

Part<a>.

Input

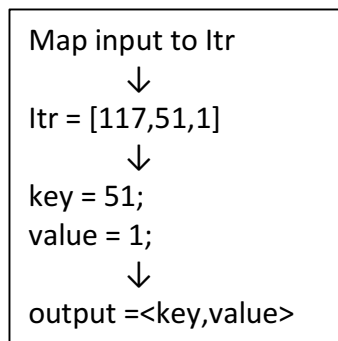
Src	tgt	weight
117	51	1
194	51	1
299	51	3
230	151	51
194	151	79
51	130	10

Function map:

Input : text <117 51 1>

Output: text:key,int:value <51 1>

Process:

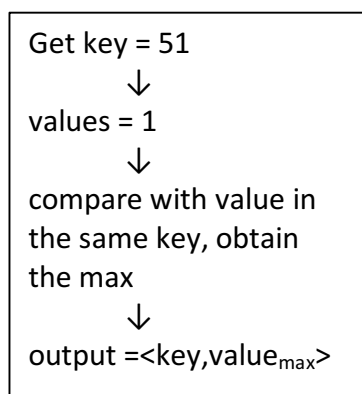


Function reduce:

Input: text:key,int:value <51 1>

Output: text:key,int:value <51 max>

Process:



Output:

Key(unique) value(max)

Part<b>.

Map: map each pair of nodes into a vector<source, target>

Reduce:

```
For each vector <source,target>
  key = source
  intermediate = target
  For each vector <source,target>
    if (source == intermediate & target != key)
      result = target
      output = <result, key>
```

To explain my algorithm, take <4,3> as an example.

```
For vector <4,3>
  key = 4
  intermediate = 3
  For each vector <source,target>
    if (source == intermediate & target != key)
      (which is vector<3,2>)
      result = 2
      output = <2, 4>
```

<2,4> is the last output in the example result, and path is 4->3->2.