Date: __/__/____

1.

a) For a turing decidable language L, the machine M decides language ~~L~~, then the complament is ~~M~~ M' on input w.

M' - on input w

1. Accepts if M rejects
2. Else rejects

M' decides the complement of L as it does the opposite of M.

So, decidable languages are closed under complementaton

1.

1a) Let L_1 and L_2 be two decidable languages and M_1 and M_2 be the turing machines that decides them.

Turing Machine M_0:

$L(M_0) = L_1 \circ L_2$

M_0 = on input w. :

1. Split w into 2 parts , $w = w_1 w_2$
2. Run M_1 on w_1 , if M_1 rejected, then reject
3. Run M_2 on w_2, if M_2 rejected, then reject.
4. Else accept.

Try each possible combination of w. If the parts are accepted vice versa, then w is accepted by M_0. Else, it is rejected.

So, $L(M_0) = L_1 \circ L_2$

The decidable languages are closed under concatanation.

1.

c) Let L be the Turing decidable language and M be the Turing machine that decides L.

There is a turing machine M' such that, it decides the star of $L \Rightarrow L(M') = L^*$.

$M' \Rightarrow$ on input of ω.

1. Split ω into n ~~pars~~ parts: $\omega = \omega_1 \omega_2 \cdots \omega_n$

2. Run M on ω_i for $i = 1, 2, \ldots \ldots n$.

   If M accepts each of these strings, ω_i, accept.

3. All cuts have been tried without success, reject.

If ω is cut into different substrings such that every string is accepted by M, then ω belongs to the star of L, and M' accepts ω after finite no. of steps. Since there are finitely many possible cuts of ω, M' will halt after finitely many steps.

So, $L(M') = L^*$. The decidable languages are closed under star.

Date: __/__/____

2. A DFA accepts some string if and only if reaching an accept state from the start state by travelling along the arrows of the DFA is possible.

let X = " on input <A>, where A is a DFA":

1. Mark the start state of A.

2. Run a breadth first seach from that start state and mark all reachable states from the start state. Repeat until no new states get marked.

3 If no accept state is marked, accept, otherwise reject.

Date: __/__/____

3. Each element in B is an infinite sequence $(b_1, b_2, b_3 \dots)$, where each $b_i \in \{0, 1\}$. Suppose B is countable. Then we can define a correspondance f between $N = \{1, 2, 3\}$ and B.

Specifically for $n \in N$, let $f(n) = (b_{n_1}, b_{n_2} \dots)$ where b_{ni} is the ith bit in the n sequence.

Now, we define the infinite sequence ~~$c = (c_1, c_2, c_3)$~~ $c = (c_1, c_2, c_3 \dots) \in B$, where $c_i = 1 - b_{ii}$ for each $i \in N$. The ith bit in c is the opposite of the ith bit in the ith sequence.

For each $n = 1, 2, 3 \dots$, $c \in B$ differs from the nth sequence in the nth bit, so c does not equal $f(n)$ for any n, which is a contradiction.

Hence B is uncountable.

Date: __/__/____

4.

$A \subseteq B$, $B \subseteq A$, $A \cup B = \Sigma *$.

A and B are recognizable by languages M_A and M_B.

M_A and M_B will accept in a finite amount of time, so M_C will accept or reject in a finite amount of time given that one of these machines ~~accept~~ accepts. As $A \cup B = \Sigma *$, one will accept.

Thus for any string ~~w ∈~~ $w \in \Sigma *$, MC will halt. Language C is thus decidable.

M_C : input string $w \in \Sigma *$.

1. step M_A and M_B over w.
2. If M_B accepts, accept
3. If M_A ~~accepts~~ accepts, ~~reject~~ reject.
4. Otherwise, repeat.

The definition of seperatability indicates that A and B are separable if there exists a decidable language C → such that $A \subseteq C$ and $B \subseteq C$. As shown above, C is decidable. $A \subseteq B \subseteq C$ and $B \subseteq A \subseteq C$ where A and B are disjoint. So the constraints of seperatability have been satisfied.

Date: __/__/___

5.

Let P be the set of all pushdown automata. Let language $L = \{x \in P \mid x$ has a useless state$\}$.

To show L is decidable, we design a turing machine which accepts only strings in L. We know that weather a PDA has an empty language is decidable, and we may reduce the question of whether a given state q is useless to this question by making q the only accept state and then determining whether the resulting PDA has an empty language. If yes, then q is a useless state. Our Turing Machine may therefore solve the question of whether there are any useless states by performing this test for each state in order.

Date: __/__/____

6. Assume A is defined as follows

$A = \{ a^n b^n \mid n \geq 0 \}$ and $B = \{ b \}$, over input $\Sigma = \{a, b\}$.

$f : \Sigma^* \to \Sigma^* :=$

$$f(w) = \begin{cases} b & \text{if } w \in A, \\ a & \text{if } w \notin A \end{cases}$$

→ Notice, if A is a context free language, then it is Turing-decidable.

f is a computable function.

$w \in A$ iff $f(w) = b$, which is true iff $f(w) \in B$

Hence, A is not a regular language, but B is as it is finite.

7 -

The Rice theorem states that whenever we try to analyze any non-trivial property of a recursively enumerable language; then the property will always be undecidable.

a) Here, infiniteness is a non-trivial property as it is satisfied by some languages ($L = \Sigma^*$) and not by others ($L = \{(aa), (ab)\}$.).

So by Rice Theorem, we can say that the given language is undecidable.

b) Here, $L(M)$ being equal to Σ^* signifies that we want $L(M)$ to accept the same strings as Σ^*. This is a non-trivial property as it is satisfied by $L = \Sigma^*$, and rejected by other language.

So, by Rice Theorem, we can say that the given language is undecidable.

Date: __/__/____

8.

A universal Turing machine can be created within the game of life rules. A computer, that can compute the game of life steps, could use this process to calculate if the cells comprise a glider.