1. Let set $A$ be infinite recursive, $B$ be non-recursive and $C$ be non-re. Using the terminology $(REC)$ recursive, $(RE)$ re, non-recursive, $(NR)$ non-re (possibly co-re), categorize each set by dealing with the cases I present, saying whether or not the set can be of the given category and briefly, but convincingly, justifying each answer (BE COMPLETE). You may assume sets like $\mathbb{N}$ are infinite REC; $K$ and $K_0$ are RE; and $TOTAL$ is non-re. You may also assume, for any set $S$, the existence of comparably hard sets $S_E = \{2x | x \in S\}$ and $S_D = \{2x+1 | x \in S\}$.

   a.) $A + B = \{ x | x = y + z, \text{for some } y \in A \text{ and some } z \in B \}$

      REC: $A = \mathbb{N}$, $B = K_E$, $A + B = \{ x | x \geq \min y \in K \}$.

      This is the complement of a finite set and is hence decidable as the finite set is.

   b.) $A \cap C = \{ x | x \in A \text{ and } x \in C \text{ and } x \notin A \}$

      RE: $A = E = \{2x | x \in \mathbb{N}\}$, $C = TOTAL_D \cup K_E$.

      $A \cap C = K_E$ which is RE.

2. Choosing from among (REC) recursive, (RE) re non-recursive, (coRE) co-re non-recursive, (NR NC) non-re/non-co-re, categorize each of the sets in a) through d). Justify your answer by showing some minimal quantification of some known recursive predicate.

   a) $A = \{ <f,g> | \exists x \varphi(x) \downarrow \text{ and } \varphi_g(x) = \varphi(x) \}$.

      $\exists x, t > [STP(f,x,t) \& STP(g,x,t) \& Value(f,x,t) = Value(g,x,t)]$ \hspace{1cm} RE

   b.) $B = \{ f | \text{range}(\varphi_f) \text{ is empty} \}$

      $\forall x, t > \lnot STP(f,x,t)$ \hspace{1cm} co-RE

   c.) $C = \{ <f, x> | \varphi_f(x) \downarrow \text{ but takes at least 10 steps to do so} \}$

      $\exists [STP(f,x,t) \& \lnot STP(f,x,9)]$ \hspace{1cm} RE

   d.) $D = \{ f | \varphi_f \text{ diverges for some value of } x \}$

      $\exists x \forall t \lnot STP(f,x,t)$ \hspace{1cm} NRNC

3. Looking back at Question 1, which of these are candidates for using Rice’s Theorem to show their unsolvability? Check all for which Rice Theorem might apply.

   a) $\checkmark$ \hspace{1cm} b) $\checkmark$ \hspace{1cm} c) \hspace{1cm} d) $\checkmark$

4. Let $S$ be an arbitrary semi-decidable set. By definition, $S$ is the domain of some partial recursive function $g_s$. Using $g_s$, constructively show that $S$ is the range of some partial recursive function, $f_S$. No proof is required; just the construction is needed here.

   $f_S(x) = x \ast \exists t \{ STP(x, g_s, t) \}$ or

   $f_S(x) = x \ast (g_s(x) - g_s(x) + 1)$
5. Using the definition that $S$ is recursively enumerable iff $S$ is the range of some effective procedure $f_S$ (partial recursive function), prove that if both $S$ and its complement $\sim S$ are recursively enumerable (using semi-decision effective procedures $f_S$ and $f_{\sim S}$) then $S$ is decidable. To get full credit, you must show the characteristic function for $S$, $\chi_S$, in all cases. Also, be sure to discuss why your $\chi_S$ works.

Define $\chi_S(x) = f_S(\mu <y, t> | (STP(f_S, y, t) & (VALUE(f_S, y, t) = x)) \text{ or } (STP(f_{\sim S}, y, t) & (VALUE(f_{\sim S}, y, t) = x)) )$

If $x \in S$ then $\exists y f_S(y) = x$ and so $\chi_S(x) = 1$ (true)

If $x \notin S$ then $\exists y f_{\sim S}(y) = x$ and so $\chi_S(x) = 0$ (false)

Thus, $\chi_S(x)$ meets our requirements.

6. Rice’s Theorem deals with attributes of certain types of problems $P$ about partial recursive functions and their corresponding sets of indices $S_P$. The following image describing a function $f_{x,y,r}$ is central to understanding Rice’s Theorem.

Explain the meaning of this by indicating:

a.) What assumption do we make about what kind of functions are not in $P$?

We assume no function with empty domain has property $P$.

b.) What is $r$, how is it chosen and how can we guarantee its existence?

$r$ is the index of some function with property $P$. One must exist since $P$ is non-trivial.

c.) Using recursive function notations, write down precisely what $f_{x,y,r}$ computes for the Strong Form of Rice’s Theorem.

$$f_{x,y,r}(z) = \phi_x(y) - \phi_y(y) + \phi_r(z)$$

How does this function $f_{x,y,r}$ behave with respect to $x,y$ and $r$, and how does that relate to the original problem, $P$, and set, $S_P$?

If $\phi_r(y) \downarrow$ then $f_{x,y,r}(z) = \phi_r(z) \forall z$ and $f_{x,y,r} \in S_P$.

If $\phi_r(y) \uparrow$ then $f_{x,y,r}(z) \uparrow \forall z$ and $f_{x,y,r} \notin S_P$.

Thus, we could decide the halting problem if we could decide membership in $S_P$, so $P$ is an undecidable problem.
7. Define \( \text{NAT} = \{ f \mid \text{range}(f) = \mathbb{N} \} \). That is, \( f \in \text{NAT} \) iff \( f \)'s range includes every natural number.

a.) Show some minimal quantification of some known recursive predicate that provides an upper bound for the complexity of \( \text{NAT} \).

\[
\forall k \exists <x, t> \ [\text{STP}(f, x, t) \&\& (\text{Value}(f, x, t) = k)]
\]

b.) Use Rice’s Theorem to prove that \( \text{NAT} \) is undecidable.

First, \( \text{NAT} \) is non-trivial as the identity, \( I(x)=x \), is in \( \text{NAT} \) and the Constant Zero, \( Z(x)=0 \), is not.

Second, let \( f \) and \( g \) be arbitrary indices of arbitrary effective procedures, such that \( \text{range}(f) = \text{range}(g) \).

\( f \) is in \( \text{NAT} \) iff \( \text{range}(f) = \mathbb{N} \) iff \( \text{range}(g) = \mathbb{N} \).

This means \( \text{NAT} \) satisfies both properties of the weak form of Rice’s Theorem associated with ranges and is therefore undecidable.

c.) Show that \( \text{TOT} \leq_m \text{NAT} \), where \( \text{TOT} = \{ f \mid \forall x \ \varphi(f)(x) \downarrow \} \).

Let \( f \) be arbitrary. Define an algorithmic mapping \( G_f \) from indices to indices as \( G_f(x) = f(x) - f(x) + x \).

Now, \( G_f(x) = I(x) \) (the Identity function) iff \( f \in \text{TOTAL} \) and \( \exists x \notin \text{range}(G_f) \) iff \( f \notin \text{TOTAL} \). This will be any \( x \) where \( \varphi(f)(x) \uparrow \).

Thus, \( f \) is in \( \text{TOT} \) iff \( G_f \) is in \( \text{NAT} \). Thus, \( \text{TOTAL} \leq_m \text{NAT} \).

8. Why does Rice’s Theorem have nothing to say about the following? Explain by showing some condition of Rice’s Theorem that is not met by the stated property.

\( \text{AT-LEAST-LINEAR} = \{ f \mid \forall y \ \varphi_0(y) \text{ converges in no fewer than } y \text{ steps } \} \).

We can deny the 2\textsuperscript{nd} condition of Rice’s Theorem since \( Z \), where \( Z(x) = 0 \), implemented by the TM \( R \) converges in one step no matter what \( x \) is and hence is not in \( \text{AT-LEAST-LINEAR} \).

\( Z' \), defined by the TM \( \mathcal{R} \ R \), is in \( \text{AT-LEAST-LINEAR} \).

However, \( \forall x \ [ Z(x) = Z'(x) ] \), so they have the same I/O behavior and yet one is in and the other is out of \( \text{AT-LEAST-LINEAR} \), denying the 2\textsuperscript{nd} condition of Rice’s Theorem.

9. Consider the following set of independent tasks with associated task times:

\((T_1,4), (T_2,5), (T_3,2), (T_4,7), (T_5,1), (T_6,4), (T_7,8)\)

Fill in the schedules for these tasks under the associated strategies below.

Greedy using the list order above:

<table>
<thead>
<tr>
<th>T1</th>
<th>T1</th>
<th>T1</th>
<th>T1</th>
<th>T3</th>
<th>T5</th>
<th>T6</th>
<th>T6</th>
<th>T6</th>
<th>T7</th>
<th>T7</th>
<th>T7</th>
<th>T7</th>
<th>T7</th>
<th>T7</th>
</tr>
</thead>
<tbody>
<tr>
<td>T2</td>
<td>T2</td>
<td>T2</td>
<td>T2</td>
<td>T2</td>
<td>T4</td>
<td>T4</td>
<td>T4</td>
<td>T4</td>
<td>T4</td>
<td>T4</td>
<td>T4</td>
<td>T4</td>
<td>T4</td>
<td>T4</td>
</tr>
</tbody>
</table>

Greedy using a reordering of the list so that longest running tasks appear earliest in the list:

<table>
<thead>
<tr>
<th>T7</th>
<th>T7</th>
<th>T7</th>
<th>T7</th>
<th>T7</th>
<th>T7</th>
<th>T1</th>
<th>T1</th>
<th>T1</th>
<th>T1</th>
<th>T6</th>
<th>T6</th>
<th>T6</th>
</tr>
</thead>
<tbody>
<tr>
<td>T4</td>
<td>T4</td>
<td>T4</td>
<td>T4</td>
<td>T4</td>
<td>T4</td>
<td>T2</td>
<td>T2</td>
<td>T2</td>
<td>T2</td>
<td>T3</td>
<td>T3</td>
<td>T5</td>
</tr>
</tbody>
</table>
10. We described the proof that 3SAT is polynomial reducible to Subset-Sum. You must repeat that.

a.) Assuming a 3SAT expression \((a + a + \overline{b}) (\overline{a} + b + c)\), fill in all omitted values (zeroes elements can be left as omitted) of the reduction from 3SAT to Subset-Sum.

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>b</th>
<th>c</th>
<th>a + a + \overline{b}</th>
<th>\overline{a} + b + c</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>~a</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>b</td>
<td></td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>~b</td>
<td></td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>c</td>
<td></td>
<td></td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>~c</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C1</td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>C1’</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>C2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C2’</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

b.) List some subset of the numbers above (each associated with a row) that sums to 1 1 1 3 3. Indicate what the related truth values are for a, b and c.

\(a = T \); \(b = T \); \(c = T \)

1 0 0 1 0 or 1 0 0 2 0
0 1 0 0 1 0 1 0 0 1
0 0 1 0 1 0 0 1 0 1
0 0 0 1 0 0 0 1 0
0 0 0 1 0 0 0 0 1
0 0 0 0 1

11. Present a gadget used in the reduction of 3-SAT to some graph theoretic problem where the gadget guarantees that each variable is assigned either True or False, but not both. Of course, you must tell me what graph theoretic problem is being shown NP-Complete and you must explain why the gadget works.

**Vertex Cover**
- Must Cover each Edge
- Set goal to min vertices
- Must choose one but not both are needed
- This translates to choosing a or \(\overline{a}\)

**3-Color**
- Cannot choose B for either a or \(\overline{a}\)
- So one must be T and other F

![Diagram](image)
12. Let \(Q\) be some problem (an optimization or decision problem). Assuming \(\leq p\) means many-one reducible in polynomial time and \(\leq tp\) means Turing-reducible in polynomial time, categorize \(Q\) as being in one of \(P\), \(NP\), \(co-NP\), \(NP-Complete\), \(NP-Easy\), \(NP-Hard\), or \(NP-Equivalent\) (see first two pages for definitions of each of these concepts). For each case, choose the most precise category. I filled in one answer already.

<table>
<thead>
<tr>
<th>Description of (Q)</th>
<th>Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Q) is decidable in deterministic polynomial time</td>
<td>(P)</td>
</tr>
<tr>
<td>For some (R) in (NP), (Q \leq tp R)</td>
<td>(NP-Easy)</td>
</tr>
<tr>
<td>(Q) is both (NP-Easy) and (NP-Hard)</td>
<td>(NP-Equivalent)</td>
</tr>
<tr>
<td>(Q) is in (NP) and if (R) is in (NP) then (R \leq p Q)</td>
<td>(NP-Complete)</td>
</tr>
<tr>
<td>A solution to (Q) is verifiable in deterministic polynomial time</td>
<td>(NP)</td>
</tr>
<tr>
<td>(Q)’s complement is in (NP)</td>
<td>(Co-NP)</td>
</tr>
</tbody>
</table>

13. A graph \(G\) is \(k\)-Colorable if its vertices can be colored using just \(k\) (or fewer colors) such that adjacent vertices have different colors. The Chromatic Number of a graph \(G\) is the smallest number \(k\) for which \(G\) is \(k\)-Colorable, \(k\)-Colorable is a decision problem that has parameters \((G, k)\), whereas the Chromatic Number problem is a function with a single parameter \(G\). In all cases, assume \(G\) has \(n\) vertices.

a.) Show that \(k\)-Colorable \(\leq tp\) Chromatic Number \((\leq tp\) means Turing reducible in polynomial time).

\(G\) is \(k\)-Colorable iff its Chromatic Number is some \(j \leq k\)

This can be checked by just one invocation of the Oracle for Chromatic Number

b.) Show that Chromatic Number \(\leq tp\) \(k\)-Colorable \((\leq tp\) means Turing reducible in polynomial time).

\(G\)’s Chromatic Number is no worse than \(n\), the number of vertices. Doing a binary search, we can make at most \(\log_2 n\) calls to the oracle for \(k\)-Colorable to determine the smallest number for which \(G\) is \(k\)-colorable

14. Partition refers to the decision problem as to whether some set of positive integers \(S\) can be partitioned into two disjoint subsets whose elements have equal sums. Subset-Sum refers to the decision problem as to whether there is a subset of some set of positive integers \(S\) that precisely sums to some goal number \(G\).

a.) Show that Partition \(\leq p\) Subset-Sum.

\(Look\ at\ notes\)

b.) Show that Subset-Sum \(\leq p\) Partition.

\(Look\ at\ notes\)
15. **QSAT** is the decision problem to determine if an arbitrary fully quantified Boolean expression is true. Note: **SAT** only uses existential, whereas **QSAT** can have universal qualifiers as well so it includes checking for Tautologies as well as testing Satisfiability. What can you say about the complexity of **QSAT** (is it in **P**, **NP**, **NP-Complete**, **NP-Hard**)? Justify your conclusion.

**QSAT** is **NP-Hard**. This is so since **SAT** trivially reduces to **QSAT** (it is a subproblem of **QSAT**). Since **SAT** is known to be **NP-Complete** then some **NP-Complete** problem polynomially reduces to **QSAT**. This makes **QSAT** **NP-Hard**. As we cannot (at least not yet) show **QSAT** is in **NP**, then **NP-Hard** is the best we can do.

16. Given the following instance of 2SAT, \( E = (a \lor b) \land (\neg a \lor \neg b) \land (\neg a \lor \neg c) \land (a \lor c) \), display the associated implication graph, show its strongly connected components and then show how this leads to an assignment of variables that satisfies \( E \).

![Implication Graph](image)

\( a = T, b = F, c = F \) or \( a = F, b = T, c = T \)

17. Specify True (T) or False (F) for each statement.

<table>
<thead>
<tr>
<th>Statement</th>
<th>T or F</th>
</tr>
</thead>
<tbody>
<tr>
<td>Every Regular Language is also a Context Free Language</td>
<td>T</td>
</tr>
<tr>
<td>Phrase Structured Languages are the same as RE Languages</td>
<td>T</td>
</tr>
<tr>
<td>The Context Free Languages are closed under Complement</td>
<td>F</td>
</tr>
<tr>
<td>A language is recursive iff it and its complement are re</td>
<td>T</td>
</tr>
<tr>
<td>PCP is undecidable even for one letter systems</td>
<td>F</td>
</tr>
<tr>
<td>Membership in Context Sensitive Languages is undecidable</td>
<td>F</td>
</tr>
<tr>
<td>Every RE language is Turing reducible to its complement</td>
<td>T</td>
</tr>
<tr>
<td>Emptiness is undecidable for Context Sensitive Languages</td>
<td>T</td>
</tr>
<tr>
<td>The complement of a trace language is Context Free</td>
<td>T</td>
</tr>
<tr>
<td>The word problem for two-letter Semi-Thue Systems is decidable</td>
<td>F</td>
</tr>
</tbody>
</table>