INSTRUCTIONS
This **Individual HW5** must be completed without any collaboration with other students in this class. The only allowed sources of help for this homework are the class textbook, notes, and podcast, and the instructional team.

Your homework **must be typed**. We recommend that you submit early drafts to Gradescope so that in case of any technical difficulties, at least some of your work is present. You may update your submission as many times as you'd like up to the deadline.

READING Sipser Sections 3.1, 3.2, 3.3

KEY CONCEPTS Formal definitions of Turing machines, computations of Turing machines, halting computations, implementation-level descriptions of Turing machines, recognizable languages, decidable languages.

1. (**10 points**) The alphabet for this question is $\{0, 1, 2\}$. The state diagrams of the Turing machines for this question are on the last page of the assignment. For each question below, list **all** the machines from the list M_1, M_2, M_3, M_4, M_5 for which the answer is "yes", or write **None** otherwise.

   a. Which of the machines accept the string 0?

   b. Which of the machines accept the string ε?

   c. For which of these machines is it true that the contents of the tape are never changed during a computation?

   d. Which of the machines recognize the set $\emptyset$?

   e. Which of the machines decide the set $\emptyset$?

   f. For which of these machines is it true that on at least one input, the computation of the machine halts and that on at least one input, the computation of the machine does not halt?

No justifications required for credit for this question; but, as always, they're a good idea for your own benefit.

2. (**10 points**). The alphabet for this question is $\{0, 1, 2\}$. The state diagrams of the Turing machines for this question are on the last page of the assignment.

a. The implementation-level description below is **not** a description of M_2.

> "On input w:
>
> 1. If w is the empty string, accept.
> 2. Otherwise, sweep through the tape from left to right, erasing all input characters, until you reach the end of w, and accept."

(i) Change item 2 above to get an implementation-level description that does describe M_2.

(ii) When w is of length k ($k > 0$), how many steps are there in the computation of M_2 on w?

b. In this part of the problem, you will write the formal definition for a new machine that is related to M_4. The new Turing machine is described by the implementation-level description:

> "On input w:
>
> 1. If w is the empty string, reject.
> 2. Otherwise, scan through the tape from left to right until you reach the end of w, and accept."

(i) Fill in the blanks below to define the machine described by this implementation-level description:

$$M = (\{q_0, q_1, q_{acc}, q_{rej}\}, \{0, 1, 2\}, \{0, 1, 2, \square\}, \delta, q_0, q_{acc}, q_{rej})$$

where

$$\delta((q_0, \square)) = \underline{\hspace{3cm}}$$
$$\delta((q_0, x)) = \underline{\hspace{3cm}} \qquad \text{if } x \in \Sigma$$
$$\delta((q_1, \square)) = \underline{\hspace{3cm}}$$
$$\delta((q_1, x)) = \underline{\hspace{3cm}} \qquad \text{if } x \in \Sigma$$
$$\delta((q_{acc}, x)) = (q_{acc}, \square, L) \qquad \text{if } x \in \Gamma$$
$$\delta((q_{rej}, x)) = (q_{acc}, \square, L) \qquad \text{if } x \in \Gamma$$

Hint: The transition function of the Turing machine M has domain

$$\{q_0, q_1, q_{acc}, q_{rej}\} \times \{0, 1, 2, \square\}$$

and codomain

$$\{q_0, q_1, q_{acc}, q_{rej}\} \times \{0, 1, 2, \square\} \times \{L, R\}$$

Make sure the outputs you specify when you fill in the blank are **ordered triples** of the right type.

(ii) What is $L(M)$?

(iii) Is M a decider?

No justifications required for credit for this question; but, as always, they're a good idea for your own benefit.

3. (**10 points**) Suppose M_1 and M_2 are Turing machines. Consider the Turing machines given by the high-level descriptions:

"$M = $ On input w,

1. Run M_1 on input w. If M_1 accepts w, accept. If M_1 rejects w, go to 2.
2. Run M_2 on input w. If M_2 accepts w, accept. If M_2 rejects w, reject."

"$\tilde{M} = $ On input w,

1. Run M_1 on input w. If M_1 rejects w, reject. If M_1 accepts w, go to 2.
2. Run M_2 on input w. If M_2 rejects w, reject. If M_2 accepts w, accept."

For each of the following claims, answer **Always true** if the statement is true for all possible M_1 and M_2; answer **Always false** if the statement is false for all possible M_1 and M_2; and answer **Neither** otherwise.

a. If $w \in L(M_1)$ then $w \in L(M)$.

b. If $w \in L(M_2)$ then $w \in L(M)$.

c. If $w \notin L(M_1)$ then $w \notin L(M)$.

d. If $w \notin L(M_2)$ then $w \notin L(M)$.

e. If $w \in L(M)$ then either $w \in L(M_1)$ or $w \in L(M_2)$ or both.

f. If $w \in L(M_1)$ then $w \in L(\tilde{M})$.

g. If $w \in L(M_2)$ then $w \in L(\tilde{M})$.

h. If $w \notin L(M_1)$ then $w \notin L(\tilde{M})$.

i. If $w \notin L(M_2)$ then $w \notin L(\tilde{M})$.

j. If $w \in L(M_1)$ and $w \in L(M_2)$ then $w \in L(\tilde{M})$.

No justifications required for credit for this question; but, as always, they're a good idea for your own benefit.

State diagrams of Turing machines for Problems 1 and 2. Recall that the alphabet is $\Sigma = \{0, 1, 2\}$.

M_1

q1: self-loop
```
2 ; □ , R
1 ; □ , R
0 ; □ , R
```

q_acc → q1:
```
2 ; □ , R
1 ; □ , R
0 ; □ , R
```

q1 → q_rej:
```
□ ; □ , R
```

q_rej self-loop:
```
□ ; □ , R
2 ; □ , R
1 ; □ , R
0 ; □ , R
```

q_acc → q_rej:
```
□ ; □ , R
```

M_2

q_rej self-loop:
```
2 ; □ , R
1 ; □ , R
0 ; □ , R
```

q0 → q_rej:
```
2 ; □ , R
1 ; □ , R
0 ; □ , R
```

q_rej → q_acc:
```
□ ; □ , R
```

q_acc self-loop:
```
□ ; □ , R
2 ; □ , R
1 ; □ , R
0 ; □ , R
```

q0 → q_acc:
```
□ ; □ , R
```

M_3

q0 self-loop:
```
0 ; □ , R
□ ; □ , R
1 ; □ , R
2 ; □ , R
```

q_acc self-loop:
```
0 ; □ , L
□ ; □ , L
1 ; □ , L
2 ; □ , L
```

q_rej self-loop:
```
0 ; □ , L
□ ; □ , L
1 ; □ , L
2 ; □ , L
```

M_4

q0 self-loop:
```
2 ; 2 , L
1 ; 1 , L
0 ; 0 , L
```

q0 → q_rej:
```
□ ; □ , L
```

q_rej → q_acc:
```
□ ; □ , L
2 ; □ , L
1 ; □ , L
0 ; □ , L
```

q_acc self-loop:
```
□ ; □ , L
2 ; □ , L
1 ; □ , L
0 ; □ , L
```

M_5

q0 self-loop:
```
2 ; □ , L
1 ; □ , L
0 ; 0 , R
```

q0 → q_rej:
```
□ ; □ , L
```

q_rej → q_acc:
```
□ ; □ , L
2 ; □ , L
1 ; □ , L
0 ; □ , R
```

q_acc → q0:
```
0 ; □ , R
1 ; □ , L
2 ; □ , L
□ ; □ , L
```