1. [18 pts] Problem reduction:
   (a) What does it mean that a language $A$ is reducible to another language $B$?
   
   [5pts]
   Let $\Sigma_1$, $\Sigma_2$ be the alphabets of $A$ and $B$, respectively. Then $A$ is said to be reducible to $B$ if there is a computable function $f: \Sigma_1^* \rightarrow \Sigma_2^*$ such that $\forall x \in \Sigma_1^*$ $x \in A \Longleftrightarrow f(x) \in B$.

   (b) [5pts] Suppose $A$ is reducible to $B$. Explain why if $A$ is not Recursive then neither is $B$.
   It suffices to show that if $B$ is recursive then so is $A$.
   Let $M$ be any total TM accepting $B$, and $F$ any total TM computing $f$.
   Since $B$ is supposed to be recursive and $f$ is computable, both machines must exist. Now consider the composite machine $F \cdot M$, which on input $x$, first runs $F$ to get a result $f(x)$ and then feeds $f(x)$ to $M$ and accepts (or rejects) if $M$ accepts (or rejects) on $f(x)$.
   For any input $x$, if $x \in A$, then $f(x) \in B$ and then $M$ accepts $f(x)$ and hence $F \cdot M$ accepts. On the other hand, if $x \notin A$, then $f(x) \notin B$ and then $M$ rejects and hence $F \cdot M$ rejects. As a result, we know that $F \cdot M$ is a total TM which accepts $A$ and hence $A$ is recursive.

   (c) Let $\text{TH} = \text{def} \{ "M" \mid \text{Turing machine } M \text{ is a total machine.} \}$ and $\text{SH} = \text{def} \{ "M_1", "M_2" \mid \text{Turing machine } M_1 \text{ and } M_2 \text{ halt on the same set of inputs} \}$.
   Describe how to reduce $\text{TH}$ to $\text{SH}$. As a result, since $\text{TH}$ is not decidable(recursive), the problem of deciding if two Turing machines halt on the same set of inputs is not decidable.

   Let $M_2$ be a constant machine which halts on all inputs. (This machine can have only one non-final state and goes to the accept state immediately after the first action.) Then for any input "M", let $f("M") = "M" \cdot "M_2"$. Obviously, such function $f$ is computable. Moreover, if $M$ is total(i.e., "$M" \in \text{TH}$), then both $M$ and $M_2$ halt on all inputs(i.e., "$M" \cdot "M_2" \in \text{SH}$). But if $M$ is not total(i.e., "$M" \notin \text{TH}$), then there is an input $x$ which make machine $M_2$ halt but does not make $M$ halt, as a result $M$ and $M_2$ do not halt on the same set of inputs(i.e., "$M" \cdot "M_2" \notin \text{SH}$). Accordingly, we conclude that $f$ is a reduction from $\text{TH}$ to $\text{SH}$. 
2. (10 pts) Fill in the following table about Chomsky hierarchy.

<table>
<thead>
<tr>
<th>language type</th>
<th>language class name</th>
<th>recognition machine model</th>
<th>grammar model</th>
</tr>
</thead>
<tbody>
<tr>
<td>type 3</td>
<td>Regular Language</td>
<td>Finite automata (FA)</td>
<td>right-linear grammar</td>
</tr>
<tr>
<td>type 2</td>
<td>context-free language</td>
<td>Pushdown Automata</td>
<td>CFG</td>
</tr>
<tr>
<td>type 1</td>
<td>recursive language</td>
<td>LBA(Linear-bounded automaton)</td>
<td>Context-sensitive Grammar</td>
</tr>
<tr>
<td>type 0</td>
<td>r.e. language</td>
<td>Turing Machine</td>
<td>GPSG</td>
</tr>
</tbody>
</table>

3. (15 pts) We want to apply the game-theoretic version of the pumping lemma for CFL to show that the language $L = \{ a^n b^n c^m \mid m > n \}$ is not context-free. Fill in the following procedure to derive a winning strategy for Y:

(a) Suppose D pick a number $k > 0$.

(b) Y choose a string $z = a^{k+i} b^k c^k$ ________. What is the restriction the selected $z$ must obey?
   \[ z \in L \text{ and } |z| \geq k \]

(c) Suppose D decompose $z$ into $uvwxy$. What are the constraints that decomposition $uvwxy$ must satisfy:
   \[ |vwx| \leq k \text{ and } |vx| > 0 \]

(d) How should Y choose $i$:
   Y chooses $i = 0$ if $vwx$ contains $c$, and chooses $i = 2$ if $vwx$ does not contain $c$.

Under what condition will we say that Y wins the game?
   \[ uv^iwx^iy \not\in L \]

Explain why Y always win the game according to the strategy you wrote at step (b) and (d).

If $i = 0$, then $vwx$ contains $c$ and $vwx$ does not contain any a’s since $|vwx| \leq k$. Then $uv^iwx^iy$ contains $k$ a’s but contains fewer than $k+1$ c’s and hence $uv^iwx^iy \not\in L$.

On the other hand, if $i = 2$, then all c’s appear in $y$, and hence the number of either a’s or b’s or both in $uv^iwx^iy$ is increased to more than $k$ but the number of c’s remains $k+1$. As a result, $uv^iwx^iy \not\in L$. 
4. [10 pts] Consider the following Post corresponding system:

\[ CP = [(10, 1010), (0, 1), (101, 1), (11, 1)] \]

Find two CFG \( G_1 \) and \( G_2 \) such that \( CP \) has a solution \( x \) with corresponding solution index \( y \) if and only if \( L(G_1) \) and \( L(G_2) \) has a common member of the form \( x\#y^{-1} \), where \( y^{-1} \) is the reverse of \( y \).

Solution:

\( G_1: S_1 \rightarrow 10\#1 | 0\#2 | 101\#3 | 11\#4 \\
\quad 10S_11 | 0S_12 | 101S_13 | 11S_14 \)

\( G_2: S_2 \rightarrow 1010\#1 | 1\#2 | 1\#3 | 1\#4 \\
\quad 1010S_21 | 1S_22 | 1S_23 | 1S_24 \)

5. [10pts] Design a total standard Turing machine \( M \) with 4 states (including accept state \( t \) and reject state \( r \)) to accept the language \{ \( x \in \{a,b\}^* \mid x \) contains an even number of \( a \)'s and zero or more \( b \)'s. \}. The machine should have the tape alphabet ‘a’, ‘b’, ‘ ’ and ‘ ‘(blank).

(a) Draw a state transition diagram for \( M \). Note that all states must be labeled and the start, accept and reject states must be labeled as ‘s’, ‘t’ and ‘r’, respectively. [8pts]

(b) Fill in the following transition table according to the Turing machine you drew at (a). [7pts]

<table>
<thead>
<tr>
<th>state ( \rightarrow ) symbol</th>
<th>a</th>
<th>b</th>
<th>[ (left-end)</th>
<th>● (blank)</th>
</tr>
</thead>
<tbody>
<tr>
<td>( s )</td>
<td>p, R</td>
<td>s, R</td>
<td>s, R</td>
<td>t, x</td>
</tr>
<tr>
<td>( p )</td>
<td>s, R</td>
<td>p, R</td>
<td>x (don’t care)</td>
<td>r, x</td>
</tr>
<tr>
<td>( tF )</td>
<td>- (no action)</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>( rF )</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>
6. [12 pts] Let G be a context-free grammar consisting of the following rules:
\[ S \to aB | bA \quad A \to aS | bA \quad B \to bS | aA | a \]
where S is the start symbol. We want to find a push down automata to accept \( L(G) \).
Let \( M = (\{q\}, \{a, b\}, \{A, B, S\}, \delta, q, S, \{\epsilon\}) \) be a single-state NPDA for the language \( L(G) \) that accepts by empty stack. Fill in the following table for the transition relation \( \delta \). Note: since there is only one state, you need only specify the stack part action in the table entry. Also note that since the machine is not deterministic, some table entries may have more than one action.

<table>
<thead>
<tr>
<th>(state,input) \ stack</th>
<th>S</th>
<th>A</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>(q,a)</td>
<td>B</td>
<td>S</td>
<td>( \epsilon, BA )</td>
</tr>
<tr>
<td>(q,b)</td>
<td>A</td>
<td>( \epsilon, AB )</td>
<td>A</td>
</tr>
</tbody>
</table>

7. [20 pts] True or false questions. Note: total = halts on all inputs; r.e. set = accepted by a TM; recursive set = accepted by a total TM.
   (a) It is decidable whether a given CFG G is empty (i.e., \( L(G) = \emptyset \)).
   (b) It is decidable whether a given CFG G is universal (i.e., \( L(G) = \Sigma^* \)).
   (c) It is decidable whether two input CFGs can generate a common terminal string (i.e., \( L(G_1) \cap L(G_2) \neq \emptyset \)).
   (d) Three is a semi-Thue system the word problem of which is not decidable.
   (e) \( \{a^k b^k c^k | k \geq 0 \} \) is a context sensitive language.
   (f) The language generated by a CFG in Chomsky normal form could not contain the empty string.
   (g) If L is a CFL, then so is its complement.
   (h) The halting problem for universal Turing machine is neither recursive nor r.e.
   (i) If A is context free, then so is \( \{xx | x \in A \} \).
   (j) If L is not r.e., then its complement must not be recursive.

   solution: oxxooxxxoo

8. [6 pts] Let G be the context free grammar (CFG):
\[ S \to aSbb | T \quad T \to bTaa | S | \epsilon \]
Find a grammar G1 which has no \( \epsilon \)-rule and \( L(G1) = L(G) - \{ \epsilon \} \).

   Ans: \( S \to aSbb | abb | T \quad T \to bTaa | baa | S \)
9. [9 pts] Let G be the CFG: \( S \rightarrow aAc \mid aSc \quad A \rightarrow b \mid bA \)

Find a grammar equivalent to G which is in Chomsky normal form (cnf).

\[
\begin{align*}
S & \rightarrow aA \mid aS \\
A & \rightarrow bA \\
aA & \rightarrow aA \\
aS & \rightarrow aS \\
a & \rightarrow a \\
b & \rightarrow b \\
c & \rightarrow c
\end{align*}
\]

10. [20 pts] Consider the grammar:

\[
S \rightarrow ST \mid a \\
T \rightarrow BS \\
B \rightarrow b
\]

(a) The input ‘ababa’ is a member of \( L(G) \). Draw a parse tree for it. [5pts]

(b) Give a left-most derivation for the input ‘ababa’. [5pts]

\[
S \rightarrow ST \rightarrow aT \rightarrow aBS \rightarrow abS \rightarrow abST \rightarrow abaT \rightarrow abaBS \rightarrow ababS \rightarrow ababa
\]

(c) Run the CKY algorithm on the input string ‘ababa’. Fill the results in the following table.

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>S</td>
<td>-</td>
<td>S</td>
<td>-</td>
<td>S</td>
<td>-</td>
</tr>
<tr>
<td>1</td>
<td>T</td>
<td>-</td>
<td>T</td>
<td>B</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>S</td>
<td>-</td>
<td>S</td>
<td>-</td>
<td>S</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>T</td>
<td>B</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>