Generally useful information.

- The notation \( z = <x, y> \) denotes the pairing function with inverses \( x = <z>_1 \) and \( y = <z>_2 \).
- The minimization notation \( \mu \ y \ P(\ldots y) \) means the least \( y \) (starting at 0) such that \( P(\ldots y) \) is true. The bounded minimization (acceptable in primitive recursive functions) notation \( \mu \ y \ (u \leq y \leq v) \ P(\ldots y) \) means the least \( y \) (starting at \( u \) and ending at \( v \)) such that \( P(\ldots y) \) is true. Unlike the text, I find it convenient to define \( \mu \ y \ (u \leq y \leq v) \ P(\ldots y) \) to be \( v+1 \), when no \( y \) satisfies this bounded minimization.
- The tilde symbol, \( \sim \), means the complement. Thus, set \( \sim S \) is the set complement of set \( S \), and predicate \( \sim P(x) \) is the logical complement of predicate \( P(x) \).
- A function \( P \) is a predicate if it is a logical function that returns either 1 (true) or 0 (false). Thus, \( P(x) \) means \( P \) evaluates to true on \( x \), but we can also take advantage of the fact that true is 1 and false is 0 in formulas like \( y \times P(x) \), which would evaluate to either \( y \) (if \( P(x) \)) or 0 (if \( \sim P(x) \))
- A set \( S \) is recursive if \( S \) has a total recursive characteristic function \( \chi_S \), such that \( x \in S \iff \chi_S(x) \). Note \( \chi_S \) is a predicate. Thus, it evaluates to 0 (false), if \( x \notin S \).
- If I say a function \( g \) is partially computable, then there is an index \( g \) (I know that’s overloading, but that’s okay as long as we understand each other), such that \( \Phi_g(x) = \Phi(x, g) = g(x) \). Here \( \Phi \) is a universal partially recursive function.
- The notation \( f(x)_\downarrow \) means that \( f \) converges when computing with input \( x \), but we don’t care about the value produced. In effect, this just means that \( x \) is in the domain of \( f \).
- The notation \( f(x)_\uparrow \) means \( f \) diverges when computing with input \( x \). In effect, this just means that \( x \) is not in the domain of \( f \).
- The Halting Problem for any effective computational system is the problem to determine of an arbitrary effective procedure \( f \) and input \( x \), whether or not \( f(x)_\downarrow \). The set of all such pairs, \( K_0 \), is a classic re non-recursive one.
- The Uniform Halting Problem is the problem to determine of an arbitrary effective procedure \( f \), whether or not \( f \) is an algorithm (halts on all input). The set of all such function indices is a classic non re one.
- \( A \leq_m B \) (A many-one reduces to B) means that there exists a total recursive function \( f \) such that \( x \in A \iff f(x) \in B \). If \( A \leq_m B \) and \( B \leq_m A \) then we say that \( A =_m B \) (A is many-one equivalent to B). If the reducing function is 1-1, then we say \( A \leq_1 B \) (A one-one reduces to B) and \( A =_1 B \) (A is one-one equivalent to B).
1. Choosing from among (REC) recursive, (RE) re non-recursive, (coRE) co-re non-recursive, (NR) 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.) \{ f | \text{domain}(f) \text{ is finite} \} 

b.) \{ f | \text{domain}(f) \text{ is empty} \} 

c.) \{ <f,x> | f(x) \text{ converges in at most 20 steps} \} 

d.) \{ f | \text{domain}(f) \text{ converges in at most 20 steps for some input x} \} 

2. Let set A be recursive, B be re non-recursive and C be non-re. Choosing from among (REC) recursive, (RE) re non-recursive, (NR) non-re, categorize the set D in each of a) through d) by listing all possible categories. No justification is required.
   
a.) D = \sim C 

b.) D \subseteq A \cup C 

c.) D = \sim B 

d.) D = B - A 

3. Prove that the Halting Problem (the set HALT = K_0 = L_0) is not recursive (decidable) within any formal model of computation. (Hint: A diagonalization proof is required here.)
4. Using reduction from the known undecidable \textbf{HasZero}, $HZ = \{ f \mid \exists x \ f(x) = 0 \}$, show the non-recursiveness (undecidability) of the problem to decide if an arbitrary primitive recursive function $g$ has the property \textbf{IsZero}, $Z = \{ f \mid \forall x \ f(x) = 0 \}$. Hint: there is a very simple construction that uses \textbf{STP} to do this. \textit{Just giving that construction is not sufficient; you must also explain why it satisfies the desired properties of the reduction.}
5. Define \( \text{RANGE\_ALL} = \{ f \mid \text{range}(f) = \beth \} \).

a.) Show some minimal quantification of some known recursive predicate that provides an upper bound for the complexity of this set. (Hint: Look at c.) and d.) to get a clue as to what this must be.)

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

c.) Show that \( \text{TOTAL} \leq_m \text{RANGE\_ALL} \), where \( \text{TOTAL} = \{ f \mid \forall y \varphi(f)(y) \downarrow \} \).

d.) Show that \( \text{RANGE\_ALL} \leq_m \text{TOTAL} \).

e.) From a.) through d.) what can you conclude about the complexity of \( \text{RANGE\_ALL} \)?
6. This is a simple question concerning Rice’s Theorem.
   a.) State the strong form of Rice’s Theorem. Cover all conditions for it to apply; don’t skimp on details.

b.) Describe a set of partial recursive functions whose membership cannot be shown undecidable through Rice’s Theorem. What condition is violated by your example?

7. Using the definition that $S$ is recursively enumerable iff $S$ is either empty or the range of some algorithm $f_S$ (total recursive function), prove that if both $S$ and its complement $\sim S$ are recursively enumerable then $S$ is decidable. To get full credit, you must show the characteristic function for $S$, $\chi_S$, in all cases. Be careful to handle the (two) extreme cases. Hint: This is not an empty suggestion.