1. **Choose the correct answer and justify you answer.**

(1) (6%) Let $A = \{0^n 1^n 2^n | n \in \mathbb{N}\}$ and

(a) $A \subseteq B$, then B is decidable.

(b) $B \subseteq A$, then B is decidable.

(c) $B \leq_\tau A$, where recursive function τ is a reduction from A to B, then B is decidable.

The correct answer is ( ). ______

我喵，这样的题目形式我不解释都不行了……

（b） A 是可判定的，那么 B 就用同样的机器不久判定了吗？（a）错在 B 是 A 的超集，不一定是可判定的。（c）错在方向反了，A 归约到 B，那么 B 可判定那么 A 可判定，可是 B 本身我们不知道。

(2) (6%) Let A and B be any languages such that $A \leq_\tau B$. Under what conditions is it the case that $\overline{A} \leq \overline{B}$?

(a) Only when both A and B are decidable.

(b) Only when both A and B are recursively enumerable.

(c) Always.

The correct answer is ( ). ______

（a） 递归语言对补封闭，因此 B 可以归约到 A，B 的补当然可以归约到 A 的补，这个是显然的。如果 A 不是递归但是是递归可枚举，而 B 是递归的，那么（b）（c）都不成立。

(3) (6%) Just as we encoded Turing Machines as strings, we can also encode DFAs as strings. Let "M" be the encoding string of DFA M. Consider the following language $L^d_{DFA} = \{\text{"}M\text{"} | \text{"}M\text{"} \notin L(M)\}$. What can we say about L^d_{DFA} ?

(a) L^d_{DFA} is regular.

(b) L^d_{DFA} is not regular but it is decidable.

(c) L^d_{DFA} is not recursively enumerable.

The correct answer is ( ). ______

“M”∉L（M），说明 M 是一个自己不接受自己的编码的 DFA 的编码的集合。

我们把那个 L^d_{DFA} 简写成 L，否则我输入太麻烦了……构造图灵机 M*判定了 L，首先“M*”∉L。对于输入“M”，我们只需要 M*去模拟 M 收到输入“M”的情况。因为是 DFA，所以“M”的字符总会读完，读完的时候是终结状态就接受，否则就拒绝。拒绝说明“M”∉L(M)，M*给出 yes，否则反之。因此就知道选（b）了。

至于为什么不是 regular 的，直观说，一个 DFA 给出的判定是这个 DFA 是否接受这个字符串，而不是收到的编码的原 DFA 是否能接受自己的编码，它木有能力模拟和判断别人，所以木有 DFA 来接受它。

(4) **(6%)** Just as we encoded Turing Machines as strings, we can also encode PDAs as strings. Let "M" be the string encoding of PDA M. Consider the following language $L^d_{PDA} = \{\text{"}M\text{"} \mid \text{"}M\text{"} \notin L(M)\}$. What can we say about L^d_{PDA}?

(a) L^d_{PDA} is decidable.

(b) L^d_{PDA} is not decidable but it is recursively enumerable.

(c) L^d_{PDA} is not recursively enumerable.

The correct answer is (   ). ______________________________

只是 DFA 改成 PDA。不失一般性地我们假设这个 PDA 是非确定性的，然后只要一条路是通向终结状态并且栈最后变空了那么就算是 M 接受了“M”。我们构造非确定图灵机 M*判定 L。我们需要多条带，一条存着输入“M”，一条模拟 PDA 的堆栈。然后给定输入 M*模拟 M 对输入“M”的操作。因为 M*也是非确定的，所以每次可以非确定使用 PDA 的规则。如果其中一次成功了那么“M”∈L(M) 然后拒绝这个输入，否则全部情况都不接受那么 M*就接受输入。但是，非确定的 PDA 的其中一些运算不一定可以停机，它完全可以在某条路上死循环给不出接受的答案。因此，这是个 co-r.e. 的问题，选择（c）。

(5) **(6%)** Let A and B be disjoint, recursively enumerable languages. Further let $\overline{A \cup B}$ also be recursively enumerable. What can you say about A and B?

(a) It is possible that neither A nor B is decidable.

(b) At least one among A and B is decidable.

(c) Both A and B are decidable.

The correct answer is (   ). ______________________________

Disjoint 在这里应该是无不相交的意思。

（a）设 A 是{“M”a| 图灵机 M 在输入 a 上停机}，B 是{“M”b|图灵机 M 在输入 b 上停机}，这两个语言都可以用通用图灵机半判定，都是递归可枚举的，但是两者都不是可判定的。

2. **(12%)** Using the pumping theorem to show that

$$L_1 = \{w \in \{a, b\}^* \mid w \text{ has an equal number of } a'\text{s and } b'\text{s}\}$$

is not regular.

解析：用泵定理证明。设 L2={$a^n b^n$}，L2 包含于 L1。给定整数 k，考虑字符串 w=$a^k b^k$，w∈L2，我们可以把 w 重写为 w=xyz，且|xy|≤k，且 y≠e，即 y=a^i，i＞0.但是 xz=$a^{k-i} b^k$∉L2，与泵定理矛盾。所以 L1 不是正则的。

3. **(16%)**

(a) Construct a context-free grammar that generates language

$$L_2 = \{a^m b^n c^k | m, n, k \in \mathbb{N}, \text{ and } m + n \leq k\}.$$

(b) Construct a pushdown automata that accepts language L_2.

（a） G= （V，Σ，R，S）

V={a，b，c，S，S1，S2，S3}

Σ={a，b，c}

R={

S→S1

S1→aS1c

S1→S2

S2→bS2c

S2→S3

S3→S3c

S3→e

}

S 是起始符

（b）M=（K，Σ，Γ，Δ，p，F）

K={p，q}

Σ={a，b，c}

Γ={a，b，c，S1，S2，S3}

p 为初始状态

F={q}

Δ={

$(p, e, e)\ (q, S)$

$(q, e, S)\ (q, S_1)$

$(q, e, S_1)\ (q, aS_1c)$

$(q, e, S_1)\ (q, S_2)$

$(q, e, S_2)\ (q, bS_2c)$

$(q, e, S_2)\ (q, S_3)$

$(q, e, S_3)\ (q, S_3c)$

$(q, e, S_3)\ (q, e)$.

~~(q, e, S) (q, e)~~

$(q, a, a)\ (q, e)$

$(q, b, b)\ (q, e)$

$(q, c, c)\ (q, e)$.

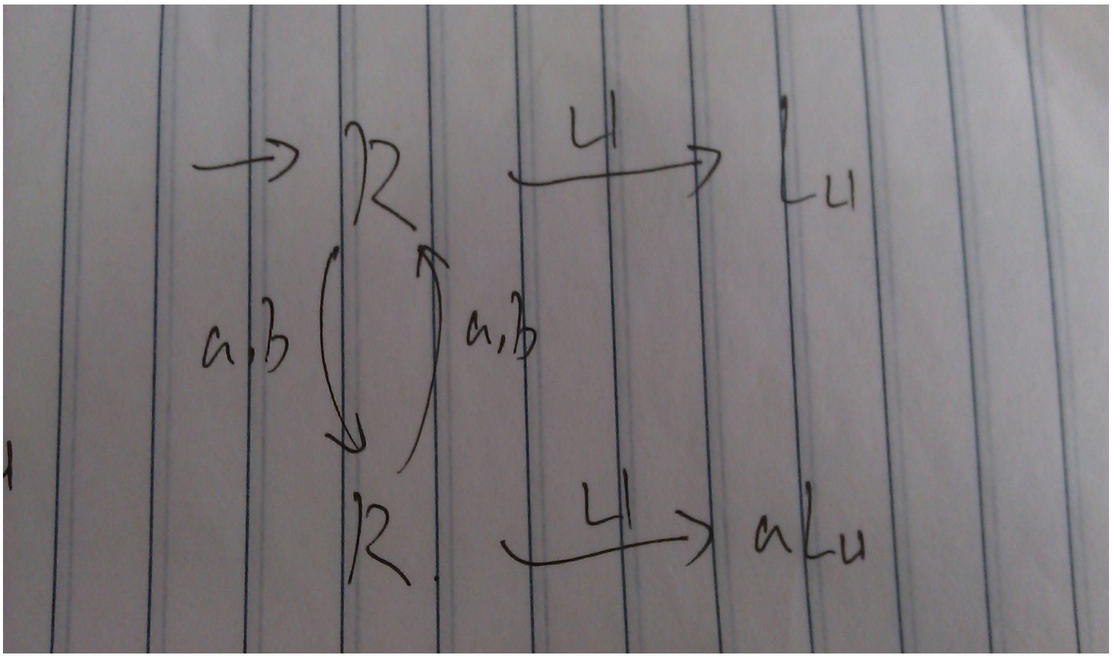
}

字迹潦草请见谅……

4. **(12%)** Construct a Turing machine that computes the funcion $f : \{a, b\}^* \to \{a, b\}^*$ given by

$$f(w) = \begin{cases} w, & \text{if the length of } w \text{ is even} \\ wa, & \text{if the length of } w \text{ is odd.} \end{cases}$$

When describing the Turing machines above, you can use the elementary Turing machines described in textbook. Always assume that the Turing machines start computation from the configuration $\rhd\sqcup w$ where $w \in \{a, b\}^*$ is the input string.



5. **(15%)** Classify whether each of the following languages are recursive, recursively enumerable-but-not-recursive, or non-recursively enumerable. Prove your answers.
   (a) $L_4 = \{\text{“}M\text{”} \mid \text{Turing machine } M \text{ halts on “}M\text{”}\}$;
   (b) $L_5 = \{\text{“}M\text{”} \mid \text{Turing machine } M \text{ does not halt on “}M\text{”}\}$.

（a）L4 可用通用图灵机半判定，所以是递归可枚举的但不是递归的，详细说明见课本 164 页。

（b）L5 不是递归可枚举的。可以用对角化原理证明，详细说明同见课本 164 页。

6. **(15%)**

(a) Give the definition of $\mathcal{NP}$-Complete problem.

(b) Describe carefully what the ingredients of an $\mathcal{NP}$-completeness proof are.

(c) Consider the following problem, called the **MAX-SAT** problem: Given a set F of clauses, and an integer k, is there a truth assignment that satisfies at least k clauses? Show that **MAX SAT** problem is $\mathcal{NP}$-complete.

（a）抄自定义 7.1.2，设语言 L 是Σ*的子集，如果
（1）L∈NP，并且
（2）对每个语言 L'∈NP，存在从 L'到 L 的多项式归约，
那么 L 称为是 NP 完全的。

（b）本人水平有限，不知道 NPC 完全的证明需要什么 ingredient。个人认为方法有两种，都是归约，给定问题多项式时间归约到一个 NPC 问题，那么这个问题也是 NPC，或者一个 NPC 问题多项式时间归约到给定问题，那么这个问题也是 NPC。前者用定理，后者用定义。
（c）MAX SAT 显然是 NP 问题。因为我们可以构造一个 NTM 在多项式时间内判定它。
我们尝试把可满足性问题归约到最大可满足性问题上面，于是就可以证明其是 NPC。
归约是这样的：给定可满足性带有 m 个子句的实例 F，那么 F 就相当于 K=m 的最大可满足实例，即原来最大可满足性满足 k 个子句，现在要求全部子句都满足，那么和可满足性问题等价。
显然，至少满足 F 的 m 个子句的真值复制当且仅当存在满足 F 的所有子句的真值赋值。
因此，可满足性问题可以归约到最大可满足性问题，于是后者是 NPC 的。（抄自定理 7.2.4 的证明）