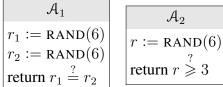
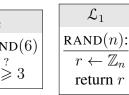
Exercise 2

2.1 Basics on libraries (5pt)

1. Below are two calling programs A_1, A_2 and two libraries L_1, L_2 with a common interface:



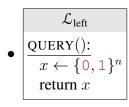




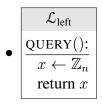
- What is $\Pr[A_1 \diamond L_1 \Rightarrow 1]$?
- What is $\Pr[A_1 \diamond L_2 \Rightarrow 1]$?
- What is $\Pr[A_2 \diamond \mathcal{L}_1 \Rightarrow 1]$?
- What is $\Pr[A_2 \diamond L_2 \Rightarrow 1]$?
- 2. In each problem, a pair of libraries are described. State whether or not $\mathcal{L}_{left} \equiv \mathcal{L}_{right}$. If so, show how they assign identical probabilities to all outcomes. If not, then describe a successful *distinguisher*.

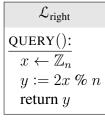
Assume that both libraries use the same value of n. Does your answer ever depend on the choice of n?

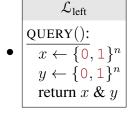
Note that \bar{x} denotes the bitwise-complement of x and x & y denotes the bitwise AND of the two strings:











$\mathcal{L}_{ ext{right}}$
QUERY():
$z \leftarrow \{0,1\}^n$
return z

2.2 Security of a modified One-time Pad (OTP) (3pt)

Suppose we modify one-time pad to add a few 0 bits to the end of every ciphertext:

$\mathcal{K} = \{0, 1\}^{\lambda}$ $\mathcal{M} = \{0, 1\}^{\lambda}$	KeyGen:	Enc(k,m):	Dec(k,c):
$\mathcal{M} = \{0, 1\}^{\lambda}$	$\overline{k \leftarrow \{0,1\}^{\lambda}}$	$c := k \oplus m$	$\overline{}$ remove last 2 bits of c
$\mathcal{C} = \{0, 1\}^{\lambda+2}$	return k	return $c 00$	$m := k \oplus c$
			return m

(In Enc, || refers to concatenation of strings.) Show that the resulting scheme still satisfies one-time secrecy. Your proof can use the fact that one-time pad has one-time secrecy.

2.3 Construction of a distinguisher (2pt)

Show that the following encryption scheme does not have one-time secrecy, by constructing a program that distinguishes the two relevant libraries from the one-time secrecy definition.

2.4* Size of the OTP key space (Bonus: +3pt)

Prove that if an encryption scheme Σ has $|\Sigma.\mathcal{K}| < |\Sigma.\mathcal{M}|$ then it cannot satisfy one-time secrecy. Try to structure your proof as an explicit attack on such a scheme (i.e., a distinguisher against the appropriate libraries).

You may consider Enc to be a deterministic function, as in the one-time pad. To obtain even more bonus points, prove this statement for randomized Enc. However, you may assume that Dec is deterministic.

Hint: The definition of interchangeability doesn't care about the running time of the distinguisher or the calling program. So even an exhaustive brute-force attack would be valid.