Problem 1

Riken Maharjan

a) $F'(K,x) = F(K,0^n) \circ f(K,x)$

$f'(K,n)$ is not secure.

$f^1 : K \to 2K$ | $m_1 \| m_2$ — first K^{th} bit, last K^{th} bit

A

$m_1 \| m_2 := f'_K(x_1)$

$\Rightarrow$ $m_3 \| m_4 := f'_K(x_2)$ $\diamond$

return$(m_1 \overset{?}{=} m_3)$

f^1_real

$K \leftarrow \{0,1\}^n$

$f'_K(x)$:

return$(F(K,0^n) \| F(K,x))$

As $F(K,0^n)$ will always spit the same bits. Hence we can query two times and see if our first K^{th} bits from $(m_1 \| m_2)$ match the first K^{th} bits from $(m_3 \| m_4)$

This would not be possible if we were quering a real pseudorandom function.

$$\Pr[A \diamond f'\text{\_real} \to 1] - \Pr[A \diamond \overbrace{f^1\text{\_rand}}^{\text{just a doubling PRF.}} \to 1]$$

$$\Rightarrow \underset{\substack{\uparrow \\ \text{big}}}{1} - \frac{1}{2^K} \leftarrow \text{neglgible}$$

It is non-negligible

(b)

$$f^2(K,x) = F(K,x) \circ F(K,\bar{x})$$

$F^2(K,x)$ is a secure PRF.

<u>Given</u> $\mathcal{L}^{F}_{prf\text{-}real} \approx \mathcal{L}^{f}_{prf\text{-}rand}$

To prove $\mathcal{L}^{f^2}_{prf\text{-}real} \approx \mathcal{L}^{f^2}_{prf\text{-}rand}$

$\mathcal{L}^{F^2}_{prf\text{-}real}$

```
K <- {0,1}^λ
Query(x ∈ {0,1}^n)
return (f(K,x) || f(K,x̄))
```

$\approx$

```
A():
return
(Query'(x) || Query'(x̄))
```

$\diamond$

Just Moving Code

$\mathcal{L}^{f}_{prf\text{-}real}$

```
K <- {0,1}^λ
Query'(x ∈ {0,1}^n)
return (f(K,x))
```

$\approx$

```
A():
return
(Query'(x) || Query'(x̄))
```

$\diamond$

$\mathcal{L}^{f}_{prf\text{-}rand}$

```
T := empty array
Query'(x ∈ {0,1}^n)
if T[x] undefined
   T[x] <- {0,1}^out
return T[x]
```

Given $\mathcal{L}^{f}_{prf\text{-}real} \approx \mathcal{L}^{f}_{prf\text{-}rand}$

$\approx$

Just moving code

```
A():
T := empty array
Query(x ∈ {0,1}^n):
if T[x] undefined:
   a  <- {0,1}^out
   b  <- {0,1}^out
   T[x] : a || b
return T[x]
```

$\approx$

$\mathcal{L}^{f^2}_{prf\text{-}rand}$

```
A():
T := empty array
Query(x ∈ {0,1}^n):
if T[x] undefine
   T[x] <- {0,1}^2out
return T[x]
```

(c)

$$f^3(k, x) = f(0^\lambda, x) \oplus f(k, x)$$

Not a PRF

The key is 0^λ is known by the adversary. Hence $f(0^\lambda, x)$ is not a pseudorandom function. K needs to be sampled from a uniform keyspace to make our pseudorand

A

$m_1 \| m_2 := f^3_k(x)$:

return $(f(0^\lambda, x) \overset{?}{=} m_1)$

$\diamond$

$\mathcal{L}^3_{PRF\text{-}real}$

$k \leftarrow \{0,1\}^\lambda$

$f^3_k(x)$:

return $(f(0^\lambda, x) \| f(k, x))$

As the Adversary knows the PRF $f(0^\lambda, x)$ he can easily distinguish $\mathcal{L}^3_{PRF\text{-}real}$ from $\mathcal{L}^3_{PRF\text{-}random}$.

The advantage is non negligible.

$$\Pr[A \diamond \mathcal{L}^3_{PRF\_real} \to 1] - \Pr[A \diamond \mathcal{L}^3_{PRF\text{-}rand} \to 1]$$

$$\Rightarrow \quad \underset{\substack{\uparrow \\ \text{not neglig}}}{1} - \underset{\substack{\uparrow \\ \text{negligible}}}{\frac{1}{2^k}}$$

where $k = |m_1| = |m_2|$

Hence his advantage is negligible.

(a)

$f^4(k,x) = \underline{g(f(k,x))}$

This is a secure pseudorandom function

Given $L^{f}_{prf\text{-}real} \approx L^{f}_{prf\text{-}rand}$ and $L^{g}_{prg\text{-}real} \approx L^{g}_{prg\text{-}rand}$

To prove

$$L^{f4}_{PRf\text{-}real} \approx L^{f4}_{Prf\text{-}rand}$$

$L^{f4}_{Prf\text{-}real}$

```
k ← {0,1}^λ
query(x ∈ {0,1}^n)
return( g(f(k,x)) )
```

$\approx$

```
A( ):
return
  g(query'(x))
```

$\diamond$ $L^{f}_{Prf\text{-}real}$

```
k ← {0,1}^λ
query'(x ∈ {0,1}^n)
return(f(k,x))
```

$\approx$

```
A( ):
return
(g(query'(x)))
```

$\diamond$ $L^{f}_{Prf\text{-}rand}$

```
T := empty array
query'(x ∈ {0,1}^n)
if T[x] undefined
   T[x] ← {0,1}^out
return T[x]
```

Given $L^{f}_{Prf\text{-}real}$ & $L^{f}_{PRf\text{-}rand}$

$\approx$

```
T := empty array
query(x ∈ {0,1}^n)
if T[x] undefined
   a ← {0,1}^out
   T[x] := g(a)
return( T[x] )
```

a is uniformly random so g(x) can double it and also we know g(x) is a PRG which can be replaced with $\underline{L^{g}_{prg\text{-}rand}}$

Q4 pt 2

$\approx$

```
T := empty array
query(x ∈ {0,1}^n)
if T[x] undefined
    T[x] = query'()
return (T[x])
```

$\diamond$

$L^{q}_{prg\text{-}real}$

```
query'()
a ← {0,1}^out
return G(a)
```

moving the code

$\approx$

```
T := empty array
query(x ∈ {0,1}^n)
if T[x] undefined:
    T[x] = query'()
    return (T[x])
```

$\diamond$

$L^{q}_{prg\text{-}rand}$

```
query'():
z ← {0,1}^2out
return z
```

as $L^{q}_{prg\text{-}real} \approx L^{q}_{prg\text{-}rand}$

$L^{f_q}_{PRF\text{-}rand}$

$\approx$

```
T := empty array
query(x ∈ {0,1}^n)
if T[x] undefined:
    T[x] ← {0,1}^2out
return (T[x])
```

Hence $L^{f_q}_{PRF\text{-}rand} \approx L^{f_q}_{PRF\text{-}real}$

2) Problem

A'

```
Query(M_L, M_R):
  C_1 ← A.Challenge(M_L)
  K_1 ← A.findKey
  If Σ.Dec(K_1, C_1) =? m_L
     return 1
  else:
     return 0
```

A

$\diamond$

L_{cpa}^{Σ}

```
k ← KeyGen
Challenge(m ∈ Σ.M)
  c := Σ.Enc(k, m)
  return c
```

In the m_L world we would always return 1 whereas on in m_R world our adversary may not return 1. It returns 1 with negligible probability

Hence we can distinguish L_{cpa}^{Σ} left from L_{cpa}-right