COT 6410 Fall 2010 Exam#2 Name: KEY

12 **1**. Choosing from among **(REC) recursive**, **(RE) re non-recursive, (coRE) co-re non-recursive**, **(NRNC) 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 | f(f) ↓}

$\exists$ t STP(f,f,t) — ***RE***

b.) B = { f | range(f) is a proper subset of ℵ }

$\exists$x $\forall$<y,t> [STP(f,x,t)$\Rightarrow$ Value(f,y,t) $\neq$ x] — ***NRNC***

c.) C = { f | f(0) take at least 100 steps to converge, if at all }

~STP(f,0,99) — ***REC***

d.) D = { f | f diverges everywhere }

$\forall$<x,t> ~STP(f,x,t) — ***coRE***

6 **2**. Define, compare and contrast the notions of **Countable** and **Recursively Enumerable** for some set **S**. Question 7 actually gives one of these definitions.

A set S is countable iff it can be placed in a 1-1 correspondence with a subset of the Natural numbers. That is, S is countable iff there is an injective mapping (not necessarily computable) that associates each element of S with a unique element of ℵ. Alternatively, S is countable iff S is empty or there is a surjective mapping (not necessarily computable) that associates each element of ℵ with a unique element of S.

A set S is recursively enumerable (re) iff it is either empty or there exists a total computable function that effectively maps the Natural numbers onto the set. That is, S is re, non-empty iff there is an algorithm (a total computable function), f, whose domain is ℵ and whose range is S.

Every re set is countable, but not every countable set is re. The issue is that re requires a mapping that is computable, whereas countable just requires the existence of a mapping; whether or not that mapping is computable is not relevant. Thus, all subsets of ℵ are countable, but only a countable number are re since there can only be a countable number of algorithms, whereas there are an uncountable number of subsets of ℵ.

3 **3**. In your first exam you were asked to use diagonalization to show that the set **TOTAL** is non-re. Why does this kind of proof fail to show the effective procedures are non-re? You do not have to present the whole diagonalization proof, you are just to describe the key contradiction and point out why this fails to be a contradiction for effective procedures.

The essential contradiction is that we were able to take the enumerating algorithm for the set of algorithms and use it to create an algorithm that contradicted its own existence by computing two different values for the same input. Specifically, the algorithm was called D and, if its index was d, it computed D(d) as D(d)+1. This is not necessarily a contradiction if D is an effective procedure, because D might diverge at d, in which case D(d) and D(d)+1 are both undefined and so equal.

20 **4**. Let set **A** be **infinite recursive**, **B** be **re non-recursive** and **C** be **non-re**. Using the terminology **(REC) recursive**, **(RE) re**, **(NR) non-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 $\aleph$ are infinite REC; **K** and $\mathbf{K_0}$ are RE; and **TOTAL** is non-re. You may also assume, for any set **S**, the existence of comparably hard sets $\mathbf{S_E = \{2x|x\in S\}}$ and $\mathbf{S_D = \{2x+1|x\in S\}}$.

a.) $\mathbf{A \oplus B = \{ x \mid x \in A}$ or $\mathbf{x \in B}$, but x does not belong to both A and B }

RE: ***YES. Let $A = \{2x \mid x \in \aleph\}$. Let $B = K_D = \{2f+1 \mid f(f)\downarrow\}$. A is infinite recursive. B is re non-recursive (actually re-complete). $A \oplus B = A \cup B$ since the sets are disjoint. Thus, $A \oplus B = \{2x \mid x \in \aleph\} \cup K_D$. This has the complexity of K since all even numbers are in and the odd are in iff they are in K_D. Thus, $A \oplus B$ is re, non-recursive.***

NR: ***YES. Let $A = \aleph$. Let $B = K$. $A \oplus B = \sim B = \sim K = \{f \mid f(f)\uparrow\}$ which is co-re and hence NR.***

b.) min(A,C) = { min(x,y) | x ∈ A and y ∈ C }

REC: ***YES. Let $A = \aleph$. Let C = TOTAL. Min(A,C) = $\aleph$ as every element of $\aleph$ is dominated by some element of C since C has no upper bound.***

c.) ~C = { x | x ∉ C }

REC: ***NO. If ~C were recursive then C is recursive, but C is not even RE (it could be co-RE or even NRNC).***

RE: ***YES. Let C =~K. This set is co-RE and so its complement is RE as desired.***

5. Define **SemiConstant (SC) = { f | |range(f)| = 1 }**.

3 **a.)** Show some minimal quantification of some known recursive predicate that provides an upper bound for the complexity of this set.

$\exists$<x, t> $\forall$<y, s> [STP(f,x,t) & (STP(f,y,s) $\Rightarrow$ Value(f,x,t) = Value(f,y,s))]

6 **b.)** Use Rice's Theorem to prove that **SC** is undecidable.

First, SC is non-trivial as the constant Zero is in the set and the successor function S is not.

Second, let f and g be arbitrary indices of arbitrary effective procedures, such that $\forall$x f(x)=g(x). Clearly, since the functions have the same I/O behavior, their ranges (and domains) are the same. Thus, f is in SC iff |range(f)| = 1 iff |range(g)| = 1 iff g is in SC. Thus, SC is an I/O property.

This means SC satisfies both properties of Rice's Theorem and is therefore undecidable.

6 **c.)** Show that $\mathbf{K \le_m SC}$, where $\mathbf{K = \{ f \mid \varphi_f(f)\downarrow \}}$.

Let f be arbitrary. Define an algorithmic mapping G from indices to indices as $G_f(x) = f(f)$. Now, the range of $G_f = \{f(f)\}$. If f is in K, then this range is a singleton value and so G_f is in SC. If f is not in K, then this range is empty and so G_f is not in SC. Thus, $K \le_m SC$.

3 **6.** 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.
NOT_SIMPLE_QUADRATIVE (NSQ) = { f | ∃y $\varphi_f(y)$ fails to converge in $(y+1)^2$ steps }.

Rice's does not apply because NSQ is not an I/O property. Consider ZERO(x) = 0 is not in NSQ. However, KZERO(x) = $\mu y\ (y > (x+1)^2)$ - $\mu y\ (y > (x+1)^2) = 0$ is in NSQ as it takes $2(x+1)^2$ steps for all x.

8 **7.** Using the definition that **S** is recursively enumerable iff **S** is either empty or the range of some algorithm $\mathbf{f_S}$ (total recursive function), prove that if both **S** and its complement **~S** are recursively enumerable then **S** is decidable. To get full credit, you must show the characteristic function for **S**, χ_S, in all cases. Be careful to handle the extreme cases (there are two of them). Hint: This is not an empty suggestion. Also, be sure to discuss why your χ_S works.

Let S = ∅ then ~S = ℵ and χ_S(x) = 0 for all x.

Let S = ℵ then ~S = ∅ and χ_S(x) = 1 for all x.

Assume S ≠∅ and ~S ≠∅ then each has an enumerating algorithm. Call these f_S and $f_{\sim S}$.

Define $\chi_S(x) = f_S(\mu y[f_S(y) = x \,||\, f_{\sim S}(y) = x] = x$

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

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

Thus, χ_S(x) meets our requirements.