $(X \to \sigma) \in P$ *and*
- $\sigma \Rightarrow_G^* \tau$ *if* $\sigma = \tau$ *or* $\sigma \Rightarrow_G \sigma'$ *and* $\sigma' \Rightarrow_G^* \tau$ *for some* $\sigma' \in (V \cup \Sigma)^*$.

Question 3.2 **10 marks**

Consider the following grammar G (only the productions are shown; the start symbol is *FunCall*):

$$\begin{aligned}
\mathit{FunCall} &\to \mathit{id}\ (\ \mathit{Args}\ )\\
\mathit{Args} &\to \varepsilon\\
\mathit{Args} &\to \mathit{Arg}\ \mathit{MoreArgs}\\
\mathit{MoreArgs} &\to \varepsilon\\
\mathit{MoreArgs} &\to\ ,\ \mathit{Arg}\ \mathit{MoreArgs}\\
\mathit{Arg} &\to \mathit{id}\\
\mathit{Arg} &\to \mathit{num}\\
\mathit{Arg} &\to \mathit{FunCall}
\end{aligned}$$

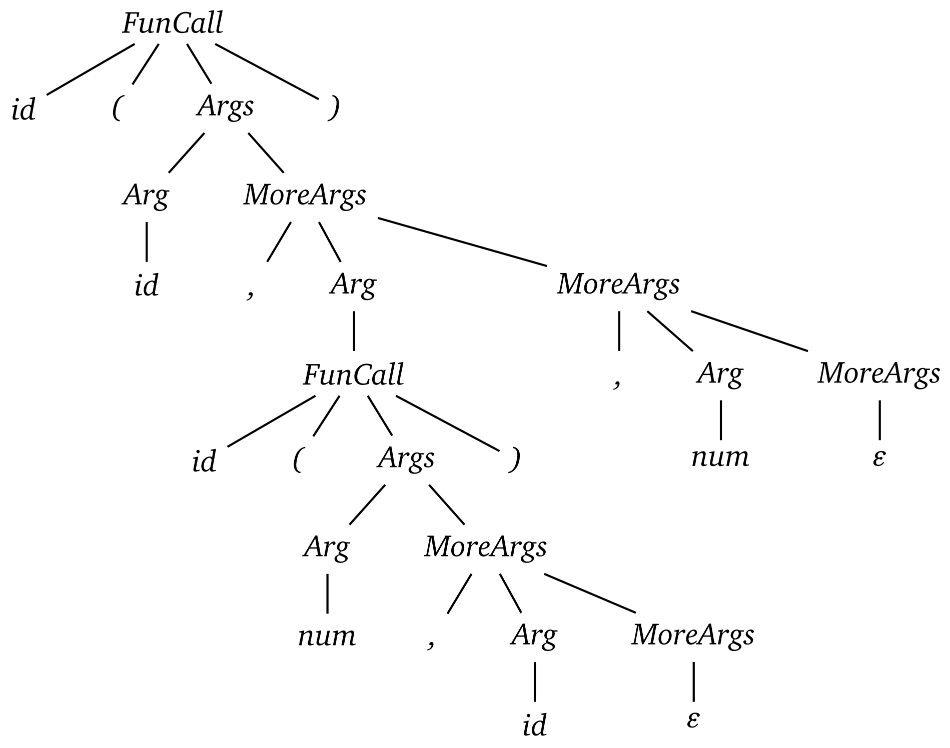
and the following strings:

$$\mathit{id}\,(\,\mathit{num}\,,\mathit{id}\,(\,\mathit{id}\,,)\,)\qquad \mathit{id}\,(\,\mathit{id}\,,\mathit{id}\,(\,\mathit{num}\,,\mathit{id}\,)\,,\mathit{num}\,)\qquad \mathit{id}\,(\,\mathit{id}\,\mathit{num}\,)$$

(a) Exactly one of the strings is in the language defined by this grammar. Which one?

$$\mathit{id}\,(\,\mathit{id},\,\mathit{id}\,(\,\mathit{num},\,\mathit{id}\,)\,,\,\mathit{num}\,)$$

(b) For the string σ you provided in answer to question (a), provide a parse tree using the grammar G that has σ as its yield.



(c) Provide a left-most derivation for the string σ in (a) using G.

$$\begin{aligned}
\mathit{FunCall} &\Rightarrow \mathit{id}\,(\,\mathit{Args}\,) \\
&\Rightarrow \mathit{id}\,(\,\mathit{Arg}\,\mathit{MoreArgs}\,) \\
&\Rightarrow \mathit{id}\,(\,\mathit{id}\,\mathit{MoreArgs}\,) \\
&\Rightarrow \mathit{id}\,(\,\mathit{id}\,,\,\mathit{Arg}\,\mathit{MoreArgs}\,) \\
&\Rightarrow \mathit{id}\,(\,\mathit{id}\,,\,\mathit{FunCall}\,\mathit{MoreArgs}\,) \\
&\Rightarrow \mathit{id}\,(\,\mathit{id}\,,\,\mathit{id}\,(\,\mathit{Args}\,)\,\mathit{MoreArgs}\,) \\
&\Rightarrow \mathit{id}\,(\,\mathit{id}\,,\,\mathit{id}\,(\,\mathit{Arg}\,\mathit{MoreArgs}\,)\,\mathit{MoreArgs}\,) \\
&\Rightarrow \mathit{id}\,(\,\mathit{id}\,,\,\mathit{id}\,(\,\mathit{num}\,\mathit{MoreArgs}\,)\,\mathit{MoreArgs}\,) \\
&\Rightarrow \mathit{id}\,(\,\mathit{id}\,,\,\mathit{id}\,(\,\mathit{num}\,,\,\mathit{Arg}\,\mathit{MoreArgs}\,)\,\mathit{MoreArgs}\,) \\
&\Rightarrow \mathit{id}\,(\,\mathit{id}\,,\,\mathit{id}\,(\,\mathit{num}\,,\,\mathit{id}\,\mathit{MoreArgs}\,)\,\mathit{MoreArgs}\,) \\
&\Rightarrow \mathit{id}\,(\,\mathit{id}\,,\,\mathit{id}\,(\,\mathit{num}\,,\,\mathit{id}\,)\,\mathit{MoreArgs}\,) \\
&\Rightarrow \mathit{id}\,(\,\mathit{id}\,,\,\mathit{id}\,(\,\mathit{num}\,,\,\mathit{id}\,)\,,\,\mathit{Arg}\,\mathit{MoreArgs}\,) \\
&\Rightarrow \mathit{id}\,(\,\mathit{id}\,,\,\mathit{id}\,(\,\mathit{num}\,,\,\mathit{id}\,)\,,\,\mathit{num}\,\mathit{MoreArgs}\,) \\
&\Rightarrow \mathit{id}\,(\,\mathit{id}\,,\,\mathit{id}\,(\,\mathit{num}\,,\,\mathit{id}\,)\,,\,\mathit{num}\,)
\end{aligned}$$