COMP9021 Principles of Programming
Term 1, 2024

Coding Quiz 1

Worth 4 marks and due Week 3 Thursday @ 9pm

Description

You are provided with a **stub** in which you need to **insert your code where indicated without doing any changes to the existing code** to complete the task. The current code will generate a **mapping** (that is, a **dictionary**) based on a **seed** and an **upper bound** values provided by the user. Your task is to process the **list of cycles** based on the generated mapping and the **reversed dictionary** as described below.

Marking

List of Cycles	**2 marks**
Reversed Dictionary	**2 marks**
----------	**----------**
Total	**4 marks**

Due Date and Submission

Quiz 1 is due **Week 3 Thursday 29 February 2024 @ 9.00pm** (Sydney time).

Note that **late** submission with **5% penalty per day** is allowed **up to 3 days** from the due date, that is, any late submission after **Week 3 Sunday 3 March 2024 @ 9pm** will be discarded.

Make sure not to change the filename **`quiz_1.py`** while submitting by clicking on **[Mark]** button in **Ed**. It is your responsibility to check that your submission did go through properly using **Submissions** link in Ed otherwise your mark will be **zero** for Quiz 1.

Test Cases

```
$ python quiz_1.py

Enter two integers: 0 4

The generated mapping is:
   {2: 3, 4: 1}

The keys are, from smallest to largest:
   [2, 4]

Properly ordered, the cycles given by the mapping are:
   []

The (triply ordered) reversed dictionary per lengths is:
{1: {1: [4], 3: [2]}}


$ python quiz_1.py

Enter two integers: 0 6

The generated mapping is:
   {1: 1, 3: 3, 5: 6, 6: 6}

The keys are, from smallest to largest:
   [1, 3, 5, 6]

Properly ordered, the cycles given by the mapping are:
   [[1], [3], [6]]

The (triply ordered) reversed dictionary per lengths is:
{1: {1: [1], 3: [3]}, 2: {6: [5, 6]}}


$ python quiz_1.py

Enter two integers: 0 11

The generated mapping is:
   {2: 7, 3: 11, 4: 10, 5: 10, 7: 2, 9: 5, 10: 10, 11: 5}

The keys are, from smallest to largest:
   [2, 3, 4, 5, 7, 9, 10, 11]

Properly ordered, the cycles given by the mapping are:
   [[2, 7], [10]]

The (triply ordered) reversed dictionary per lengths is:
{1: {2: [7], 7: [2], 11: [3]}, 2: {5: [9, 11]}, 3: {10: [4, 5, 10]}}
```

```
$ python quiz_1.py

Enter two integers: 10 9

The generated mapping is:
   {1: 5, 2: 6, 3: 5, 4: 5, 5: 6, 6: 7, 7: 1, 9: 6}

The keys are, from smallest to largest:
   [1, 2, 3, 4, 5, 6, 7, 9]

Properly ordered, the cycles given by the mapping are:
   [[1, 5, 6, 7]]

The (triply ordered) reversed dictionary per lengths is:
{1: {1: [7], 7: [6]}, 3: {5: [1, 3, 4], 6: [2, 5, 9]}}



$ python quiz_1.py

Enter two integers: 20 11

The generated mapping is:
   {2: 4, 3: 9, 4: 4, 5: 8, 6: 2, 7: 5, 8: 11, 9: 1, 10: 10, 11: 5}

The keys are, from smallest to largest:
   [2, 3, 4, 5, 6, 7, 8, 9, 10, 11]

Properly ordered, the cycles given by the mapping are:
   [[4], [5, 8, 11], [10]]

The (triply ordered) reversed dictionary per lengths is:
{1: {1: [9], 2: [6], 8: [5], 9: [3], 10: [10], 11: [8]},
 2: {4: [2, 4], 5: [7, 11]}}



$ python quiz_1.py

Enter two integers: 50 15

The generated mapping is:
   {1: 5, 2: 14, 3: 15, 4: 3, 5: 5, 6: 5, 7: 15, 8: 6, 9: 10, 10: 15, 11: 12,
12: 15, 13: 14, 14: 8, 15: 9}

The keys are, from smallest to largest:
   [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15]

Properly ordered, the cycles given by the mapping are:
   [[5], [9, 10, 15]]

The (triply ordered) reversed dictionary per lengths is:
{1: {3: [4], 6: [8], 8: [14], 9: [15], 10: [9], 12: [11]},
 2: {14: [2, 13]},
 3: {5: [1, 5, 6]},
 4: {15: [3, 7, 10, 12]}}
```

```
$ python quiz_1.py


Enter two integers: 12 38

The generated mapping is:
   {1: 11, 2: 13, 3: 38, 4: 38, 5: 6, 6: 36, 7: 9, 8: 37, 9: 4, 10: 9, 11: 36,
12: 6, 13: 3, 15: 29, 16: 8, 17: 13, 19: 22, 20: 3, 21: 38, 22: 33, 24: 12, 25:
4, 27: 11, 28: 23, 29: 22, 30: 3, 31: 11, 32: 17, 33: 9, 34: 26, 35: 30, 36:
31, 37: 22, 38: 37}

The keys are, from smallest to largest:
   [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 15, 16, 17, 19, 20, 21, 22, 24,
25, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38]

Properly ordered, the cycles given by the mapping are:
   [[4, 38, 37, 22, 33, 9], [11, 36, 31]]

The (triply ordered) reversed dictionary per lengths is:
{1: {8: [16],
     12: [24],
     17: [32],
     23: [28],
     26: [34],
     29: [15],
     30: [35],
     31: [36],
     33: [22]},
 2: {4: [9, 25], 6: [5, 12], 13: [2, 17], 36: [6, 11], 37: [8, 38]},
 3: {3: [13, 20, 30],
     9: [7, 10, 33],
     11: [1, 27, 31],
     22: [19, 29, 37],
     38: [3, 4, 21]}}
```

```
$ python quiz_1.py


Enter two integers: 34 56

The generated mapping is:
   {1: 34, 2: 8, 3: 35, 4: 11, 5: 28, 6: 47, 7: 24, 9: 27, 10: 38, 11: 4, 12:
38, 15: 4, 16: 55, 17: 39, 19: 35, 20: 55, 23: 22, 24: 33, 25: 2, 26: 12, 27:
35, 28: 13, 29: 1, 30: 53, 31: 38, 32: 2, 33: 29, 34: 12, 35: 1, 36: 8, 37: 48,
38: 55, 39: 33, 40: 42, 41: 41, 43: 25, 44: 50, 45: 56, 47: 6, 48: 35, 49: 5
2, 50: 4, 51: 1, 52: 40, 53: 43, 54: 17, 55: 48, 56: 41}

The keys are, from smallest to largest:
   [1, 2, 3, 4, 5, 6, 7, 9, 10, 11, 12, 15, 16, 17, 19, 20, 23, 24, 25, 26, 27,
28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 43, 44, 45, 47, 48,
49, 50, 51, 52, 53, 54, 55, 56]

Properly ordered, the cycles given by the mapping are:
   [[1, 34, 12, 38, 55, 48, 35], [4, 11], [6, 47], [41]]

The (triply ordered) reversed dictionary per lengths is:
{1: {6: [47],
     11: [4],
     13: [28],
     17: [54],
     22: [23],
     24: [7],
     25: [43],
     27: [9],
     28: [5],
     29: [33],
     34: [1],
     39: [17],
     40: [52],
     42: [40],
     43: [53],
     47: [6],
     50: [44],
     52: [49],
     53: [30],
     56: [45]},
 2: {2: [25, 32],
     8: [2, 36],
     12: [26, 34],
     33: [24, 39],
     41: [41, 56],
     48: [37, 55]},
 3: {1: [29, 35, 51], 4: [11, 15, 50], 38: [10, 12, 31], 55: [16, 20, 38]},
 4: {35: [3, 19, 27, 48]}}
```

Hints

(1) The cycles

A **cycle** is a **path** that **starts** and **ends** with the **same key**.

Similarly, a **path** of length **at least 1** in which **no key appears more than once**, except the **first key** is the **same** as the **last key**, is called a **cycle**.

A **cycle** is a list of **keys** `[k1, k2, k3, …, kn]` where the first key `k1` of the list is the `value` of the last key `kn`, that is, the following `key:value` elements must exist in the **mapping** (or **dictionary**):

`k1: k2, k2: k3, …, kn-1: kn, and kn: k1`

For instance, in the example with `10 9` as input, there is one cycle:

`[1, 5, 6, 7]`

since the following **key: value** elements are in the **mapping** (or **dictionary**):

`{1: 5}, {5: 6}, {6: 7}, and {7: 1}`

Make sure when recording the cycle do not repeat the first key at the end, that is, for the following cycle:

```
  1    5     6     7    1
{1:5} {5:6} {6:7} {7:1}
```

It should be recorded as `[1, 5, 6, 7]` not `[1, 5, 6, 7, 1]`

Please also note that the **keys in the cycle are not necessarily ordered**. The only requirement is that the first elements of the cycles are in order (and not the elements within the cycle) as shown in the example with `12 38` as input:

`[[4, 38, 37, 22, 33, 9], [11, 36, 31]]`

The two cycles above are not ordered. However, looking at the first elements of the cycles only, the two cycles are ordered since `4` is smaller than `11`.

(2) The (triply ordered) reversed dictionary per lengths

For instance, in the example with `0 4` as input:

The generated **mapping** is:

```
{2: 3, 4: 1}
```

The (**triply ordered**) reversed dictionary per lengths is:

```
{1:[4], 3:[2]}          ← first generate the reversed dictionary
{1: {1: [4], 3: [2]}}   ← final result
```

In the example with `0 6` as input:

The generated **mapping** is:

```
{1: 1, 3: 3, 5: 6, 6: 6}
```

The (**triply ordered**) reversed dictionary per lengths is:

```
{1:[1], 3:[3], 6:[5,6]}                  ← first generate the reversed dictionary
{1: {1: [1], 3: [3]}, 2: {6: [5, 6]}}    ← final result
```

Triply ordered because there are **three levels of sorting**:

- level 1: per **length** which is the **key** of the **outer dictionary**
- level 2: per **original value** which is the **key** of the **inner dictionary**
- level 3: the **values** of the **inner dictionary** which are **lists** are **sorted**