**Computer Science 4602**

**Fall 2020**

**Quiz 4**

Answer all of the questions. Clearly indicate the best answer to each multiple-choice problem (marked [MC]), even if no answer is ideal.

Read each question carefully. Be sure that you understand what the question is asking before you answer the question. Some of the questions are designed to see whether you understand definitions well enough to apply them.

Check your answers with a critical eye.

1. [MC] Function *f* is a mapping reduction from *A* to *B* provided
	1. *f* is computable and for all *x*, *f* (*x*) ∈ *B* $\leftrightarrow $ *f* (*x*) ∈ *A*.
	2. *f* is computable and for all *x*, *x* ∈ *B* $\leftrightarrow $ *f* (*x*) ∈ *A*.
	3. *f* is computable and for all *x*, *x* ∈ *A* $\leftrightarrow $ *f* (*x*) ∈ *B*.
	4. *f* is computable and for all *x*, *f* (*x*) ∈ *A* $\leftrightarrow $ *f* (*x*) ∈ *B*.
2. [MC] We say that *A* ≤*m B* if
	1. *A* is computable but *B* is not computable.
	2. *B* is computable but *A* is not computable.
	3. there exists a mapping reduction from *B* to *A*.
	4. there exists a mapping reduction from *A* to *B.*
3. [MC] One way to prove that a set *A* is uncomputable is to show that
	1. *A* ≤*m* HLT where HLT is the halting problem.
	2. *A͞* ≤*m* HLT where HLT is the halting problem.
	3. HLT ≤*m A* where HLT is the halting problem
	4. *A* is nontrivial.
	5. *A* is infinite.
4. [MC] Suppose that the alphabet includes symbol 0. Define

 *A* = {*p* | Run(*p,* 0)↓}

 HLT = {(*p,x*) | Run(*p, x*)↓}

That is, *A* is the set of all programs that terminate when their input is 0, and HLT is the halting problem. Which of the following functions is a mapping reduction from *A* to HLT?

1. *f* (*p,* 0) = *p*
2. *f* (*p*) = (*p,* 0)
3. *f* (*p, x*) = *p*
4. *f* (*p*) = *p*
5. *f* (*p*) = 0
6. Suppose that *A* and *B* are computable languages where *B* is nontrivial. (That is, *B* ≠ {} and *B͞* ≠ {}.) Assume that program *pA* computes *A* and program *pB* computes *B*. Show that *A* ≤*m B* by giving a mapping reduction from *A* to *B*.

We need a computable function $f $such that, for every $x$, $x \in A \leftrightarrow f\left(x\right)\in B.$

Since B is not empty, we can select a member Y of B. Since B does not contain all strings, we can select a nonmember N of B. Since A is computable, $f$ can ask whether $x \in A$. Here is a program that computes f.

$f(x)$:

 if $x \in A$

 return Y

 else

 return N

It should be clear that, if $x \in A$ then $f\left(x\right)\in B$, since $f(x) = Y$ and $Y \in B.$ Also, if $x \notin A$ then $f\left(x\right)\notin B$, since $f(x) = N$ and $N \notin B$.

1. If *p* is a program, *L*(*p*) is defined to be the set {*x* | Run(*p, x*) $≅$ 1}. That is, *L*(*p*) is the set of strings on which *p* answers 1. Let *D* = {*p* | *L*(*p*) is a finite set}.
	1. Read the definition of *D* carefully. Is *D* a finite set?

D is a set of programs. For every computable language L, there are infinitely many programs p such that L(p) = L. So D is an infinite set.

* 1. Is *D* computable? Justify your answer.

1. **D is nontrivial.**  There exists a program p where L(p) = {}, and that program is in D since {} is finite. There exists a program q where L(q) is the set of all strings (over Σ), and that program q is not in D since $Σ^{\*}$ is infinite.

2. **D respects equivalence.** Suppose p and q are two equivalent programs. Then L(p) = L(q). So $p \in D \leftrightarrow L\left(p\right)$ is finite $\leftrightarrow L\left(q\right)$ is finite $q \in D$.

Since D is nontrivial and D respects equivalence, D is uncomputable.

1. A program can produce a number as its result by producing a stringsuch as “400”. For this exercise, let’s restrict attention to programs that take an integer and produce an integer.

Suppose *A* = {*p* | Run(*p,* 2) = 0} and *B* = {*p* | Run(*p,* 2) = 1}.

Give a mapping reduction from *A* to *B*. Write down the property the reduction needs to have, specific to this reduction, before you define the reduction function that has that property.

Since $A$ and $B$ are both sets of programs, a mapping reduction from $A$ to $B$ must take something that has the right type for membership in $A$ (a program) and it must produce something of the right type for membership in $B$ (also a program).

Define

 $r\_{p}= $“{*r*(*x*): return Run(*p*, *x*) + 1}”

That is, $r\_{p}(x)$ returns a number that is one larger than the number returned by $p(x)$.

Define

 $ f\left(p\right)= r\_{p}.$

Notice that

$ r\_{p}\in B \leftrightarrow $ Run($r\_{p}$, 2) = 1 by the definition of $B$

$\leftrightarrow $ Run($p$,2) = 0 by the definition of $r\_{p}$

$\leftrightarrow p \in A$ by the definition of $A$

There were many mistakes on this question. Almost all of them were the result of not realizing that $A$ and $B$ are sets of programs. That means, if you say that $x \in A$, $x$ needs to be a program. Realizing that $A$ and $B$ are sets of programs involves nothing more than reading the definitions of A and B. ***Pay careful attention to definitions.***