COT 5310 Fall 2006 Midterm#1 Sample Name:_____

Generally useful information.

- The notation $\mathbf{z = \langle x,y \rangle}$ denotes the pairing function with inverses $\mathbf{x = \langle z \rangle_1}$ and $\mathbf{y = \langle z \rangle_2}$.
- The minimization notation μ **y [P(…,y)]** means the least **y** (starting at **0**) such that **P(…,y)** is true. The bounded minimization (acceptable in primitive recursive functions) notation μ **y (u≤y≤v) [P(…,y)]** means the least **y** (starting at **u** and ending at **v**) such that **P(…,y)** is true. Unlike the text, I find it convenient to define μ **y (u≤y≤v) [P(…,y)]** to be **v+1**, when no **y** satisfies this bounded minimization.
- The tilde symbol, **~,** means the complement. Thus, set **~S** is the set complement of set **S**, and predicate **~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 × P(x)**, which would evaluate to either **y** (if **P(x)**) or **0** (if **~P(x)**).
- A set **S** is recursive if **S** has a total recursive characteristic function χ_S, such that $\mathbf{x \in S \Leftrightarrow \chi_S(x)}$. Note χ_S is a predicate. Thus, it evaluates to **0** (**false**), if $\mathbf{x \notin S}$.
- When I say a set **S** is re, unless I explicitly say otherwise, you may assume any of the following equivalent characterizations:
  1. **S** is either empty or the range of a total recursive function $\mathbf{f_S}$.
  2. **S** is the domain of a partial recursive function $\mathbf{g_S}$.
- 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 $\mathbf{\Phi_g(x) = \Phi(x, g) = g(x)}$. Here Φ is a universal partially recursive function.
  Moreover, there is a primitive recursive function **STP**, such that
  STP(x, g, t) is **1** (true), just in case **g**, started on **x**, halts in **t** or fewer steps.
  STP(x, g, t) is **0** (false), otherwise.
  Finally, there is another primitive recursive function **VALUE**, such that
  VALUE(x, g, t) is **g(x)**, whenever **STP(x, g, t)**.
  VALUE(x, g, t) is defined but meaningless if **~STP(x, g, t)**.
- 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 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.

16 **1**. Choosing from among **(REC) recursive**, **(RE) re non-recursive**, **(NR) non-re**, categorize each of the sets in a) through d). Justify your answer by showing some minimal quantification of some known recursive predicate or by another clear and convincing short argument.

a.) **{ <f,x>| ∀(t>x) STP(x,f,t) }** _____

Justification:

_____

b.) **{ f** | **∀x f(x)** is a prime number **}** _____

Justification:

_____

c.) **{ f | ∃x f(x)↓ }** _____

Justification:

_____

d.) **{ x | ∀f f(x)↑ }** _____

Justification:

_____

15 **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 e) by listing **all** possible categories. No justification is required.

a.) **D = ~A ∪ B** _____

b.) **D = A ∩ ~C** _____

c.) **D = B – A** _____

d.) **D = C ∪ ~C** _____

e.) **A ⊆ D ⊆ B** _____

7 **3.** Using the definition that **S** is semi-decidable iff **S** is the domain of some procedure $\mathbf{g_S}$ (partial recursive function), prove that if both **S** and its complement **~S** are semi-decidable then **S** is decidable. Note: To get full credit, you must show the characteristic function for **S**, χ_S, and argue convincingly (not formally) that the function you specified is the correct one for **S**. Hint: The **STP** predicate is useful here. You may also use any other known total recursive functions to attack this.

10 **4.** Prove that the **Uniform Halting Problem** (the set **TOTAL**) is not recursively enumerable within any formal model of computation. You may assume the existence of a universal machine. (Hint: A diagonalization proof is required and the most direct approach is via recursive functions.)

6 **5.** Using reduction from the known undecidable **Non-Empty Problem, NE = { f | ∃x f(x)↓ }**, show the non-recursiveness (undecidability) of the problem to decide if an arbitrary effective procedure **g** has a fixed point; that is, whether or not **g(x) = x**, for some **x**. Hint: there is a very simple construction to do this. Just giving that construction is not sufficient; you must also explain why it satisfies the desired properties of the reduction.

9 **6.** Let **S** be an re (recursively enumerable) non-recursive set, and let **T** be an infinite recursive (decidable) set. Define
E = { z | z = x - y, where **x ∈ S** and **y ∈ T** and **x – y = 0** if **x<y** }.
You may assume that **S** is the range of a total recursive function $\mathbf{f_S}$, and domain of a partial recursive function $\mathbf{g_S}$. You may also assume **T** has a total characteristic function χ_T.

Can **E** be re non-recursive? (Note: if your answer is yes, you must show a non-recursive re set **S**, an infinite decidable set **T** and a resulting known re non-recursive set **E**; if your answer is no, you must prove this by either showing that **E** is always non-re or by demonstrating how to construct a characteristic function based on any arbitrary non-recursive re set **S** and infinite decidable set **T**.)
Hint: Consider making **S = { 2*x | x ∈ HALT }**.

The following is the basis for questions 7 through 9. Show that the predicate **Domination** is computable, using three different models of computation, recursive functions, register machines and factor replacement systems, where **Domination (x , y, z) = 1** if **(x+y) > z; 0** otherwise.

6 **7.** Assuming only the recursiveness of the base functions
$\mathbf{C_a(x_1,\ldots,x_n) = a}$: constants; $\mathbf{I_i^n\ (x_1,\ldots,x_n) = x_i}$: identity (projection); **S(x) = x+1** : increment
and the standard arithmetic operators +, -, * and //
and that the recursive functions are closed under composition, primitive recursion (iteration) and minimization, show the recursiveness of **Domination**
You may not take advantage of any additional closure properties with which you are familiar. Thus, you will need to show the recursiveness of any other operators you require in your demonstration.

6 **8**. Demonstrate a **Register Machine** where the numbers **x**, **y** and **z are** in registers **R2**, **R3** and **R4**, respectively, all other registers being zero. **Domination (x , y, z) = (1** if **(x+y) > z; 0)** otherwise ends up in register **R1**. The final contents of all registers except **R1** are unimportant.

6 **9**. Present a **Factor Replacement System** (ordered rules of form $\mathbf{ax \rightarrow bx}$) that, when started on the number $\mathbf{3^x\ 5^y\ 7^z}$, terminates on the number $\mathbf{2=2^1}$, if **x + y > z**, and on $\mathbf{1=2^0}$, otherwise (**Domination** encoded). When you end, no prime factor other than perhaps **2** appears in the final number.