1 (5 points) Prove that the square root of 5 is irrational.

\[ \sqrt{5} \]

Proof by Contradiction

Let's assume that \( \sqrt{5} \) is rational.

\[ \sqrt{5} = \frac{m}{n} \]

where \( m \) and \( n \) have no common divisors

- Square both sides

\[ 5 = \frac{m^2}{n^2} \]

- Get common denominator

\[ m^2 = 5n^2 \]

The right side is divisible by 5, so \( m^2 \) is divisible by 5, so \( m \) is divisible by 5

\[ m = 5p \]

for some integer \( p \)

(5p)^2 = 5n^2

25p^2 = 5n^2

\[ \frac{5}{5} \]

The left side is divisible by 5, so \( n^2 \) is divisible by 5, so \( n \) is divisible by 5

\[ m \text{ and } n \text{ have a common factor 5, therefore we get a contradiction! Our assumption is wrong} \]

Therefore \( \sqrt{5} \) is not rational.
2 (15 points)

2a. Use a general algorithm to design a non-deterministic finite automaton recognizing the language $(0 \cup 1^*)0$.

2b. Use the general algorithm to design a deterministic finite automaton recognizing this same language.
3 (30 points)

3a. Use a general algorithm to transform the following finite automaton (for checking whether a given binary number is odd) into the corresponding regular expression. This automaton recognizes unsigned binary integers. This automaton has three states:

- the starting state $s$,
- the state $q_0$ meaning that we have just read 0, and
- the state $q_1$ meaning that we have just read 1.

The state $q_1$ is the only final state. The transition is as follows:

- when we see 0, we move to $q_0$,
- when we see 1, we move to $q_1$.

3b. On the example of this automaton, explain, in detail, how the sequence 0101 will be presented as $xyz$ according to the pumping lemma. For this sequence, check -- by tracing step-by-step -- that the sequence $xy^iz$ for $i = 2$ is indeed accepted by the automaton.

3c. Use a general algorithm to describe a context-free grammar corresponding to this finite automaton.

3d. On the example of this grammar, explain, in detail, how the sequence 0101 will be presented as $uvxyz$ according to the pumping lemma for context-free grammars. For this sequence, check -- by tracing step-by-step -- that the sequence $uv^ixy^iz$ for $i = 2$ is indeed derived by this grammar.

3e. Use the general algorithm to transform this grammar into Chomsky normal form.

file:///Q:/cs3350.15/final.html
3c. Context Free Grammar

\[
\begin{align*}
S &\rightarrow 0Q_0 \\
S &\rightarrow 1Q_1 \\
Q_0 &\rightarrow 0Q_0 \\
Q_0 &\rightarrow 1Q_1 \\
Q_1 &\rightarrow 0Q_0 \\
Q_1 &\rightarrow 1Q_1 \\
Q_1 &\rightarrow \epsilon
\end{align*}
\]

3d. 0101

\[
\begin{align*}
u &= 0 \\
v &= 10 \\
x &= 1 \\
y &= 3 \\
z &= 3
\end{align*}
\]

For \( i = 2 \)

3e. Is it on the back of the movie, before
3e. CFG to PDA

CFG
S → 0Q₀
S → 1Q₁
Q₀ → 0Q₀
Q₀ → 1Q₁
Q₁ → 0Q₀
Q₁ → 1Q₁
Q₁ → ε

PDA

![PDA Diagram]
4 (10 points) Use the Pumping Lemma for context-free languages to prove that the language \( L \) consisting of all the words of the type \( x^n y^n z^n \), \( n = 0, 1, 2, \ldots \), is not context-free.

**Proof by contradiction**

Let's assume that \( L \) can be recognized by a Context-Free Grammar. This means there exists a \( p \) in all words of language \( L \) with \( |w| \geq p \) and there exists a \( u, v, x, y, z \) with \( w = uvxyz \) and the \( |vy| > 0 \) and \( |vxy| \leq p \) and for all \( i \) in \( uv^ixy^iz \) still pertains to the language.

Take \( w = x^p y^p z^p t^p \)

\[
\begin{array}{c}
\text{x} \quad \text{x} \quad \ldots \quad \text{x} \quad \text{y} \quad \ldots \quad \text{y} \quad \text{z} \quad \ldots \quad \text{z} \\
\hline
\text{x}^p \quad \text{y}^p \quad \text{z}^p \quad \text{t}^p
\end{array}
\]

- \( vxy \) can be in all \( x \), \( y \), \( z \), or \( t \) meaning that if we pump then we will have more of a symbol than the others.
- \( vxy \) can be in between \( x \) and \( y \) and if we pump then we will have more \( x \) or \( y \) than \( z \) and \( t \).
- \( vxy \) can be in between \( y \) and \( z \) and if we pump then we will have more \( y \) or \( z \) than \( x \) and \( t \).
- \( vxy \) can be in between \( z \) and \( t \) and if we pump then we will have more \( z \) or \( t \) than \( x \) and \( y \).

Any combination gives us a word that is not part of the language, therefore we get a contradiction. Therefore the language is not context-free.
5 (10 points)

5a. Describe a Turing machine that replaces all 0s with 1s and all 1s with 0s. Illustrate, step-by-step, how this Turing machine works, on the example of input 10, the result should be 01. For the first two intermediate states, show how the state of the Turing machine can be represented by a finite automaton with two stacks.

5b. Formulate Church-Turing thesis about Turing machines. Is it a mathematical theorem? Is it a statement about the physical world?

5a.  

<table>
<thead>
<tr>
<th>Start</th>
<th>Replace R</th>
</tr>
</thead>
<tbody>
<tr>
<td>Replace 0 → 1, R</td>
<td></td>
</tr>
<tr>
<td>Replace 1 → 0, R</td>
<td></td>
</tr>
<tr>
<td>Replace W → back, L</td>
<td></td>
</tr>
<tr>
<td>Back 0 → back, L</td>
<td></td>
</tr>
<tr>
<td>Back 1 → back, L</td>
<td></td>
</tr>
<tr>
<td>Back W → halt</td>
<td></td>
</tr>
</tbody>
</table>

Stacks

<table>
<thead>
<tr>
<th>1</th>
<th>W</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>W</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

5b. Church-Turing thesis states that anything that can be computed on any physical device can also be computed on a Turing Machine (or Java program). It is not a mathematical theorem, it is a statement about the physical world.
6 (25 points)

6a. What is the current definition of a feasible algorithm. Give two examples explaining why this definition does not fully capture the commonsense meaning of feasible.

Not a perfect definition

\[ t_A(x) = 10^{100} \times \text{len}(x) \rightarrow \text{Feasible in the sense of the definition, not feasible in practice} \]

\[ t_A(x) = 2^{10^{12} \times \text{len}(x)} \rightarrow \text{Feasible in practice but not feasible according to the formal definition} \]

6b. A problem from the class NP has:

1) A feasible algorithm \( t(x,y) \) that given strings \( x \) and \( y \)
2) A polynomial \( P_e(n) \)

And given \( x \) finds \( y \) such that \( t(x,y) \) is true and \( \text{len}(y) \leq P_e(\text{len}(x)) \).

6c. A problem is said to be NP-hard if an algorithm for solving it can be translated into one for solving any other NP-problem, meaning every problem from the class NP can be reduced to this problem.

6d. The knapsack problem is an example of an NP-hard problem.

6e. It is not known whether every NP-hard problem can be solved by a feasible algorithm. It is still an open problem.
7 (10 points)

7a. Let us assume that we have the following sequence of functions: \( f_0(n) = 0 \), \( f_1(n) = n \), \( f_2(n) \) is nowhere defined, \( f_3(n) = n^2 \), etc. Write down the first 4 values \( f(0) \), \( f(1) \), \( f(2) \), and \( f(3) \) of the following diagonal function:

- \( f(n) = f_n(n) + 1 \) if the function \( f_n \) is applicable to \( n \) and
- \( f(n) = 0 \) otherwise.

7b. This diagonal construction is used to prove that there is a problem for which no general algorithm is possible. What is this problem?

7c. For extra credit: how is the diagonal construction used in this proof?

<table>
<thead>
<tr>
<th>( n )</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>( f_0 )</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>( f_1 )</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>( f_2 )</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>( f_3 )</td>
<td>0</td>
<td>1</td>
<td>4</td>
<td>9</td>
</tr>
</tbody>
</table>

7b. This diagonal is used to prove that no algorithm is possible that given a Java program and input data, checks whether it stops or not (halt-checker).

7c. The diagonal construction is used in the proof to consider Java programs w/one input and to show that at the end we have a contradiction.

\( \text{how?} \)
3b. So by pumping lemma there exists a $p$ such that all of $S$ have $\text{len}(s) \leq p$ in which there exists $x,y,z$ such that $s = xyz$ and $\text{len}(y) > 0$ and $\text{len}(xy) \leq p$ and $\forall (xy)^iz \in L$.

$x$ is all before the first instance of a repetition
$y$ is all after the first instance of repetition and before the second instance
$z$ is all after the second instance of repetition.

For $i = 2$

$X = 0$
$Y = 10$
$Z = 1$

3c. ON BACK OF THIS SHEET