CS 361 Meeting 1 — 2/7/20

Course Introduction

1. What's in a name:

   - This course is officially called "Theory of Computation" but that really doesn't tell you what the course is about.
   - A better name for the course, might be "Limits of Computation" because what we are really going to be looking at this semester are the fundamental limits on what we can accomplish with a computer.
   - If you have looked through the course text, however, you might not have found much that reminded you of computers as you know them. This is because "real" computers are overwhelmingly complicated devices that make reasoning about their fundamental abilities complicated.
   - Instead, we will discuss very simple, imaginary computing devices that are more tractable mathematically, but (we will claim) fundamentally as powerful as real computers.
   - Thus, a fair name for this course might be "Models and Limits of Computing".

2. The Turing Machine

   - To get a sense of what these models look like and how we might ultimately claim they capture the abilities of real computers, it helps to look at the contents of Figure 1 which contains a excerpt from the motivation for the *Turing Machine*, offered by its designer, Alan Turing.
   - Perhaps the most important thing to note about this quote is the publication date of the article from which it is taken — 1936. This was almost a decade before anything we would call a working computer was actually constructed.
   - Reading certain sentences of the article reinforces this. For example, Turing says "The behaviour of the computer at any moment

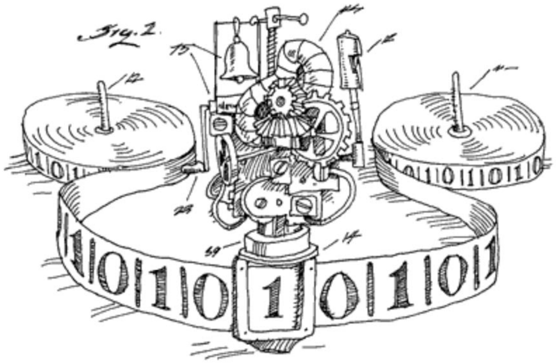
"Computing is normally done by writing certain symbols on paper. We may suppose this paper is divided into squares like a child's arithmetic book. In elementary arithmetic the two-dimensional character of the paper is sometimes used. But such a use is always avoidable, and I think that it will be agreed that the two-dimensional character of paper is no essential of computation. I assume then that the computation is carried out on one-dimensional paper, i.e. on a tape divided into squares. I shall also suppose that the number of symbols which may be printed is finite. If we were to allow an infinity of symbols, then there would be symbols differing to an arbitrarily small extent. The effect of this restriction of the number of symbols is not very serious. It is always possible to use sequences of symbols in the place of single symbols. Thus an Arabic numeral such as 17 or 999999999999999 is normally treated as a single symbol. Similarly in any European language words are treated as single symbols (Chinese, however, attempts to have an enumerable infinity of symbols). The differences from our point of view between the single and compound symbols is that the compound symbols, if they are too lengthy, cannot be observed at one glance. This is in accordance with experience. We cannot tell at a glance whether 9999999999999999 and 999999999999999 are the same.
The behaviour of the computer at any moment is determined by the symbols which he is observing and his "state of mind" at that moment. We may suppose that there is a bound B to the number of symbols or squares which the computer can observe at one moment. If he wishes to observe more, he must use successive observations. We will also suppose that the number of states of mind which need be taken into account is finite. The reasons for this are of the same character as those which restrict the number of symbols. If we admitted an infinity of states of mind, some of them will be "arbitrarily close" and will be confused. Again, the restriction is not one which seriously affects computation, since the use of more complicated states of mind can be avoided by writing more symbols on the tape."

A.M. Turing, On Computable Numbers, with an Application to the Entscheidungsproblem, 1936

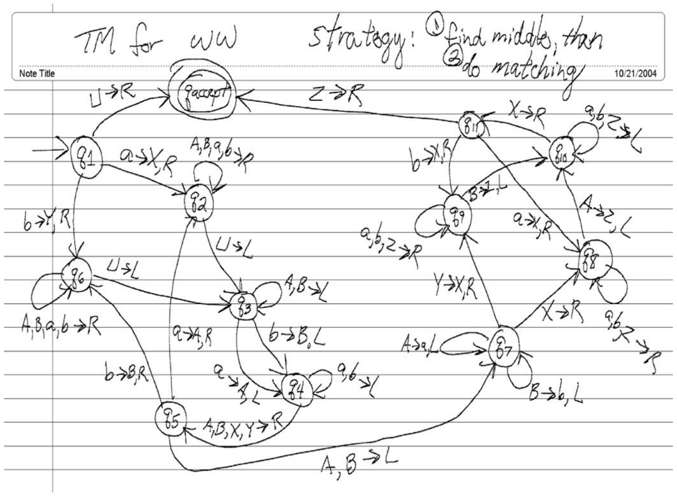
Figure 1: Quote from Turing

Click here to view the slides for this class is determined by the symbols which **he** is observing and **his** "state of mind" at that moment." He is not using the masculine pronoun in place of "it" here to refer to a machine. He is talking about the process of computing as performed by a human. His motivation is to extract the essential aspects of the process of computing whether performed by a device or a human.

- The model that Turing concocted given this motivation is captured (somewhat comically or at least creatively) by the following diagram:[1]



- Rather than being stored in RAM or SSD, the information processed by this machine is stored on a "tape" whose symbols can be read or written one symbol at a time by the control mechanism.
- The tape in the figure shows binary data, but all Turing required was a finite alphabet.
- The control mechanism (shown with bells and whistles in the diagram) is limited to having a finite number of states (that somehow are used to remember key properties of what it has seen on the tape).
- The "programs" written for this type of "machine" tend to look like the diagram shown below:



- The circles with names like "q2" in them represent the states of the control mechanism.
- The arrows represent the actions the machine performs when it sees a certain symbol on its tape when in a given state.
- Like any well written program, this one contains "informative" comments (i.e., the text written in the top margin of the page).
- We will draw lots of diagrams like this throughout the semester (though ours will hopefully be more readable).

3. Topics covered in the course
   - Models of Computation
     - Several alternatives:
       - Turing machines
       - Finite State Automata (finite state machines) - Read only tape.
       - Push-down Automata - Read only tape + writable stack.
     - For each model considered, we will determine what types of computations can and cannot be described using the model by sketching example "programs" and proving/understanding theorems about the model's power.

[1] This diagram came from the website http://www.worldofcomputing.net/theory/turing-machine.html

  - In some sense, FSAs and PDAs are strawmen designed to get you ready for Turing machines, but they also have some important practical consequences/applications (regular expressions and context-free parsing).
- Decidablity
  - The Halting problem and other examples of problems for which it is impossible to write a program that will reliably find a solution.
- Complexity
  - NP-completeness — Problems for which all known algorithmic solutions require so much time that they cannot be practically applied to problems of even modest size.

4. Goals of the course
   (a) Explore/understand limits of computation.
   (b) Develop skills in problem analysis and reduction
   (c) Develop skills in devising and presenting clear proofs

Grimy Details

1. Quick Attendance
2. Highlights of syllabus
   - Where to find it online: http://cs.williams.edu/˜tom/courses/361
     - As the semester progresses the online syllabus will include links to lecture slides, lecture notes and problem sets.
   - Office hours are a promise not a limitation
   - TAs - Christopher Anton, Will Burford, Spence Carrillo and Audrey Lee.
   - The text - Sipser 3rd edition.
   - Prerequisites (CS 256 or a 300-level Math course and permission)
     - You should at least be comfortable with the material covered in chapter 0 of the text!!
   - Exams - Take home midterm and final.
   - Homeworks - weekly, mostly due Fridays
   - Latex should be used to prepare all the work you submit!
   - Honor Code
     - You may discuss homework assignments with fellow students.
       - ∗ The work you submit should ultimately be your own. A good test of this is to ensure that you can write your own solution down without consulting notes from any group discussion.
       - ∗ You must list the names of other students you worked with on the work you submit.
       - ∗ You should limit the size of the groups you work in (to 2-3 students).
     - You should not search the Internet for homework solutions.

Countable sets and Computable Functions

1. Countable Numbers.
   - Consider the following “programming assignment”:

     > Write a program named PrintPI (in the language of your choice) that takes an integer n as its input and prints out the first n digits of the decimal expansion of π.

   - This should not seem like a hard program, particularly since there is an infinite sum that converges to π:

     $$\Sigma_{k=0}^{\infty}\frac{(-1)^k 3^{\frac{1}{2}-k}}{k+\frac{1}{2}}$$

     and some programming languages even support arbitrary precision arithmetic.
   - Similarly, you should be able to imagine writing such programs for other interesting real number like e, $\sqrt{2}$, etc.
   - Do you think it is possible to write such a program for any real number you can describe?

- The probably surprising answer is that there are real numbers for which it is impossible to write programs that can approximate them in this manner.
- Thus, the title of Turing's paper reflected the idea that there are numbers that are computable and numbers that are not computable.
- By the end of the semester, we will be able to describe specific examples of uncomputable numbers. Today, I just want to briefly give an argument that such numbers must exist.

2. Remember sets?

   - You better!!
     - Knowledge of sets is one of the "mathematical prerequisites" for this course that you should have picked up in Discrete Math or other math courses. This material can be found in Chapter 0 of the text. We will not cover this material explicitly in class. You should review Chapter 0 on your own to make sure you are comfortable with the material.
   - Even though we won't review most of Chapter 0, we will take a moment to review the basic facts about sets (since you haven't had a chance to do that yourselves yet).
   - A set is just a collection of things. We typically describe sets by listing their elements between curly braces or by placing a description of the elements between curly braces as in:
     - $\{2, 4, 6, 8\}$
     - $\{2n \mid 0 < n < 5\}$
     - $\{8, 4, 6, 2\}$
     - $\{2, 4, 6, 4, 2, 8\}$
   - All 4 examples shown above describe the same set. In particular, I included the last two examples to emphasize the facts that the order of the elements in a set is unimportant and that the number of times an element appears is irrelevant.
     - There is a construct called a *multiset* in which the number of times an element occurs matters, but almost no one ever uses them!
     - When order matters, we use sequences rather than sets (and replace the curly braces with parentheses. The following describe three distinct sequences:
       * $(2, 4, 6, 8)$
       * $(8, 4, 6, 2)$
       * $(2, 4, 6, 4, 2, 8)$
   - Most interesting sets and sequences are infinite!
     - $\{2, 4, 6, 8, \ldots\}$
     - $\{2n \mid 0 < n\}$
     - $(1, 1, 2, 3, 5, 8, 13, \ldots)$
     - $(1, 2, 4, 8, 16, \ldots)$
   - There are many operations one can perform on sets that you should be familiar with.
     - Operations that produce a truth value:

       membership The symbols $\in$ and $\notin$ are used to denote whether an element does or does not belong to a set.

       $$18 \in \{2, 4, 6, \ldots\} \quad \text{but} \quad 19 \notin \{2, 4, 6, \ldots\}$$

       subset/superset The symbol $\subset$ is used to indicate that one set is contained in another with the more specialized symbols $\subseteq$ and $\subsetneq$ used when it matters whether or not the sets are not equal to one another.

       $$\{2, 4, 6, \ldots\} \subset \{1, 2, 3, 4, \ldots\}$$

     - Operations that produce related sets (of similar type):

       union The union of two sets is the set containing all elements appearing in either set. Formally:

       $$A \cup B = \{x \mid x \in A \text{ or } x \in B\}$$

intersection The intersection of two sets is the set containing all elements appearing in both sets. Formally:

$$A \cap B = \{x \mid x \in A \text{ and } x \in B\}$$

complement The complement of a set is the set of all elements from the **universe** of elements under consideration that are not in the original set. Somewhat formally:

$$\bar{A} = \{x \mid x \notin A\}$$

- Operations that produce related sets (of different type):

power sets The power set of a set is the set of all subsets of the original set.

$$2^A = \{\pi \mid \pi \subset A\}$$

* $2^{\{x,y\}} = \{\{\}, \{x\}, \{y\}, \{x,y\}\}$, or equivalently
* $2^{\{x,y\}} = \{\emptyset, \{x\}, \{y\}, \{x,y\}\}$

products The Cartesian product of two sets is the set of all pairs (sequences of length two) with one element from the first set and one from the second.

$$A \times B = \{(a,b) \mid a \in A \text{ and } b \in B\}$$

* $\{x,y\} \times \{1,2,3\} = \{(x,1),(x,2),(x,3),(y,1),(y,2),(y,3)\}$

powers

$$A^k = \underbrace{A \times \ldots \times A}_{\text{k times}} = \{(a_1, \ldots, a_k) \mid a_i \in A\}$$

Note: When we take the cartesian product of n-sets, we expect a set of n-tuples rather than a set of pairs of elements from the first (or last) set in the product and the product of the remaining sets. That is,

$$\mathbb{R}^3 = \{(x,y,z) \mid x,y,z \in \mathbb{R}\}$$

but

$$\mathbb{R}^3 \neq \mathbb{R} \times (\mathbb{R} \times \mathbb{R}) = \{(x,(y,z)) \mid x,y,z \in \mathbb{R}\}$$

3. Countable sets

- I suspect that when you were children, many of you at some point got into an argument with a little friend about who had the most candies or whose house had the most Christmas lights, or...
- There is a song based on such childish behavior by a group named "The Wyrd Sisters" called "3000 million" that includes the lines:

> Standing in the playground, you typical young boys
> Arguing over who has the most toys
> One to the other with a gleam in his eye
> Well I've got so many you can't count that high
>
> I've got 3000 million twenty four hundred ten
> I've got 3000 million twenty four hundred ten
> And there are 14 more where that came from

- Of course, the kid who always wins such arguments is the one who knows enough to say "I have infinity toys..." (that is until someone else says "I have infinity plus 10 toys...").
- How many of you know the difference between countably and uncountably infinite sets?
  - For those who do, who first identified these mathematical notions and when? (Georg Cantor, 1875-1885 — 150 years after calculus)
  - For those who do not, which set is larger, the natural numbers, N, (i.e., the positive integers) or the even natural numbers?
- How can you compare the size of two sets if you cannot count the elements!

- Think about how a small child would compare collections of objects whose sizes were larger than how high the child could count. You match them up!
- Mathematically, you find a one-to-one function between the two sets. That is, you find a bijective (or one-to-one) function $f : X \to Y$, that is it is both:

  surjective $\forall y \in Y, \exists x \in X$ such that $f(x) = y$, and

  injective $f(x_1) = f(x_2) \iff x_1 = x_2$
- Cantor's "axiom" is that if there is a one-to-one mapping between two sets, then the two sets are of the same *cardinality* (i.e., the same size).
- For example, the function

  $$f(n) = 2n$$

  provides a one-to-one correspondence between the natural numbers and the even natural numbers. Based on this, we would claim that there are just as many even natural numbers as there are natural numbers!
- Any infinite set that is equivalent in size to the natural numbers in this sense is said to be *countably infinite*. If a set is infinite but it is not possible to match its elements up with the natural numbers in this way we say it is *uncountable*.

• This approach to comparing the sizes of infinite sets leads to some surprises.

- The fact that we don't consider the set of all natural numbers significantly larger than the set of even numbers should not surprise a computer scientist since the size of the sets differ by at most a constant factor. Their sizes are clearly the same big-O!
- Consider, however, the question of whether the set of positive rational numbers is larger than the set of natural numbers.
  - ∗ There are infinitely many rationals between every pair of integers (rather than just 1 extra member as in the case of even integers vs. integers).
  - ∗ In other words the rationals appear to be "infinity-squared" large.
- Surprisingly, if we follow Cantor's logic we can prove that the set of rationals is of the same size as the set of natural numbers.[2]
  - ∗ All we have to do is define a one-to-one mapping from the natural numbers to the rationals.
  - ∗ To do this, first imagine a table containing all positive fractions like this:

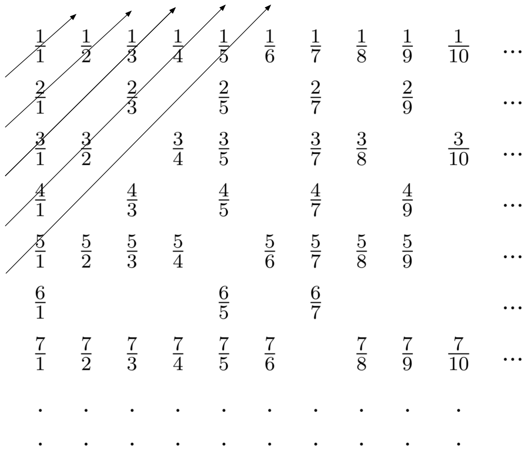
$$\begin{array}{ccccccccccc}
\frac{1}{1} & \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & \frac{1}{7} & \frac{1}{8} & \frac{1}{9} & \frac{1}{10} & \cdots \\
\frac{2}{1} & \frac{2}{2} & \frac{2}{3} & \frac{2}{4} & \frac{2}{5} & \frac{2}{6} & \frac{2}{7} & \frac{2}{8} & \frac{2}{9} & \frac{2}{10} & \cdots \\
\frac{3}{1} & \frac{3}{2} & \frac{3}{3} & \frac{3}{4} & \frac{3}{5} & \frac{3}{6} & \frac{3}{7} & \frac{3}{8} & \frac{3}{9} & \frac{3}{10} & \cdots \\
\frac{4}{1} & \frac{4}{2} & \frac{4}{3} & \frac{4}{4} & \frac{4}{5} & \frac{4}{6} & \frac{4}{7} & \frac{4}{8} & \frac{4}{9} & \frac{4}{10} & \cdots \\
\frac{5}{1} & \frac{5}{2} & \frac{5}{3} & \frac{5}{4} & \frac{5}{5} & \frac{5}{6} & \frac{5}{7} & \frac{5}{8} & \frac{5}{9} & \frac{5}{10} & \cdots \\
\frac{6}{1} & \frac{6}{2} & \frac{6}{3} & \frac{6}{4} & \frac{6}{5} & \frac{6}{6} & \frac{6}{7} & \frac{6}{8} & \frac{6}{9} & \frac{6}{10} & \cdots \\
\frac{7}{1} & \frac{7}{2} & \frac{7}{3} & \frac{7}{4} & \frac{7}{5} & \frac{7}{6} & \frac{7}{7} & \frac{7}{8} & \frac{7}{9} & \frac{7}{10} & \cdots \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots &
\end{array}$$

- Note that many rationals have several representations as fractions. For example, 3/2, 6/4, 9/6, etc. are all representations of the same number.
- Each rational is represented uniquely by one fraction in lowest terms (i.e., in which the numerator and denominator have no common factors). So, we can prune all fractions that are not in lowest terms from our table to obtain:

[2] Just to keep things simple, when I say rationals think positive rationals.

$\frac{1}{1}$	$\frac{1}{2}$	$\frac{1}{3}$	$\frac{1}{4}$	$\frac{1}{5}$	$\frac{1}{6}$	$\frac{1}{7}$	$\frac{1}{8}$	$\frac{1}{9}$	$\frac{1}{10}$	...
$\frac{2}{1}$		$\frac{2}{3}$		$\frac{2}{5}$		$\frac{2}{7}$		$\frac{2}{9}$		...
$\frac{3}{1}$	$\frac{3}{2}$		$\frac{3}{4}$	$\frac{3}{5}$		$\frac{3}{7}$	$\frac{3}{8}$		$\frac{3}{10}$	...
$\frac{4}{1}$		$\frac{4}{3}$		$\frac{4}{5}$		$\frac{4}{7}$		$\frac{4}{9}$		...
$\frac{5}{1}$	$\frac{5}{2}$	$\frac{5}{3}$	$\frac{5}{4}$		$\frac{5}{6}$	$\frac{5}{7}$	$\frac{5}{8}$	$\frac{5}{9}$		...
$\frac{6}{1}$				$\frac{6}{5}$		$\frac{6}{7}$				...
$\frac{7}{1}$	$\frac{7}{2}$	$\frac{7}{3}$	$\frac{7}{4}$	$\frac{7}{5}$	$\frac{7}{6}$		$\frac{7}{8}$	$\frac{7}{9}$	$\frac{7}{10}$	...
.	.	.	.	.	.	.	.	.	.	
.	.	.	.	.	.	.	.	.	.	

- We can now produce a numbered list that includes every rational number by pulling the fractions in lowest terms out of this table in a peculiar order. Namely, starting with the first element of each row in order from row 1 on down, collect all of the lowest term fractions found on the diagonal leading up and to the right from that entry. This pattern is illustrated by the table below. It produces the list $\frac{1}{1}$, $\frac{2}{1}$, $\frac{1}{2}$, $\frac{3}{1}$, $\frac{1}{3}$, $\frac{4}{1}$, $\frac{3}{2}$, $\frac{2}{3}$, $\frac{1}{4}$, $\frac{5}{1}$, etc.



$\frac{1}{1}$	$\frac{1}{2}$	$\frac{1}{3}$	$\frac{1}{4}$	$\frac{1}{5}$	$\frac{1}{6}$	$\frac{1}{7}$	$\frac{1}{8}$	$\frac{1}{9}$	$\frac{1}{10}$	...
$\frac{2}{1}$		$\frac{2}{3}$		$\frac{2}{5}$		$\frac{2}{7}$		$\frac{2}{9}$		...
$\frac{3}{1}$	$\frac{3}{2}$		$\frac{3}{4}$	$\frac{3}{5}$		$\frac{3}{7}$	$\frac{3}{8}$		$\frac{3}{10}$	...
$\frac{4}{1}$		$\frac{4}{3}$		$\frac{4}{5}$		$\frac{4}{7}$		$\frac{4}{9}$		...
$\frac{5}{1}$	$\frac{5}{2}$	$\frac{5}{3}$	$\frac{5}{4}$		$\frac{5}{6}$	$\frac{5}{7}$	$\frac{5}{8}$	$\frac{5}{9}$		...
$\frac{6}{1}$				$\frac{6}{5}$		$\frac{6}{7}$				...
$\frac{7}{1}$	$\frac{7}{2}$	$\frac{7}{3}$	$\frac{7}{4}$	$\frac{7}{5}$	$\frac{7}{6}$		$\frac{7}{8}$	$\frac{7}{9}$	$\frac{7}{10}$	...
.	.	.	.	.	.	.	.	.	.	
.	.	.	.	.	.	.	.	.	.	

- The function that maps the natural number n to the fraction at the nth position in this list is a 1-to-1 correspondence between the natural numbers and the rationals. Therefore, we can conclude that the set of all positive rationals are countably infinite.

- At this point, it might seem as if every infinite set is countable. In fact, there are many examples of sets that are uncountably infinite.
  - Cantor gave a proof that the real numbers, $\mathbb{R}$, are not countable.
  - The proof uses a technique known as diagonalization (for reasons that will become obvious) that will prove important later in the semester. It is also an example of proof by contradiction.
  - We start by assuming that the real numbers are countable. That is, that we can find some function $f : \mathbb{N} \to \mathbb{R}$ such that for every real number, r, there is some n such that $f(n) = r$.
  - This is basically equivalent to saying we can produced a numbered list containing every real number. $f(1)$ would be the first entry in this list, $f(2)$ the second entry, and so on.
  - Each $f(n)$ can be written as its (possibly infinite) decimal expansion. This will consist of some integer part followed by many digits after the decimal point. Suppose we let i^n represent the integer part of $f(n)$ and d_j^n represent the jth digit after the decimal point in the expansion of $f(n)$.
    That is, $f(n) = i^n.d_1^n d_2^n d_3^n d_4^n d_5^n...$
  - Consider the real number $r = 0.d_1^* d_2^* d_3^* d_4^* d_5^*...$ where each d_j^* is chosen so that it does not equal d_j^j. For example, we could let $d_j^* = (d_j^j + 5) \bmod 10$
  - The number r is "designed" to differ from each number in the purported list of all reals in at least one position. These positions fall along the diagonal of the table formed by listing all the numbers. But this means that r is distinct from every number in our assumed "list of all real numbers". That is, r is a real number that does not appear in our list of all real numbers.

  - This is a contradiction. So our assumption that such a list of real number exists must be false. In other words, the reals must be uncountable.

- This brings us to the punchline! What do countably infinite sets have to do with computing?
  - Remember the "programming assignment" mentioned above:

    > Write a program named PrintPI (in the language of your choice) that takes an integer n as its input and prints out the first n digits of the decimal expansion of π.

  - As I said earlier, you might imagine writing such a program for every real number, but there would be uncountably many such programs.
  - How many programs are there?
    - In a computer's memory or file system, each program is represented as a string of 0s and 1s.
    - This sequence of binary digits could be interpreted as a long binary number giving us a way to number every program uniquely.
    - This implies that the set of programs can be matched 1-to-1 with a subset of the natural numbers. Therefore, the set of programs must be countable.
  - As a result, for many real numbers it must not be possible to write programs that can print out their approximations!
- The counting argument we just discussed makes it clear that there are reasonable computations we might want to describe that in fact cannot be described. A major goal of the material in this course is to gain a understanding of the types of computations that are indeed impossible in this sense.