Lab10-Turing Machine

CS214-Algorithm and Complexity, Xiaofeng Gao & Lei Wang, Spring 2021.

∗ If there is any problem, please contact TA Yihao Xie.
∗ Name:Yanjie Ze Student ID:519021910706 Email: zeyanjie@sjtu.edu.cn

1. Design a one-tape TM M that computes the function $f(x, y) = \lfloor x/y \rfloor$, where x and y are positive integers ($x > y$). The alphabet is $\{1, 0, \square, \triangleright, \triangleleft\}$, and the inputs are x "1"s, $\square$ and y "1"s. Below is the initial configuration for input $x = 7$ and $y = 3$. The result $z = f(x, y)$ should also be represented in the form of z "1"s on the tape with pattern of $\triangleright 111 \cdots 111 \triangleleft$, which is $\triangleright 11 \triangleleft$ for the example.

Initial Configuration

	$\triangleright$	1	1	1	1	1	1	1	$\square$	1	1	1	$\triangleleft$	
	$\uparrow$													
	q_S													

(a) Please describe your design and then write the specifications of M in the form like $\langle q_S, \triangleright \rangle \to \langle q_1, \triangleright, R \rangle$. Explain the transition functions in detail.

Solution.
Set of States:

$$Q = \{q_S, q_Y, q_{CY}, q_{CX}, q_R, q_W, q_{WW}, q_F, q_{FF}, q_{FFF}, q_H\}$$

The meanings of the states are shown in the following table.

state	meaning
q_S	the start state
q_Y	the reading head goes to y
q_{CY}	the reading head does comparison in y
q_{CX}	the reading head does comparison in x
q_W	recovering 1 in y
q_{WW}	recovering 1 in y and avoiding entering x
q_R	writing 1 to the result
q_F	the final state at x
q_{FF}	the final state at y
q_{FFF}	the final state in the output tape
q_H	the end state

Transition Functions:

i. Go to find the start of y:

$$\langle q_S, \triangleright \rangle \to \langle q_Y, \triangleright, R \rangle$$
$$\langle q_Y, 1 \rangle \to \langle q_Y, 1, R \rangle$$
$$\langle q_Y, 0 \rangle \to \langle q_Y, 0, R \rangle$$
$$\langle q_Y, \square \rangle \to \langle q_{CY}, \square, R \rangle$$

ii. Begin to compare the number of 1 in x and y.

A. Find 1 in y and transform it into 0, to represent it has been read. Once we find a 1, we begin to move left.

$$\langle q_{CY}, 0\rangle \to \langle q_{CY}, 0, R\rangle$$
$$\langle q_{CY}, 1\rangle \to \langle q_X, 0, L\rangle$$

B. The reading head needs to move to the position of x.

$$\langle q_X, \square\rangle \to \langle q_{CX}, \square, S\rangle$$
$$\langle q_X, 0\rangle \to \langle q_X, 0, L\rangle$$
$$\langle q_X, 1\rangle \to \langle q_X, 1, L\rangle$$

C. Find 1 in x and transform it into 0, to represent it has been read. Once we find a 1, we begin to move right.

$$\langle q_{CX}, \square\rangle \to \langle q_{CX}, \square, L\rangle$$
$$\langle q_{CX}, 1\rangle \to \langle q_Y, 0, R\rangle$$
$$\langle q_{CX}, 0\rangle \to \langle q_{CX}, 0, L\rangle$$

D. Once all ones in y have been transformed into 0, and the reading head reaches $\triangleleft$, we can **add 1 to the result**. We first transform $\triangleleft$ to $\square$ then write results behind $\square$.

$$\langle q_{CY}, \triangleleft\rangle \to \langle q_R, \square, R\rangle$$
$$\langle q_{CY}, \square\rangle \to \langle q_R, \square, R\rangle$$
$$\langle q_R, \square\rangle \to \langle q_W, 1, L\rangle$$
$$\langle q_R, 1\rangle \to \langle q_R, 1, R\rangle$$

E. After writing one result, we need to recover all ones in y, because they are currently 0. We first move the reading head back and then begin to write.

$$\langle q_W, 0\rangle \to \langle q_W, 0, L\rangle$$
$$\langle q_W, 1\rangle \to \langle q_W, 1, L\rangle$$
$$\langle q_W, \square\rangle \to \langle q_{WW}, \square, L\rangle$$
$$\langle q_{WW}, 0\rangle \to \langle q_{WW}, 1, L\rangle$$
$$\langle q_{WW}, 1\rangle \to \langle q_{WW}, 1, L\rangle$$
$$\langle q_{WW}, \square\rangle \to \langle q_{CY}, \square, S\rangle$$

F. If we successfully transform 1 in y to 0 but fail to find a corresponding 1 in x, the reading head will reach $\triangleright$, this means the result is determined. All 1 for the result has been written.

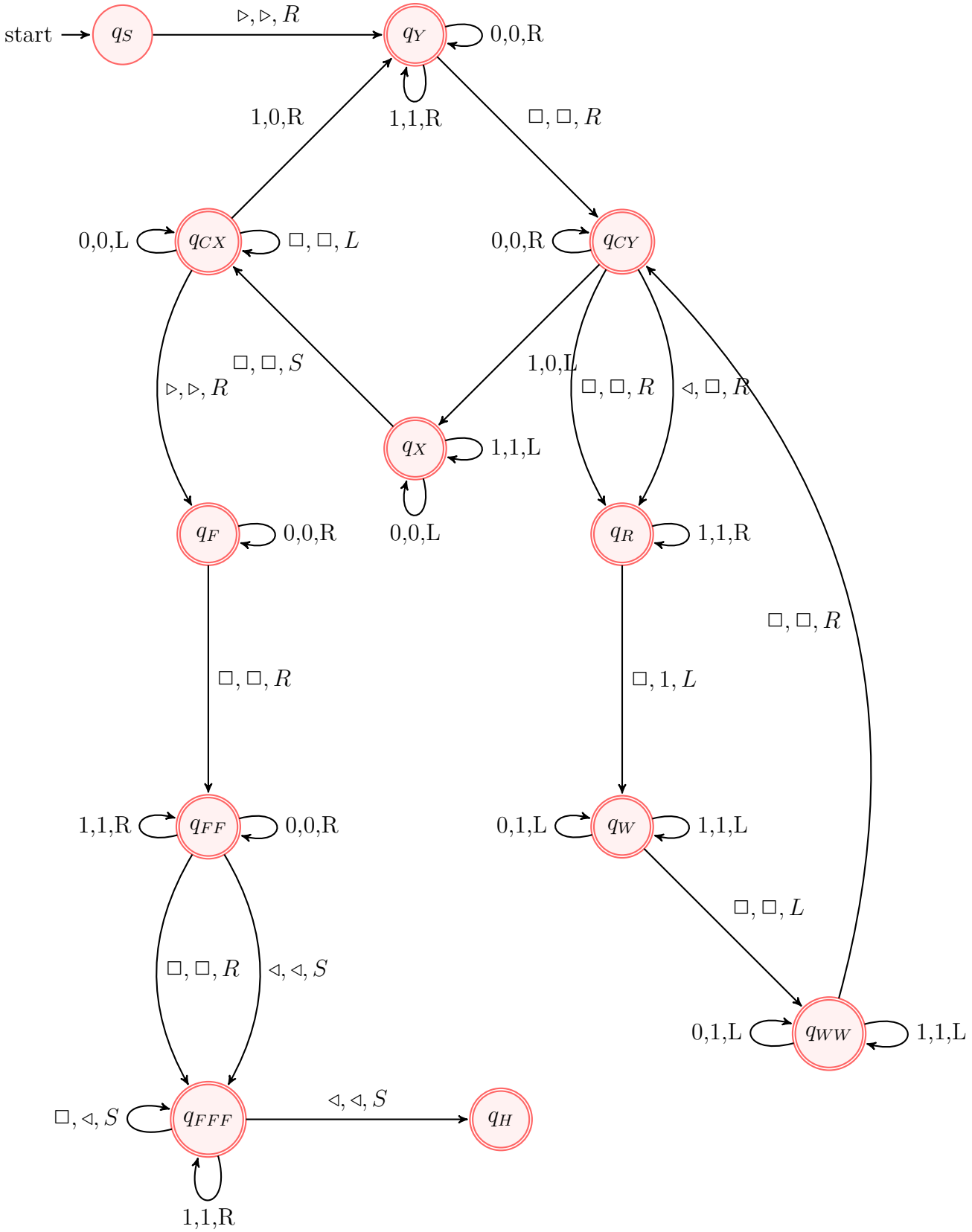
$$\langle q_{CX}, \triangleright\rangle \to \langle q_F, \triangleright, R\rangle$$
$$\langle q_F, 0\rangle \to \langle q_F, 0, R\rangle$$
$$\langle q_F, \square\rangle \to \langle q_{FF}, \square, R\rangle$$
$$\langle q_{FF}, 0\rangle \to \langle q_{FF}, 0, R\rangle$$
$$\langle q_{FF}, 1\rangle \to \langle q_{FF}, 1, R\rangle$$
$$\langle q_{FF}, \square\rangle \to \langle q_{FFF}, \square, R\rangle$$
$$\langle q_{FF}, \triangleleft\rangle \to \langle q_{FFF}, \triangleleft, S\rangle \quad (\textit{special case for no } 1 \textit{ in output})$$
$$\langle q_{FFF}, 1\rangle \to \langle q_{FFF}, 1, R\rangle$$
$$\langle q_{FFF}, \square\rangle \to \langle q_{FFF}, \triangleleft, S\rangle \quad (\textit{write the end symbol})$$

G. The exit state:

$$\langle q_{FFF}, \triangleleft \rangle \rightarrow \langle q_H, \triangleleft, S \rangle$$

□

(b) Please draw the state transition diagram.



(c) Show briefly and clearly the whole process from initial to final configurations for input

$x = 7$ and $y = 3$. You may start like this:

$(q_s, \underline{\triangleright}1111111\square111\triangleleft) \vdash (q_1, \triangleright\underline{1}111111\square111\triangleleft) \vdash^* (q_1, \triangleright1111111\underline{\square}111\triangleleft) \vdash (q_2, \triangleright1111111\square\underline{1}11\triangleleft)$

(Note that for simplicity, we write $(q_1, \triangleright\underline{1}111111\square111\triangleleft) \vdash^* (q_1, \triangleright1111111\underline{\square}111\triangleleft)$ if the corresponding transaction repeats on multiple inputs with the same state.)

Solution.

$$
\begin{aligned}
&(q_S, \underline{\triangleright}1111111\square111\triangleleft)\\
\vdash\ &(q_Y, \triangleright\underline{1}111111\square111\triangleleft)\\
\vdash^*\ &(q_Y, \triangleright1111111\underline{\square}111\triangleleft)\\
\vdash\ &(q_{CY}, \triangleright1111111\square\underline{0}11\triangleleft)\\
\vdash\ &(q_X, \triangleright1111111\underline{\square}011\triangleleft)\\
\vdash\ &(q_{CX}, \triangleright111111\underline{0}\square011\triangleleft)\\
\vdash\ &(q_Y, \triangleright1111110\underline{\square}011\triangleleft)\\
\vdash\ &(q_{CY}, \triangleright1111110\square\underline{0}11\triangleleft)\\
\vdash\ &(q_{CY}, \triangleright1111110\square0\underline{0}1\triangleleft)\\
\vdash\ &(q_X, \triangleright1111110\square\underline{0}01\triangleleft)\\
\vdash\ &(q_X, \triangleright1111110\underline{\square}001\triangleleft)\\
\vdash\ &(q_{CX}, \triangleright111111\underline{0}\square001\triangleleft)\\
\vdash\ &(q_{CX}, \triangleright11111\underline{0}0\square001\triangleleft)\\
\vdash\ &(q_Y, \triangleright1111\underline{0}00\square001\triangleleft)\\
\vdash\ &(q_Y, \triangleright1111100\underline{\square}001\triangleleft)\\
\vdash\ &(q_{CY}, \triangleright1111100\square\underline{0}01\triangleleft)\\
\vdash^*\ &(q_{CY}, \triangleright1111100\square00\underline{0}\triangleleft)\\
\vdash\ &(q_X, \triangleright1111100\square0\underline{0}0\triangleleft)\\
\vdash^*\ &(q_X, \triangleright1111100\underline{\square}000\triangleleft)\\
\vdash\ &(q_{CX}, \triangleright1111\underline{0}0\square000\triangleleft)\\
\vdash^*\ &(q_{CX}, \triangleright111\underline{0}000\square000\triangleleft)\\
\vdash\ &(q_Y, \triangleright1110\underline{0}00\square000\triangleleft)\\
\vdash^*\ &(q_Y, \triangleright1111000\underline{\square}000\triangleleft)\\
\vdash\ &(q_{CY}, \triangleright1111000\square\underline{0}00\triangleleft)\\
\vdash^*\ &(q_{CY}, \triangleright1111000\square000\underline{\square}\square)\\
\vdash\ &(q_R, \triangleright1111000\square000\square\underline{1})\\
\vdash\ &(q_W, \triangleright1111000\square000\underline{\square}1)\\
\vdash\ &(q_{WW}, \triangleright1111000\square00\underline{1}\square1)\\
\vdash\ &(q_{WW}, \triangleright1111000\square0\underline{1}1\square1)\\
\vdash\ &(q_{WW}, \triangleright1111000\square\underline{1}11\square1)
\end{aligned}
$$

$$\vdash (q_{WW}, \triangleright 1111000\underline{\square}111\square 1)$$
$$\vdash (q_{CY}, \triangleright 1111000\square\underline{1}11\square 1)$$

We repeat the same process to get another 1, and we will get:

$$(q_{CY}, \triangleright 1000000\square\underline{0}11\square 11)$$

Then we continue showing the remaining process:

$$\vdash (q_X, \triangleright 1000000\underline{\square}011\square 11)$$
$$\vdash (q_{CX}, \triangleright 1000000\square 011\square 11)$$
$$\vdash^* (q_{CX}, \triangleright\underline{0}000000\square 011\square 11)$$
$$\vdash (q_Y, \triangleright 0\underline{0}00000\square 011\square 11)$$
$$\vdash^* (q_Y, \triangleright 0000000\underline{\square}011\square 11)$$
$$\vdash (q_{CY}, \triangleright 0000000\square\underline{0}11\square 11)$$
$$\vdash (q_{CY}, \triangleright 0000000\square 0\underline{0}1\square 11)$$
$$\vdash (q_X, \triangleright 0000000\square\underline{0}01\square 11)$$
$$\vdash (q_X, \triangleright 0000000\underline{\square}001\square 11)$$
$$\vdash (q_{CX}, \triangleright 000000\underline{0}\square 001\square 11)$$
$$\vdash^* (q_{CX}, \underline{\triangleright}0000000\square 001\square 11)$$
$$\vdash (q_F, \triangleright\underline{0}000000\square 001\square 11)$$
$$\vdash^* (q_F, \triangleright 0000000\underline{\square}001\square 11)$$
$$\vdash (q_{FF}, \triangleright 0000000\square\underline{0}01\square 11)$$
$$\vdash^* (q_{FF}, \triangleright 0000000\square 001\underline{\square}11)$$
$$\vdash (q_{FFF}, \triangleright 0000000\square 001\square\underline{1}1)$$
$$\vdash (q_{FFF}, \triangleright 0000000\square 001\square 1\underline{1}\square)$$
$$\vdash (q_{FFF}, \triangleright 0000000\square 001\square 11\underline{\triangleleft})$$
$$\vdash (q_H, \triangleright 0000000\square 001\square 11\underline{\triangleleft})$$

We reach the final state, and the number of ones behind the second square $\square$ is the result, which is $\lfloor 7/3 \rfloor = 2$. $\square$

2. Given the alphabet $\{1, 0, \square, \triangleright, \triangleleft\}$, design a time efficient 3-tape TM M to compute $f : \{0,1\}^* \to \{0,1\}$ which verifies whether the number of 0 and the number of 1 are the same in an input consisting of only 0's and 1's. M should output 1 if the numbers are the same, and 0 otherwise. For eample, for the input tape $\triangleright 001101\triangleleft$, M should output 1.

   (a) Please describe your design and then write the specifications of M in the form like $\langle q_S, \triangleright, \triangleright\rangle \to \langle q_1, \triangleright, R, R\rangle$. Explain the transition functions in detail.

   Solution.
   The whole process of our turing machine is:

   i. Read every element one by one on the first tape. When the reading head reads 0, it copies 0 to the second tape.

ii. After finishing reading the input, the reading head on the first tape and the reading head on the second tape begin to scan back.
   - If the first reading head encounters 0, it skips and scans next element.
   - If the first reading head encounters 1, and at this time the second reading head check whether there exists 0. If there exists corresponding 0, both reading heads will go to scan next element. If there doesn't exist corresponding 0, TM outputs 0 on the third tape.
   - If the first reading head has finished scanning but the second tape hasn't, TM outputs 0 on the third tape.

iii. After both reading heads finish scanning back and the third tape still has no output, this means the number of 0 and the number of 1 are the same.

Set of states:

$$Q = \{q_S, q_C, q_R, q_T\}$$

Where q_S means the start state, q_C means the copying state, q_R means the comparing state and q_T means the termination state.

Transition functions:

i. Start State:

$$\langle q_S, \triangleright, \triangleright, \triangleright \rangle \to \langle q_C, \triangleright, \triangleright, R, R, R \rangle$$

ii. Begin to copy:
   - 1st tape reads 0 and moves. 2nd tape copies 0 and moves. 3rd tape stays:

$$\langle q_C, 0, \square, \square \rangle \to \langle q_C, 0, \square, R, R, S \rangle$$

   - 1st tape reads 1 and moves. 2nd tape does no copy and stays. 3rd tape stays:

$$\langle q_C, 1, \square, \square \rangle \to \langle q_C, \square, \square, R, S, S \rangle$$

   - 1st tape reads $\triangleleft$ and moves back. 2nd tape moves back. 3rd tape stays:

$$\langle q_C, \triangleleft, \square, \square \rangle \to \langle q_R, \square, \square, L, L, S \rangle$$

iii. Begin to compare:
   - 1st tape reads 1 and 2nd tape reads 0, which is one match. Two tapes moves back. 3rd tape stays:

$$\langle q_R, 1, 0, \square \rangle \to \langle q_R, 0, \square, L, L, S \rangle$$

   - 1st tape reads 0 and 2nd tape reads 0. 1st tape skips this 0 and moves back. Another two tapes stay:

$$\langle q_R, 0, 0, \square \rangle \to \langle q_R, 0, \square, L, S, S \rangle$$

   - 1st tape reads 0 and 2nd tape reaches the head. 1st tape continues to move back:

$$\langle q_R, 0, \triangleright, \square \rangle \to \langle q_R, \triangleright, \square, L, S, S \rangle$$

   - 1st tape reads 1 and 2nd tape reaches the head. This means the number of 0 and the number of 1 are different. TM output 0:

$$\langle q_R, 1, \triangleright, \square \rangle \to \langle q_T, \triangleright, 0, S, S, S \rangle$$

- Both 1st tape and 2nd tape reach the head. This means the number of 0 and the number of 1 are the same. TM output 1:

$$\langle q_R, \triangleright, \triangleright, \square \rangle \rightarrow \langle q_T, \triangleright, 1, S, S, S \rangle$$

- 1st tape reaches the head and 2nd tape reads 0. This means the number of 0 and the number of 1 are different. TM outputs 0:

$$\langle q_R, \triangleright, 0, \square \rangle \rightarrow \langle q_T, 0, 0, S, S, S \rangle$$

□

(b) Show the time complexity for one-tape TM M' to compute the same function f with n symbols in the input and give a brief description of such M' .

Solution.
We assume there are m zeros in the input. m satisfies $0 \leq m \leq n$.
We take the input as **0101101** for an example.
The outline of the algorithm:

i. The reading head starts to copy all zeros to $n+1, n+2, ..., n+m$ positions. We use $\square$ to divide the origin and the copy.
The cost of copying is:

$$T_{copy} = (n+1) + (n+2) + ... + (n+m) = mn + \frac{m^2+m}{2}$$

ii. The reading head begins to compare the copy and the origin. Staring from $\triangleright$:
   A. The reading head reads one zero in the copy and transform it into $\square$. Then the reading head moves to the origin.
   B. If the reading head reads 1 in the origin first, this means one match. Transform it into $\square$. Then the reading head moves back to the copy.
   C. If the reading head reads 0 in the origin first, it needs to skip this to find 1. Transform 1 into $\square$ and scan next.

iii. The termintion condition is:
   A. If all the zeros in the copy have been replaced but the reading head still finds 1 in the origin, TM outputs 0.
   B. If all the ones in the origin have been replaced but the reading head still finds 0 in the copy, TM outputs 0.
   C. If all the elements in the type have become $\square$ and TM has no output yet, TM outputs 1.

Here is one example of inputing **0101101**. TM outputs 0.

After Finishing Copying

	$\triangleright$	0	1	0	1	1	0	1	$\square$	0	0	0	$\triangleleft$	

After One Comparison

	$\triangleright$	0	1	0	1	1	0	$\square$	$\square$	0	0	$\square$	$\triangleleft$	

After Two Comparisons

	▷	0	1	0	1	□	□	□	□	0	□	□	◁	

After Three Comparisons, Output **0**

	▷	0	1	0	□	□	□	□	□	□	□	□	◁	

The cost of comparison is:

$$T_{comp} = O(m+n) \times m = O(m^2 + mn)$$

The total cost is:

$$T_{total} = T_{copy} + T_{comp} = mn + \frac{m^2+m}{2} + O(m^2+mn) = O(m^2+mn)$$

Since $0 \leq m \leq n$, we have:

$$T_{total} = O(n^2)$$

□

3. Define the corresponding decision or search problem of the following problems and give the "certificate" and "certifier" for each decision problem provided in the subquestions or defined by yourself.

   (a) *3-Dimensional Matching.* Given disjoint sets X, Y, Z all with the size of n, and a set $M \subseteq X \times Y \times Z$. Is there a subset M' of M of size n where no two elements of M' agree in any coordinate?

   Solution.
   Decision Problem: Does there **exist** a subset M' of M of size n where no two elements of M' agree in any coordinate?
   Search Problem: Given $M \subseteq X \times Y \times Z$, **find** a subset M' of M of size n where no elements of M agree in any coordinate?
   Certificate: A subset M' of M of size n where no two elements agree in any coordinate
   Certifier: Check that M' is the subset of M and each two elements of M' don't agree in any coordinate, as shown in Alg. 1.

Algorithm 1: Certifier(M',M)

```
1  Input:two set M', M;
2  if M' is not the subset of M then
3      return False;
4  for each element a ∈ M' do
5      for each element b ∈ M' do
6          if a and b are not the same element then
7              if each dimension of a is different from b then
8                  Continue;
9              else
10                 return False;
```

□

(b) *Travelling Salesman Problem.* Given a list of cities and the distances between each pair of cities, find the shortest possible route that visits each city exactly once and returns to the origin city.

Solution.
Decision Problem: Does there **exist** a route that visits each city exactly once and returns to the origin city, with the length of the path $\leqslant k$?

Search Problem: **Find** the shortest possible route that visits each city exactly once and returns to the origin city.

Certificate: A path that visits each city exactly once and returns to the origin city.

Certifier: Check whether the path visits each city exactly once and returns to the origin city, and the length of the path is smaller than k, as shown in Alg. 12.

Algorithm 2: Certifier(P, k)

```
Input: path P = {e_1, e_2, .., e_n}, bound k;
if Along the path P, there exists a city not visited or visited more than once then
    return False;
if Along the path P, the visitor can't return to the origin city then
    return False;
Define sum = 0;
for each edge e in P do
    sum+ = length(e);
if sum ≤ k then
    return True;
else
    return False;
```

□

(c) *Job Sequencing.* Given a set of unit-time jobs, each of which has an integer deadline and a non-negative penalty for missing the deadline. Does there exist a job sequence that has a total penalty $w \leqslant k$?

Solution.
Decision Problem: Does there **exist** a job sequence that has a total penalty $w \leqslant k$?

Search Problem: **Find** a job sequence that has a total penalty $w \leqslant k$.

Certificate: A job sequence of unit-time jobs, each of which has an integer deadline and a non-negative penalty for missing the deadline.

Certifier: Check whether the total penalty of the job sequence is smaller than k, as shown in Alg. 3.

Algorithm 3: Certifier(S, k)

```
Input: job sequence S, bound k;
Define sum_p = 0 as the total penalty;
for each job j ∈ S do
    if j is finished later than the deadline then
        sum_p += penalty(j)
if sum_p ≤ k then
    return True;
else
    return False;
```

□

Remark: Please include your .pdf, .tex files for uploading with standard file names.