CMSC330 Spring 2023 Quiz 4 Solutions

Proctoring TA: _____ **Name:** _____

UID: _____

Problem 1: Basics [3 pts]

Please circle **True** or **False** for the following statements:

All Regular Expressions can be expressed as CFGs **True** **False**

Regular expressions have a starting character and terminals and nonterminals, all things that can be expressed as a CFG.

One could theoretically implement NFA to DFA in Lambda Calculus **True** **False**

Since lambda calculus expresses functions, one could write a function that converts an NFA to a DFA using lambda calc.

Ambiguous Grammars grammars have at maximum, one right-most and one left-most deriviation for any given string **True** **False**

Ambiguous grammars have more than one left-most derivation or more than one right most derivation of the same string.

Problem 2: Operational Semantics [6 pts]

Consider the following rules for a subset of OCaml

$$\frac{}{\text{true} \rightarrow \text{true}} \qquad \frac{}{\text{false} \rightarrow \text{false}}$$

$$\frac{A;\ e_1 \Rightarrow v_1 \qquad A;\ e_2 \Rightarrow v_2 \qquad v_1 \text{ is equal to } v_2}{A;\ e_1 <> e_2 \Rightarrow \text{false}} \qquad \frac{A;\ e_1 \Rightarrow v_1 \qquad A;\ e_2 \Rightarrow v_2 \qquad v_1 \text{ is not } v_2}{A;\ e_1 <> e_2 \Rightarrow \text{true}}$$

$$\frac{A,x{:}v\ (x) = v}{A,x{:}v;\ x \Rightarrow v} \qquad \frac{A;\ e_1 \Rightarrow v_1 \qquad A,x{:}v_1;\ e_2 \Rightarrow v_2}{A;\ \text{let } x = e_1 \text{ in } e_2 \Rightarrow v_2}$$

$$\frac{A;\ e_1 \Rightarrow v_1 \qquad A;\ e_2 \Rightarrow v_2 \qquad v_3 \text{ is the max of } v_1 \text{ and } v_2}{A;\ max(e_1,e_2) \Rightarrow v_3} \qquad \frac{}{A;n \rightarrow n}$$

Prove the following statement is valid and returns `true`

$$\text{let x = 8 in 3 <> max(3,x)}$$

$$\frac{\boxed{1} \qquad \dfrac{\dfrac{}{A, x:8; 3 \Rightarrow 3} \qquad \dfrac{\dfrac{}{A, x:8; 3 \Rightarrow 3} \qquad \dfrac{\boxed{6}}{A, x:8; x \Rightarrow 8} \qquad 8 \text{ is the max of 3 and 8}}{A, x:8; \boxed{3} \Rightarrow \boxed{4}} \qquad \boxed{5}}{A, x:8;\ 3 <> \boxed{2} \Rightarrow true}}{A; \text{let x = 8 in 3 <> } max(3,x) \Rightarrow \text{true}}$$

Blank 1: A; 8 ⇒ 8 | Blank 2: $max(3, x)$ | Blank 3: $max(3, x)$

Blank 4: 8 | Blank 5: 3 is not 8 | Blank 6: $A, x : 8(x) = 8$

- Blank 1: Following the let rule, we evaluate the expression after "x =" which is "8". Seeing that "8" is a number we would you the number axiom; thus, A; 8 ⇒ 8.
- Blank 2: Similarly, following the let rule, we evaluate the expression after "in" which is "3 <> $max(3, x)$. Seeing that the only missing part of this expression (before ⇒) is $max(3, x)$, that would be Blank 2.
- Blank 3: Followign the <> rule, 3 and $max(3, x)$ have to evaluated separately and then compared in a third hypothesis. Since on the left hand side 3 has already been evaluated, the next expression to be evaluated would be $max(3, x)$ which is why it is Blank 3.
- Blank 4: Whatever $max(3, x)$ evaluates to is the final value for this conclusion. Following the final hypothesis on the top we can see that since "8 is the max of 3 and 8", Blank 4 should be 8.
- Blank 5: Following the <> rule, and seeing at the conclusion that this evaluates to true, this hypothesis has to say that "v_1 is not v_2". Seeing the other hypotheses we can tell that v_1 is 3 and v_2 is 8 (these are the values the other hypotheses evaluate to). Therefore the final statement should be "3 is not 8".
- Blank 6: Following the rule for evaluating variables, the hypothesis should be "A, x:8(x) = 8".

[4 pts] **Problem 3: Context Free Grammars**

Consider the following Grammars:

Grammar 1	Grammar 2	Grammar 3	Grammar 4
S -> AB	S -> ASB|c	S -> Sc|AB	S -> ASB|cSc|c
A -> aA|a	A -> aA|a	A -> aA|a	A -> aA|a
B -> bbB|ϵ	B -> bbB|ϵ	B -> bbB|ϵ	B -> bbB|bb

[1 pts] (a) Which grammar accepts both "aaabb" and "aaabbcc"

Grammar 1 Grammar 2 [Grammar 3]

[1 pts] (b) Which Grammar is ambiguous?

Grammar 1 [Grammar 2] Grammar 3

[2 pts] (c) Which strings are accepted by Grammar 4?

aaacbbb [aaacbbbb] ccaaabbbbcc cacacbbbb

(a) Grammar 3 accepts both "aaabb" and "aaabbcc". Let us look at the production for both:

1. "aaabb": S -> AB -> aAB -> aaAB -> aaaB -> aaabbB -> aaabb (as B goes to ϵ)

2. "aaabbcc": S -> Sc -> Scc -> ABcc -> aABcc -> aaABcc -> aaaBcc -> aaabbBcc -> aaabbcc (as B goes to ϵ)

(b) Grammar 2 is ambiguous. Let us look at two leftmost derivations of "aac":

1. S -> ASB -> aSB -> aASB -> aaSB -> aacB -> aac
2. S -> ASB -> aASB -> aaSB -> aacB -> aac

(c) Only "aaacbbbb" is accepted by Grammar 4:

1. "aaacbbb" is not accepted as S -> ASB -> ... and B only terminates in an even number of "b"s. Since the string has 3 "b"s it will not be accepted.
2. "aaacbbbb": S -> ASB -> aASB -> aaASB -> aaaSB -> aaacB -> aaacbbB -> aaacbbbb
3. "ccaaabbbbcc" will nto be accepted as S -> cSc -> ccScc -> ccASBcc -> ... Since we have an S in between A and B, and we know S will always terminate in a c, it would only accept strings which has 1 or more c's in between a's and b's. Our string here, only consists of c's at the end and not in between aaa and bbbb, therefore, it will not be accepted.

Problem 4: Lambda Calculus [7 pts]

(a) Circle the free variables and underline the bound variables in the following lambda calculus expression [3 pts]

$$(\lambda x.((\lambda y.(\underline{x}\ \underline{y}))\underline{x}\ \boxed{z}))(\lambda z.\boxed{w})$$

Consider the following λ expressions

$$(\lambda x.(\lambda y.x\,y))((\lambda y.a)(\lambda x.x))$$

(b) Which of the following is the result of reducing the outer-most expression **once** using lazy (call by name) evaluation? [2 pts]

$(\lambda x.(\lambda y.x\,y))a$ $\quad$ $\boxed{\lambda y.((\lambda y.a)(\lambda x.x))y}$ $\quad$ $\lambda y.a\,y$

(c) Which of the following is the result of reducing the outer-most expression **once** using eager (call by value) evaluation? [2 pts]

$\boxed{(\lambda x.(\lambda y.x\,y))a}$ $\quad$ $\lambda y.((\lambda y.a)(\lambda x.x))y$ $\quad$ $\lambda y.a\,y$

(a) Since x is within the body of the outermost lambda expression, it is bound. Same for y - since it is in the body of the inner lambda expression, it is bound. z hs no binding in this lambda expression and is therefore free. We can see the same thing with w.

(b) Consider $(\lambda x.(\lambda y.x\,y))$ as e1 and $((\lambda y.a)(\lambda x.x))$ as e2. Since we are performing lazy evaluation, we do beta reduction without evaluating e2. Applying the argument e2, we get $\lambda y.((\lambda y.a)(\lambda x.x))y$

(c) Consider $(\lambda x.(\lambda y.x\,y))$ as e1 and $((\lambda y.a)(\lambda x.x))$ as e2. Now, since we are performing eager evaluation, we do beta reduction by first evaluating e2. Within e2 we have 2 lambda expressions. Applying the argument $(\lambda x.x)$ to the first expression, we get $(\lambda x.(\lambda y.x\,y))a$