这是我们（32 舍 388）参考老师的课堂内容整理的 11 年的答案，附带解析。由于时间仓促，错误必定不少，请各位见谅并吐槽指正。

G 版

(a) ( ) Let A be a recursive language and $B \subseteq A$, then B is decidable.

(b) ( ) Let A and B be two languages and τ is a recursive function. If $A \leq_\tau B$, then $\overline{A} \leq_\tau \overline{B}$.

(c) ( ) Just as Turing Machine's encoding, DFAs M can also be encoded as strings "M". Let $D_{DFA} = \{$"M"$|$ DFA M rejects "M"$\}$, then D_{DFA} is non-recursive.

(d) ( ) There are some languages that cannot be semi-decided by any Turing machine.

(e) ( ) Let $H_e = \{$"M"$|$ Turing machine M halts on $e\}$, then H_e is recursively enumerable, but not recursive.

(f) ( ) Let $H = \{$"M""w"$|$ Turing machine$M$ halts on $w\}$ and τ_1 and τ_2 are two recursive function. If $H \leq_{\tau_1} L$ and $H \leq_{\tau_2} \overline{L}$, then L is recursive enumerable but not recursive.

(g) ( ) Let A and B be two disjoint, recursively enumerable languages. If $\overline{A \cup B}$ is also be recursively enumerable, then both A and B are decidable.

(h) ( ) $\cup_{c,k}\mathbf{TIME}(c^{n^k}) = \cup_k\mathbf{NTIME}(n^k)$.

(i) ( ) Let A and B be two languages. If $A \leq_p B$ and $B \in \mathcal{P}$, then $\overline{A} \in \mathcal{P}$.

(j) ( ) Let A and B be two languages. If $A \leq_p B$ and $A \in \mathcal{NP}$, then $B \in \mathcal{NP}$.

(k) ( ) All languages in $\mathcal{NP}$-Complete are recursively enumerable.

(l) ( ) Let L be a language and $L \in \mathcal{NP}$. If there is a polynomial time reduction from language L to SAT, then L is $\mathcal{NP}$-complete.

(a) F 不一定。因为假设 A 是{Σ*}，B 是{M 在“M”上不停机}，符合题目要求，B 不是可判定的。

(b) T 如果 A 可以归约到 B，根据归约的定义，x∈A 的时候 t(x)∈B，反之，x∉A，即 x∈A 的补的时候 t(x)∉B，即 t(x)∈B 的补。

(c) F 因为 DFA 总有停机的时候，对于一个 DFA 输入“M”，这个 DFA 总会读完所有的字符然后给出 yes 或者 no，因此这个语言是递归的。老师课堂上说 D_{DFA} 是非正则的，这是可以用对角化定理证明 D_{DFA} 和每个正则语言都不一样，证明和课本 165 页中间的相似。

(d) T 这个非递归可枚举的语言就是课本 164 下面那个 H1 的补，{M 在“M”上不停机}

(e) T He 可以用通用图灵机半判定，但是无法判定。停机问题，即图灵机 M 在某个输入 w 上是否停机，无法判定，这里让 w=e 即可。

(f) F 根据定理 5.4.1，因为 H 不是递归的，所以 L 也不递归了。因为 H≤L 补，所以 H 补≤L。如果 L 是递归可枚举的，那么 H 补也是递归可枚举，那么 H 就不是递归可枚举的了，这和 H 本身是递归可枚举的事实是相悖的，因此 L 不是递归可枚举的。

(g) T 假设 M1 半判定 A，M2 半判定 B，M3 半判定 A∪B 的补。对于输入 w，交给 M1、M2 和 M3 一起做。如果 M1 接受，那么 w∈A，M2 接受，那么 w∈B，如果 M3 接受，那么 w∉(A∪B)，即不是 A 也不是 B。因为 A 和 B 是不相交的，因此 M1、M2 和 M3 的合体可以判定 A 或者 B，因此 A 和 B 都是可判定的，递归的。

(h) T 后面的是 NTIME，应该是不确定性计算的所需时间。所以，我们把 n^k 看作一体的，称为 a，那么确定计算时间是指数 c^a，非确定计算只要 a，也就是 n^k 了。

(i) T B 是 P，A 可以多项式时间归约到 B，所以 A 也是 P，P 对补封闭，所以命题正确。

(j) F 假设 A 是这样一个问题，给定一个整数集合和整数 k，求是否存在 k 个元素和大于 C。直观看我们可以用随机抽取 k 个元素然后算和然后如果其中一次大于 C 就接受。但是我们也可以通过，例如多项式时间的冒泡排序，把集合的整数降序排序，归约到问题 B，前 k 个元素的和是否大于 C，显然 B 是 P 的。所以，只能说如果 B 是 NP 的，A 可以归约到 B，那么 A 也是 NP 的。

(k) T 这个表述有点纠结，我们尚且给个 T。因为 NPC 是可计算的，有算法可解的，是递归的，当然也是递归可枚举的。

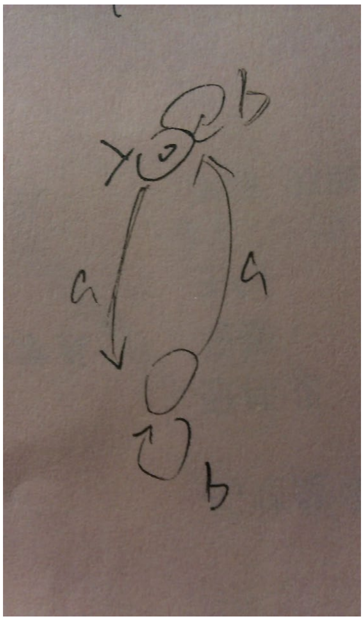
(l) F 我们直观点看，NPC 是所有 NP 问题都可以多项式归约到的问题，如果命题是正确的，那么 NP=NPC 了，所有 NP 问题都可以归约到任意一个 NP 问题上。如果这样，数学家不用花那么多精力寻找更多的 NPC 问题了。所以这么看命题是错误的。其实这个题目应该是个陷进，把 L 和 SAT 的位置换一下就对了。

2. **(12%) On FA and Regular Languages**

Say whether each of the following languages is regular or not regular? Prove your answers. Let $\#w(a)$ be the number of a in string w.

(a) $L_1 = \{xy \mid x, y \in \{a,b\}^*, \#x(a) = \#y(a)\}$

解析：是正则的。因为这相当于要求 xy 中 a 的个数是偶数就可以了，所以可以构造 DFA 来接受。



(b) $L_2 = \{xcy \mid x, y \in \{a,b\}^*, \#x(a) = \#y(a)\}$

解析：非正则。可以用泵定理证明。

假设 L_2 是正则的，则 $\exists k\in N$，$\forall xcy \in L_2$ 且 $|xcy|>k$ 都存在分解满足泵定理。

现考虑 $a^k cy \in L_2$，$\#y(a)=k$。

令 $a^k cy = uvw$，$|uv|\le k$，且 $v\ne e$，则 $v=a^i$

$\therefore uv^n w = a^{k+(n-1)i} cy$，显然 $n>1$ 时，$uv^n w \notin L_2$.

$\therefore$ L_2 不满足泵定理，不是正则的。

3. **(16%) On PDA and Context-Free Languages**

(a) Give a context-free grammar for the language

$$L_3 = \{xy^R \mid x, y \in \{a,b\}^*, |x| = |y| \text{ and } x \text{ and } y \text{ exactly differ in the first position }\}.$$

For example, $abbaabbb, abbbbbbb \in L_3$, but $aabbaa \notin L_3$.

解析：其实这是正则的，只有首尾字母不同并且长度是偶数就可以了。

G=（V，Σ，R，S）

V={a，b，S，S1，S2}

Σ={a，b}

R=

$S \to aS_1b$

$S \to bS_1a$.

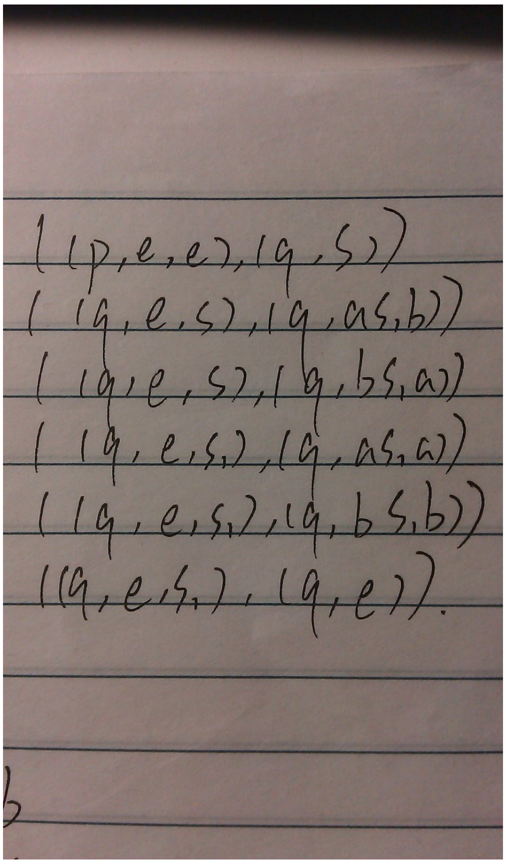
$S_1 \to aS_1a$

$S_1 \to bS_1b$

$S_1 \to e$.

(b) Design a PDA $M = (K, \Sigma, \Gamma, \Delta, s, F)$ accepting the language L_3.

下推自动机 M=（K，Σ，Γ，Δ，s，F）
K={p，q}
Σ={a，b}
Γ={a，b，S，S1，S2}
Δ={


((p,e,e),(q,S))
((q,e,S),(q,aS1b))
((q,e,S),(q,bS1a))
((q,e,S1),(q,aS1a))
((q,e,S1),(q,bS1b))
((q,e,S1),(q,e)).

((q,a,a),(q,e))
((q,b,b),(q,e))
}
S={p}
F={q}

(c) Transform the CFG of (a) into Chomsky norm form.

好像还是第一次考乔姆斯基范式。
生成乔姆斯基范式有三个阶段，把右面是大于 2 的分解掉，把右面是 e 的消掉变成右面是 1 个的，最后把右面是 1 个的消掉……
实在是太烦了，还要求闭包，这显然是给机器做的通用做法。大家不如用人脑只能直接凑一下吧，反正不是每年常考的题目……

$S \to aS'$ $\quad$ $S' \to S_1 b$

$S \to bS''$ $\quad$ $S'' \to S_1 a$

$S_1 \to aS_{11}$ $\quad$ $S_{11} \to S_1 a$

$S_1 \to bS_{12}$ $\quad$ $S_{12} \to S_1 b$

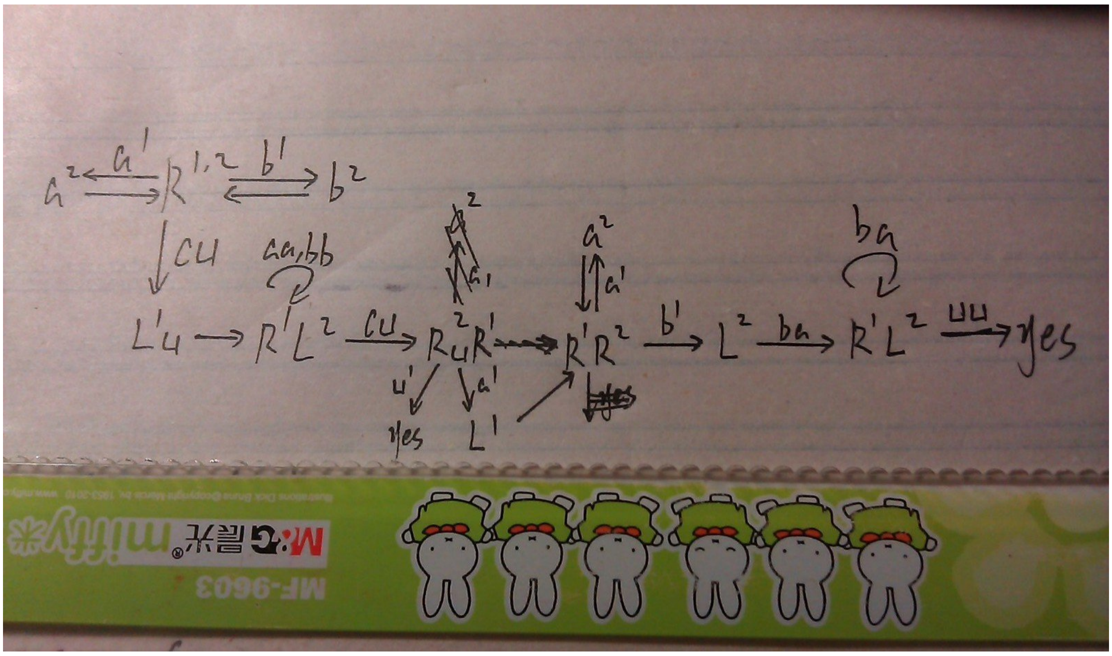
$S_1 \to aa$ $\quad$ $S_1 \to bb$

4. **(12%) On Turing Machines**

Construct a Turing machine that decides the following language:

$$L_4 = \{(xx^R)^R ca^n b^n | x \in \{a, b\}^*, n \in \mathbb{N}\}$$

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 $\triangleright\underline{\sqcup}w$ where $w \in \{a, b, c\}^*$ is the input string.



这是双带图灵机，字符右上角标记表示在第几条带操作，诸如 ab 这样的表示第一条带带头下是 a，第二条的是 b。

思路是，首先把 c 前面的内容复制到第二条带。然后第一条带带头到最左边，两条带同时移位检查 c 前面的内容是否对称。
如果对称了，那么检查后面。图上的小修改是因为 n 是 0 也可以，因此要先判断 c 后面是不是 a，如果是空格那么也是可以的。
如果输入字串的 c 后面是 a，那么把所有 a 复制到第二条带上，然后第一条带上读到 b 的时候开始检查 a 和 b 的数量是否相同。

5. (12%) On Primitive Recursive Functions

Show that the following function:

$$\varphi(x,y)=\begin{cases}(x+1)^y, & \text{if } x \text{ and } y \text{ are composite numbers,}\\ 0, & \text{otherwise.}\end{cases}$$

is primitive recursive.

解析：composite numbers 就是合数。
Φ（x,y）= (1~prime(x))(1~prime(y))exp(x+1,y)
其中 prime(x)就是判断 x 是否是质数，exp 就是求指数，~是非负减法。

$$prime(x)=(1\sim equals(x,2))\prod_{i=2}^{x-1}(rem(x,i)\neq 0)+equals(x,2)$$

Rem 是求余的函数，equals 是判断是否相等的函数，都是原始递归的，所以 prime 原始递归的

$$\exp(x,y)=(1\sim equals(y,0))(multi(\exp(x,y-1),x))+(equals(y,0))x$$

其中 multi 是乘法函数，所以 exp 也是原始递归的。
原始递归函数的合成也是原始递归的，所以原函数是原始递归的。

6. (12%) On Undecidability

Classify whether each of the following languages are recursive, recursively enumerable-but-not-recursive, or non-recursively enumerable. Prove your answers.

(a) {"M_1" "M_2" | There are some w such that both TMs M_1 and M_2 halt on w}

(b) {"M_1" "M_2" | There isn't any w such that both TMs M_1 and M_2 halt on w}.

（a）设这个语言是 L，L 是递归可枚举但非递归的，我们可以用通用图灵机 M 半判定 L。为了找到 M1 和 M2 都可以停机的字符串，我们逐轮逐轮来寻找。
第一轮，生成 w0，然后通用图灵机 UTM 模拟 M1 和 M2 在 w0 上计算一步。如果都没有停机，下一轮。
第二轮，生成 w1，然后通用图灵机 UTM 模拟 M1 和 M2 在 w0 和 w1 上计算两步。如果都没有停机，下一轮。
……
第 n 轮，生成 w_{n-1}，模拟 M1 和 M2 在 w0 到 w_{n-1} 上计算 n 步。木有停机就下一轮。
Wi 是按照字典序的Σ*的字符串。
因此，如果真的有 M1 和 M2 同时停机的字符串 w 的话，那么 UTM 的模拟最终停止然后接

受“M1”“M2”，否则不停机，因此 L 是递归可枚举的。
这里只是证明了递归可枚举，还没有证明其不是递归的。证明不是递归，可以通过把一个非递归的问题归约到 L 上，那么 L 也是非递归的。
设 He={“M”|图灵机 M 在 e 上停机}，显然这是不递归的。我们要把 L 归约到 He 上来。
考虑图灵机 C，C 启动前木有任何输入，启动之后首先在带上面自己写上“M1”“M2”，只有自己写上个 w，然后先模拟 M1 在 w 上的执行，如果停机了，那么擦掉结果，再模拟 M2 在 w 上的执行，如果 M1 和 M2 都在 w 上停机了，那么显然，C 在 e 上停机，否则不停机。所以，C 在空串上停机当且仅当 M1 和 M2 都在 w 上停机。因为 He 是不可判定的，所以 L 也是不可判定的。
（很绕很恶心，觉得不对劲的同学请参看课本上 5.4 节定理 5.4.2 的（b）的、同样绕同样恶心的证明。）

（b） 设这个语言是 L1，显然 L1 是 L 的补，L 是递归可枚举非递归，因此 L1 不是递归可枚举的。

7. **(12%) On $\mathcal{P}$ and $\mathcal{NP}$ Problems**

An *unequal assignment* to the variables of a Boolean formula φ in conjunctive normal form(CNF) makes at least one literal true and at least one literal false in every clause of φ. Let

$$\neq SAT = \{\varphi | \varphi \text{ is a 3-CNF formula with unequal assignment}\}.$$

(a) Prove that $\neq SAT$ is a $\mathcal{NP}$-problem.

(b) Prove that $\neq SAT$ is $\mathcal{NP}$-complete.

Hint: *Use a reduction from the 3SAT Problem, which is to decide the language*

$$3SAT = \{f | f \text{ is a Boolean formula in 3-CNF that is satisfiable}\}.$$

（a）我们可以设计一个 NTM M 在多项式时间之内计算≠SAT。对于给定输入 φ，M 首先随机生成一串各变量的真值赋值 F，然后代入到 φ 在 O(N)时间内（N 为输入长度）验证 F 是否满足每个子句至少一个文字为真且至少一个文字为负。所以≠SAT 是 NP 难的。

（b）我们要把 3SAT 问题多项式时间归约到≠SAT 上面。
归约的原始定义，是 L1 归约到 L2，那么如果 x∈L1 当且仅当 t(x)∈L2。（原谅我们用 t 替代 τ）因此证明的时候需要证明当和仅当。
首先是转换 t，对于输入的 3-CNF 范式 ψ，对于每个合取子句，我们这样转换

$$(y1 \vee y2 \vee y3) \Leftrightarrow (y1 \vee y2 \vee zi) \wedge (\overline{zi} \vee y3 \vee b)$$

假设有 n 个析取范式，那么我们增加 n 个变量 zi，b 取 F，于是得到 ψ（上面加一波浪线，各位意淫一下）

看下面的答案解析之前，各位哥哥姐姐且听我等一劝：这题目只占 4 分，丢卒保车更好。

U 版

1. **(24%) Determine whether the following statements are true or false. If it is true fill a ◯ otherwise a × in the bracket before the statement.**

   (a) ( ) Language $\{xyz|x, y, z \in \{a, b\}^*, y = y^R\}$ is not regular.

   (b) ( ) Let A and B be two languages. If A is regular and B is context free, then $A \oplus B$ is context free, where $A \oplus B = (A - B) \cup (B - A)$.

   (c) ( ) Language $\{a^m b^n c^k | m, n, k \in \mathbb{N} \text{ and } n \neq m + k\}$ is context free.

   (d) ( ) Language $\{xcycz|x, y, z \in \{a, b\}^* \text{ and } 3|x| = 2|z|\ \}$ is not context free.

   (e) ( ) Just as Turing Machine's encoding, Every PDA M can also be encoded as strings "M", then the language {"M"| PDA M rejects "M"} is recursive.

   (f) ( ) Nondeterministic Turing Machines can accept more languages than deterministic Turing Machines.

   (g) ( ) Language {"M" : Turing machine M halts on at least one input} is recursively enumerable.

   (h) ( ) There's a function φ such that φ can be computed by some Turing machines, yet φ is not a primitive recursive function.

   (i) ( ) Language {"M"|M is a Turing machine } is uncountable infinite.

   (j) ( ) There exists a language L such that both L and $\overline{L}$ are semi-decidable.

   (k) ( ) Language {"M""w"| Turing machine M halts on input w } is recursively enumerable, but not recursive.

   (l) ( ) Language {"M"| Turing machine M halts on empty string} is not recursively enumerable.

（a） F 令 y=e，然后整个语言就成了{a，b}*，所以是正则的。

（b） F 实在不会……这个要举反例，我们都木有想到，各位想到了可以告诉贺剑峰~

（c） T 可以构造一个 PDA 来接受。读到 a 压栈，读到 b 弹出 a 或者空栈了压入 b，读到 c 弹出 b，如果最后栈空了那么不接受。

（d） F 同样可以构造一个 PDA 来接受，读到一个 x 的字符压栈三个字母，读到一个 z 的字符出栈三个字母。

（e） F PDA 同样有停机问题，不一定可以停机，因此不是递归的。

（f） F 确定型图灵机和非确定的都是计算能力等价的。

（g） T 可以用通用图灵机半判定。通用图灵机随机生成一个输入 w，然后模拟 M(w)，如果接受那么 yes，否则就不停机。

（h） T 这句话的意思是，存在非原始递归的可计算函数，因为原始递归函数是 μ 递归函数的子集，而 μ 递归的函数才是可计算的函数，因此表述正确。

（i） F 我们可以用有限的字符来对图灵机进行编码，而有限集合的幂集是可数无穷的，因此图灵机的个数也是可数无穷个。

（j） T 语言是递归的当且仅当其本身和其补都是递归可枚举的，因此如果 L 是递归的，那么其本身和其补都是可半判定的。

（k） T 课本 5.3 节已说明

（l） F 和（k）相近，是非递归的递归可枚举

2. **(16%)** Let

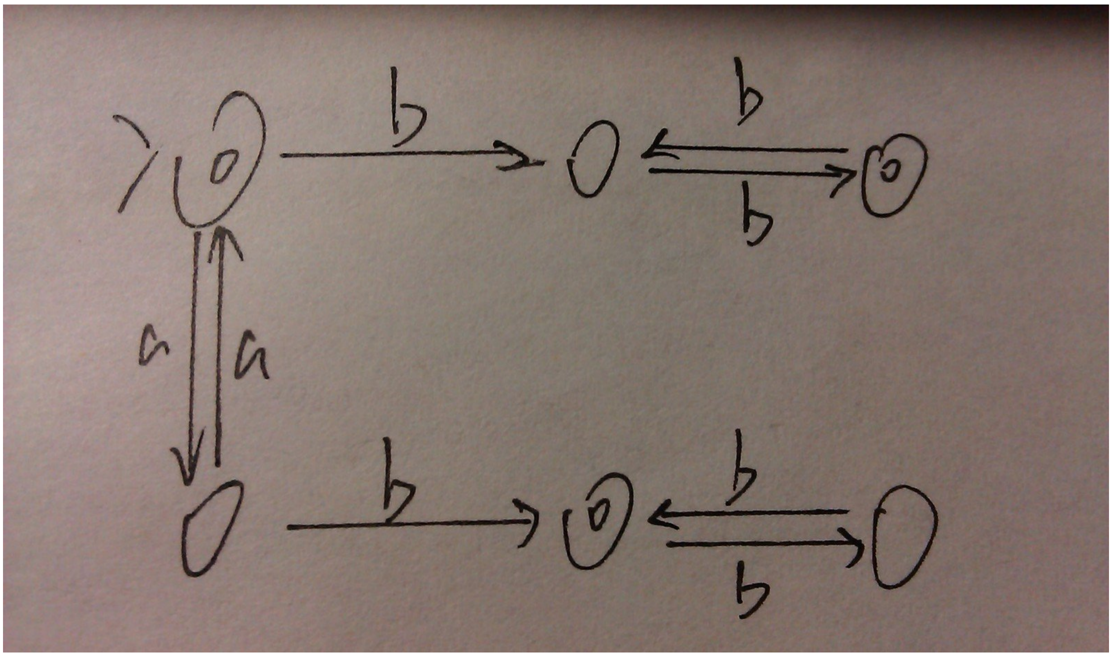
$$L_1 = \{a^m b^n | m, n \in \mathbb{N} \text{ and } m \equiv n \mod 2\}$$

(a) Give a regular expression for the language L_1.

（a） (aa)*(bb)* ∪ (aa)*a(bb)*b

(b) Construct a finite automata that accepts L_1.

(b)



3. **(10%)** Prove that $L_2 = \{a^m b^{2n} c a^n b^{3m} | m, n \in \mathbb{N}\}$ is not regular by applying the Pumping Theorem.

根据泵定理,假设 L2 是正则的,存在满足泵定理的整数 m。对于 w∈L2,考虑 $w=a^m b^{2n} c a^n b^{3m}$，$|w| \geq m$，令 w=xyz，且$|xy| \leq m$，$y \neq e$，那么 $y=a^k$，显然 $xy^i z \notin L2$，所以 L2 不是正则的。

4. **(20%)** Let $L_3 = \{a^m b^{2n} c a^n b^{3m} | m, n \in \mathbb{N}\}$.

(a) Construct a context-free grammar that generates the language L_3.

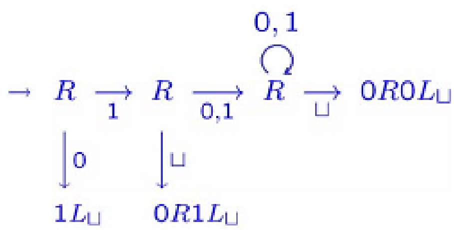
(b) Construct a pushdown automata that accepts L_3.

这是显然的，函数分段操作是原始递归的，乘法运算也是原始递归的，因此 f(x)是原始递归的。

(a)
$S \to S_1$
$S_1 \to aS_1bbb$
$S_1 \to S_2$
$S_2 \to bbS_2a$
$S_2 \to C$

(b)
$((p,e,e),(q,S))$
$((q,e,S_1),(q,aS_1bbb))$
$((q,e,S_1),(q,S_2))$
$((q,e,S_2),(q,bbS_2a))$
$((q,e,S_2),(q,c))$
$((q,a,a),(q,e))$
$((q,b,b),(q,e))$
$((q,c,c),(q,e))$

5. **(20%)** Let the following Turing machine M computes a function $f(x) : \mathbb{N} \mapsto \mathbb{N}$, where $x \in \mathbb{N}$ is represented by binary string without redundant leading 0's, and the initial configuration in form of $\triangleright\underline{\sqcup}x$.



(a) Give the configurations when M starts from the initial configuration $\triangleright\underline{\sqcup}1011$.

最后得到 101100，中间的格局大家自己推吧^_^

(b) Try to give the function $f(x)$ computed by the TM M.

X=0 时，f(x)=1
X=1 时，f(x)=101
其他，f(x)=4x

(c) Show that the function $f(x)$ is primitive recursive.

这是显然的，函数分段操作是原始递归的，乘法运算也是原始递归的，因此 f(x)是原始递归的。

6. **(10%)** Let $H_1 = \{\text{“}M\text{”} \mid \text{Turing Machine } M \text{ halts on “}\text{“}M\text{”}\}$. Show that H_1 is recursively enumerable. An informal description suffices.

可以用通用图灵机 UTM 半判定，对于输入“M”，UTM 模拟 M(“M”)的计算，如果 M 在“M”上停机那么 UTM 停机返回 yes，如果 M 在“M”上不停机那么通用图灵机也不会停机。因此这个语言是递归可枚举的。