

Generally useful information.

- The notation $z = \langle x, y \rangle$ denotes the pairing function with inverses $x = \langle z \rangle_1$ and $y = \langle z \rangle_2$.
- The minimization notation $\mu y [P(\dots, y)]$ means the least y (starting at 0) such that $P(\dots, y)$ is true. The bounded minimization (acceptable in primitive recursive functions) notation $\mu y (u \leq y \leq v) [P(\dots, y)]$ means the least y (starting at u and ending at v) such that $P(\dots, y)$ is true. Unlike the text, I find it convenient to define $\mu y (u \leq y \leq v) [P(\dots, y)]$ 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 $P(x)$.
- The minus symbol, $-$, when applied to sets is set difference, so $S - T = \{x \mid x \in S \ \&\& \ x \notin T\}$.
- The absolute value, $|z|$, is the magnitude of z . Thus, $|x - y|$ is the difference between x and y , when x and y are both non-negative.
- 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 χ_S , such that $x \in S \Leftrightarrow \chi_S(x)$. Note χ_S is a predicate. Thus, it evaluates to **0 (false)**, if $x \notin S$.
- 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 f_S .
 2. S is the domain of a partial recursive function g_S .
 3. S is recognizable by a Turing Machine.
- 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 $\Phi_g(x) = \Phi(g, x) = g(x)$. Here Φ is a universal partially recursive function.
Moreover, there is a total recursive function STP , such that $STP(g, x, t)$ is **1 (true)**, just in case g , started on x , halts in t or fewer steps.
 $STP(g, x, t)$ is **0 (false)**, otherwise.
Finally, there is another total recursive function $VALUE$, such that $VALUE(g, x, t)$ is $g(x)$, whenever $STP(g, x, t)$.
 $VALUE(g, x, t)$ is defined but meaningless if $\sim STP(g, x, t)$.
- 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 set of all such $\langle f, x \rangle$ is denoted K_0 . A related set K is the set of all f that halt on their own indices. Thus, $K = \{f \mid \Phi_f(f) \downarrow\}$ and $K_0 = \{\langle f, x \rangle \mid \Phi_f(x) \downarrow\}$.
- 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 and is often called **TOTAL**.

- 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.
 - $D = \sim C$ _____
 - $D \subseteq (A \cup C)$ _____
 - $D = \sim B$ _____
 - $D = B - A$ _____
- Prove that the **Halting Problem** (the set K_0) is not recursive (decidable) within any formal model of computation. (Hint: A diagonalization proof is required here.)
- Using reduction from the known undecidable **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 **IsZero**, $Z = \{ f \mid \forall x f(x) = 0 \}$.
- Choosing from among **(D) decidable**, **(U) undecidable**, **(?) unknown**, categorize each of the following decision problems. No proofs are required.

Problem / Language Class	Regular	Context Free
$L = \Sigma^*$?		
$L = \phi$?		
$x \in L^2$, for arbitrary x ?		

- Choosing from among **(Y) yes**, **(N) No**, **(?) unknown**, categorize each of the following closure properties. No proofs are required.

Problem / Language Class	Regular	Context Free
Closed under intersection?		
Closed under quotient?		
Closed under quotient with Regular languages?		
Closed under complement?		

6. Prove that any class of languages, C , closed under union, concatenation, intersection with regular languages, homomorphism and substitution (e.g., the Context-Free Languages) is closed under **MissingMiddle**, where, assuming L is over the alphabet Σ ,
MissingMiddle(L) = $\{ xz \mid \exists y \in \Sigma^* \text{ such that } xyz \in L \}$
 You must be very explicit, describing what is produced by each transformation you apply.
7. Use **PCP** to show the undecidability of the problem to determine if the intersection of two context free languages is non-empty. That is, show how to create two grammars G_A and G_B based on some instance $P = \langle \langle x_1, x_2, \dots, x_n \rangle, \langle y_1, y_2, \dots, y_n \rangle \rangle$ of **PCP**, such that $L(G_A) \cap L(G_B) \neq \emptyset$ iff P has a solution. Assume that P is over the alphabet Σ . You should discuss what languages your grammars produce and why this is relevant, but no formal proof is required.
8. Consider the set of indices **CONSTANT** = $\{ f \mid \exists K \forall y \mid \phi_f(y) = K \}$. Use Rice's Theorem to show that **CONSTANT** is not recursive. Hint: There are two properties that must be demonstrated.
9. Show that **CONSTANT** $\equiv_m$ **TOT**, where **TOT** = $\{ f \mid \forall y \phi_f(y) \downarrow \}$.
10. Why does Rice's Theorem have nothing to say about each of the following? Explain by showing some condition of Rice's Theorem that is not met by the stated property.
 a.) **AT-LEAST-LINEAR** = $\{ f \mid \forall y \phi_f(y) \text{ converges in no fewer than } y \text{ steps} \}$.
 b.) **HAS-IMPOSTER** = $\{ f \mid \exists g \mid g \neq f \text{ and } \forall y \mid \phi_g(y) = \phi_f(y) \mid \}$.
11. We described the proof that **3SAT** is polynomial reducible to **Subset-Sum**.

- a.) Describe **Subset-Sum**
 b.) Show that **Subset-Sum** is in **NP**
 c.) Assuming a **3SAT** expression $(a + \sim b + c)(b + b + \sim c)$, fill in the upper right part of the reduction from **3SAT** to **Subset-Sum**.

	a	b	c	$a + \sim b + c$	$b + b + \sim c$
a	1				
$\sim a$	1				
b		1			
$\sim b$		1			
c			1		
$\sim c$			1		
C1				1	
C1'				1	
C2					1
C2'					1
	1	1	1	3	3

12. Describe the gadgets used to reduce 3SAT to the Vertex Covering Problem

13. Show a first-fit schedule for the following task times on two processors

$\{T1/1, T2/7, T3/2, T4/4, T5/4, T6/2, T7/5, T8/2, T9/3, T10/4\}$

14. Use the Pumping Lemma for CFLs to show:

$\{ ww \mid w \text{ is over } \{a,b\} \}$ is not Context Free

15. Write a context-free grammar for the complement of the language $\{ ww \mid w \text{ is in } \{a,b\}^* \}$