Assignment#3 Sample Key

1. Consider L = { a^m b^n c^t | t = min(m,n) }

a.) Use the **Pumping Lemma for CFLs** to show **L is not** a CFL

Me: **L** is a CFL

PL: Provides **N>0**

Me: **z = a^N b^N c^N**

PL: **z = uvwxy, |vwx| ≤ N**, **|vx| > 0**, and **∀i≥0 uv^iwx^iy ∈ L**

Me: Since **|vwx| ≤ N**, it can consist of **a**'s and/or **b**'s or **b**'s and/or **c**'s but never all three.
Assume it contains no **c**'s then **i=0** decreases the number of **a**'s and/or the number of **b**'s, but not the **c**'s and so there are more **c**'s than the minimum of **a**'s and **b**'s.
Assume it contains **c**'s then **i=2** increases the number of **c**'s and maybe number of **b**'s, but not the **a**'s and so there are more than **N** **c**'s but just N a's.

1. Consider $L = \{ a^m b^n c^t \mid t = \min(m,n) \}$

b.) Present a CSG for **L** to show it **<u>is</u>** context sensitive

G = ({ A, B, C, <a>, <b> }, { a, b, c }, R, A)

A → aBbc | abc | a | b | λ

B → aBbC | abC | // match a's, b's and c's

a<a>bC | ab<b>C // allow more a's or b's

Cb → bC // Shuttle C over to a c

Cc → cc // Change C to a c

<a> → a<a> | a

<b> → b<b> | b

2. Language Closure under Various Operation

Consider some language **L**. For each of the following cases, write in one of **(i)** through **(vi)**, to indicate what you can say conclusively about **L**'s complexity, where

(i) **L** is definitely regular
(ii) **L** is context-free, possibly not regular, but then again it might be regular
(iii) **L** is context-free, and definitely not regular
(iv) **L** might not even be context-free, but then again it might even be regular
(v) **L** is definitely not regular, and it may or may not be context-free
(vi) **L** is definitely not even context-free

Follow each answer with example languages **A** (and **B**, where appropriate) to back up the complexity claims inherent in your answer; and/or state some known closure property that reflects a bound on the complexity of **L**.

Example.) $\mathbf{L = A \cup B}$, where **A** and **B** are both context free, and definitely not regular

L can be characterized by **Property (ii)**, above.

L is context-free, since the class of context-free languages is closed under union.

L can be regular. For example,

$\mathbf{A = \{ \ a^n b^m \ | \ m \geq n \ \}}$, $\mathbf{B = \{ \ a^n b^m \ | \ m \leq n \ \}}$,

$\mathbf{L = A \cup B = \{ \ a^n b^m \ | \ n, m \geq 0 \ \}}$, which is regular since it can be represented by the regular expression **a*b***.

But **L** is in general not guaranteed to be regular, e.g., if we just make **A** and **B** the same context-free, non-regular set, then $\mathbf{L = A \cup A = A}$, which is not regular.

2. Language Closure under Various Operation

a.) **L = A $\cap$ B**, where **A** and **B** are both context-free, non-regular

(iv)

Regular: $\mathbf{A = a^n b^n, B = c^n d^n, L = \varnothing}$

Context-Free: $\mathbf{A = a^n b^n, B = a^n b^n, L = a^n b^n}$

Non-Context-Free: $\mathbf{A = a^n b^n c^*, B = a^* b^n c^n, L = a^n b^n c^n}$

b.) **L = A $\cap$ B**, where **A** is context-free, non-regular and **B** is regular

(ii)

Regular: $\mathbf{A = a^n b^n, B = \varnothing, L = \varnothing}$

Context-Free: $\mathbf{A = a^n b^n, B = a^* b^*, L = a^n b^n}$

Non-Context-Free: Not possible as CFLs are closed under intersection with regular

3. Show prfs are closed under Fibonacci induction

Fibonacci induction means that each induction step after calculating the base is computed using the previous two values. Here,
$f(0,x)$ = some base value;
$f(1,x)$ is based on $f(0,x)$ and 0 (an invented value for two steps back);
and for $y>1$, $f(y,x)$ is based on $f(y-1,x)$ and $f(y-2,x)$.

The formal hypothesis is:
Assume g and h are already known to be prf, then so is f, where
$f(0,x) = g(x)$;

$f(1,x) = h(f(0,x), 0)$; and

$f(y+2,x) = h(f(y+1,x), f(y,x))$

Proof is by construction

Fibonacci Recursion

Let K be the following primitive recursive function, defined by induction on the primitive recursive functions, g, h, and the pairing function.

$K(0,x) = B(x)$

$B(x) = < g(x), C_0(x) >$ // this is just $<g(x), 0>$

$K(y+1, x) = J(y, x, K(y,x))$

$J(y,x,z) = < h(<z>_1, <z>_2), <z>_1 >$

// this is $< f(y+1,x), f(y,x)>$, even though f is not yet shown to be prf!!

This shows K is prf.

f is then defined from K as follows:

$f(y,x) = <K(y,x)>_1$ // extract first value from pair encoded in K(y,x)

This shows it is also a prf, as was desired.

Fibonacci Recursion (simpler form)

Let K be the following primitive recursive function, defined by induction on the primitive recursive functions, g, h, and the pairing function.

$K(0,x) = \langle g(x), 0\rangle$ // this is pair $\langle f(0,x), 0\rangle$

$K(y+1, x) = \langle h(\langle K(y,x)\rangle_1, \langle K(y,x)\rangle_2), \langle K(y,x)\rangle_1\rangle$ // this is pair $\langle f(y+1,x), f(y,x)\rangle$,

This shows K is prf.

f is then defined from K as follows:

$f(y,x) = \langle K(y,x)\rangle_1$ // extract first value from pair encoded in K(y,x)

This shows it is also a prf, as was desired.