James Sweetman (jts2939)
Jung Yoon (jey283)
Unique #51835

Assignment 10

Q1.

In a monotone QSAT problem, no variable is negated and therefore it is not beneficial to set any of the odd x_i to false. In the CNF clauses, everything is OR'd so the worst case scenario that could make the instance unsatisfiable, would be to make all the even x_i false. Therefore the decision problem is a matter of checking whether all the odd x_i set to true will satisfy the instance when all the even x_i is false. This can be done in polynomial time.

Q2.

(Note: Adapted from same proof from Real Analysis class)

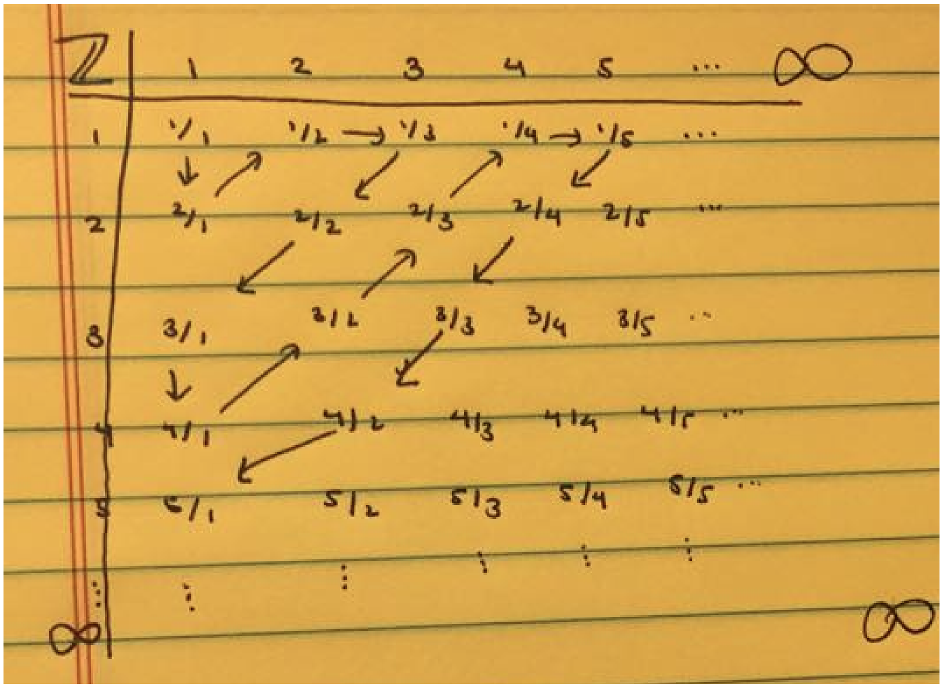
Proof:

First, integers are countably infinite (roughly proved this just in case):

> *By the definition of countable, there is a set of integers Z such that that is an injective function f from Z to the set of natural numbers N, where N = {0,1,2,3…}.*
>
> *In the case of the set of integers {...,-3,-2,-1,0,1,2,3,...}, the mapping would exist as so:*
>
> | *N* | *Z* |
> |---|---|
> | *0* | *0* |
> | *1* | *1* |
> | *2* | *-1* |
> | *3* | *2* |
> | *4* | *-3* |
> | … | … |
>
> *In order to count, we go from 0 to the first positive integer value to the first negative and so on moving from the center towards positive and negative infinity.*
>
> *Thus there exists an injection from ℕ to Z and thus, integers are countably infinite.*

Then, now that we know that integers are countably infinite, let's move onto the set of rationals. By the definition of rational, any rational in the set of all rational numbers can be represented in the form a/b where a and b are integers and b does not equal 0.

This means that we can map out the set of rationals as so:



In order to count, we move along the diagonals as drawn on the image above, going from the top left to the bottom right, which is infinite.

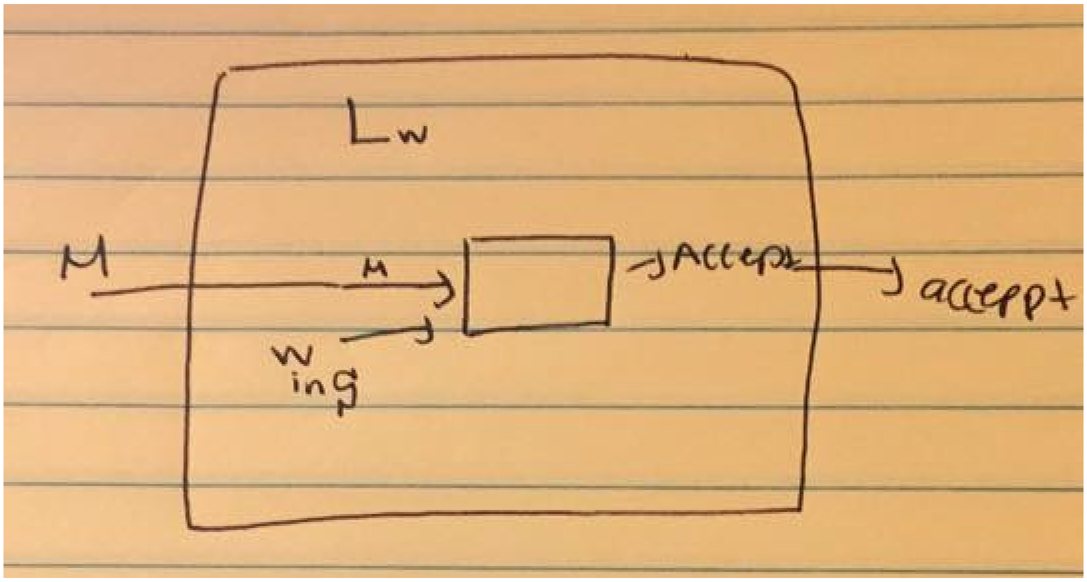
It is clear that every rational number will be represented in the given order above and that there is an injection from N to Q.

Thus, **<u>the set of all rational numbers is countably infinite</u>**, by the definition of countable.
End.

<u>Q3.</u>

$A = \{< M > | \forall w \in S M \text{ accepts } w\}$, where S is the set of all strings. This is to say that program A takes <M>, a string representation of a program, as input. Basically, M is a program that accepts all strings; M is a program and <M> is a string representation of a program M.

This can be modeled as so:



Let's say that M is a program that decides A. Then, by definition of Rice's Theorem (assuming we can use this) which states that a program is undecidable if

1. there exists a Turing machine that recognizes a language in *S*
2. there exists a Turing machine that recognizes a language not in *S,*

We know that A is undecidable. A is basically taking in any string, and thereby any language.

End.

<u>Q4.</u>

a.
{w1=1, w2 =5, w3 =4} and K = 5
The greedy algorithm will use three trucks, though the minimum possible number of trucks is two.

b.
Proof:
The minimum number of trucks needed is equal to W/K where W is the sum of all the weights of the containers. In order for an unnecessary second truck to be sent in the greedy algorithm, the sum of two consecutive boxes must be greater than K and less than 2K. It must be greater than K because:

Case 1: if it is K or less, the greedy algorithm will send only one truck.
Case 2: If it is more than 2K, then one of the boxes must be greater than K, which can't be held by a truck.

Therefore, at most only a second unnecessary truck will be sent for every consecutive pairing that adds up to greater than K. This upper bounds the total number of trucks used by the greedy algorithm to a factor of 2.
End.