Assignment#3 Key

1. Consider L = {$a^n b^s c^t$ | s>n and t>s}.

Using the Pumping Lemma for CFLs, show **L** is not a Context-Free Language.

Assume **L** is a CFL

Let **N > 0** be from PL

Chose string $\mathbf{a^N b^{N+1} c^{N+2}}$

PL breaks into **uvwxz**, **|vwx| ≤ N** and **|vx| >0** and says **∀i≥0** $\mathbf{uv^iwx^iz}$

Case 1:. **vx** contains at least one **a**. Set **i=3**, then we at least **N+2 a**'s and only **N+2 c**'s and so string is not in **L.**

Case 2: vx contains no **a**'s. Set **i = 0**, then we still have **N a**'s but one or both of the **b**'s or **c**'s have been reduced and yet **N+1** and **N+2** are as small as they can be, so the new string is not in **L**.

These cases cover all possibilities and so **L** is not a CFL.

2. Case Analysis of Languages Closures

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

2a. Present arguments for the following case

L = B - A, where **A** is context-free, non-regular and **B** is regular

Can be Regular as in case where **B ∩ A = Ø** and so **L = B**, which is Regular

Can be a CFL as in case where **B = a*b***, **A =** a^nb^n and so **L =** a^nb^m**, n≠m**, which is a CFL (we have written a CFG for it previously)

Can be a CSL as in the case
B = {**x | x∈{a,b}*** and **|x|** is even**}**,
A = {yz | y,z∈{a,b}*, y≠z, |y|=|z|};
L = {ww | w∈{a,b}*) which is a CSL (we used PL to show this)
This is case **(iv)**

2b. Present arguments for the following case

L = A - B, where **A** is context-free, non-regular and **B** is regular

Can be Regular as in case where **B = Σ*** and so **L = ∅**, which is Regular

Can be a CFL as in case where **B ∩ A = ∅** and so **L = A**, a CFL

Since **A – B = A ∩ ~B**, Regular are closed under complement, and CFLs are closed under intersection with Regular, then must be no worse than a CFL.

This is case **(ii)**

2c. Present arguments for the following case

A ⊂ L, where **A** is Context-Free

Can be Regular as in case where **L = a*b*** and **A = {$a^n b^n$ | n ≥ 0 }**

Can be a CFL as in case where **L = A**

Can be a CSL as in the case **L = $a^n b^n c^n$** and **A = {$a^n b^n$c* | n ≥ 0 }**

To be honest **L** can be arbitrarily complex. For instance, consider **L = $a^n b^n c^{f(n)}$** where **f(n)** is any total mapping, maybe not even a computable one, then **A = {$a^n b^n$c* | n ≥ 0 } ⊂ L**

This is case **(iv)**

3. Show prfs are closed under halfway mutual induction

Halfway mutual induction means that each induction step after calculating the base is computed using the **floor((y+1)/2** value of the other function.

The formal hypothesis is:
Assume **g1**, **g2**, **h1** and **h2** are already known to be prf, then so are **f1** and **f2**, where
f1(x,0) = g1(x); f2(x,0) = g2(x)
f1(x,y+1) = h1(f2(x, floor((y+1)/2))); f2(x,y+1) = h2(f1(x, floor((y+1)/2))

Note tha // does the floor of division and it is a prf

Proof is by construction

3. Building and Accessing Values in a Trace

First, we recall how our pairing function works and that we can use it to encode arbitrarily long tuples. We can essentially think of a number as representing a stack with a head and a tail. The head is a single element, and the tail is the remaining elements of the stack.

$<x, y> = 2^x * (2y+1) - 1$; $<z>_1 = exp(z+1, 0)$; $<z>_2 = (((z + 1) // 2^{<z>_1}) - 1) // 2$

Given this, we want an accessor for any arbitrary element in such a k-tuple. We will start by providing a way to get the y-th tail of a tuple.

Item(z, 0) = $<z>_1$ // essentially the Head of the list, say of <a, b, c, d, e, 0>; 0 is bottom

Item(z, y+1) = Item($<z>_2$,y) // in above case if y =0, we get a;

// if y+1=1, we get Item(<b,c,d,e,0>,0) = b

// if y+1=2 we get Item(<b,c,d,e,0>,1) = Item(<c,d,e,0>,0) = c

3. Halfway Mutual Induction (Recursion)

F will do all computations in “parallel”

$F(x,0) = \langle\langle g1(x), g2(x)\rangle, 0\rangle$ // bases for both; creating a list of pairs

$F(x, y+1) = \langle\langle\ h1(\ \langle F(x,Item((y+1)//2))\rangle_2),\ h2(\langle F(x,Item((y+1)//2))\rangle_1))\rangle,\ F(x,y)\ \rangle$

F produces a list of pairs containing the pair **f1f2**, in its first and second components, respectively. The above shows **F** is a prf.

f1 and **f2**, are then defined from **F** by getting the first component of the y-th pair. That is, itself, a pair and so we then extract its first component for **f1** and second for **f2**.

$f1(x,y) = \langle\langle\ F(x,y)\ \rangle_1\rangle_1$

$f2(x,y) = \langle\langle\ F(x,y)\ \rangle_1\rangle_2$

This shows that **f1** and **f2** are also prf’s, as was desired.