

CSE250A: Project A: Implementation of chord: DHT

Requirements to run the code:

1. Python 3 installed in machine. (It was tested locally with Python 3.8.2)
2. No extra packages required. All are met by python 3 installation

Navigate to the directory in which the code is stored using cd

To run the code simply run:

```
python chord.py
```

Test cases and screenshots:

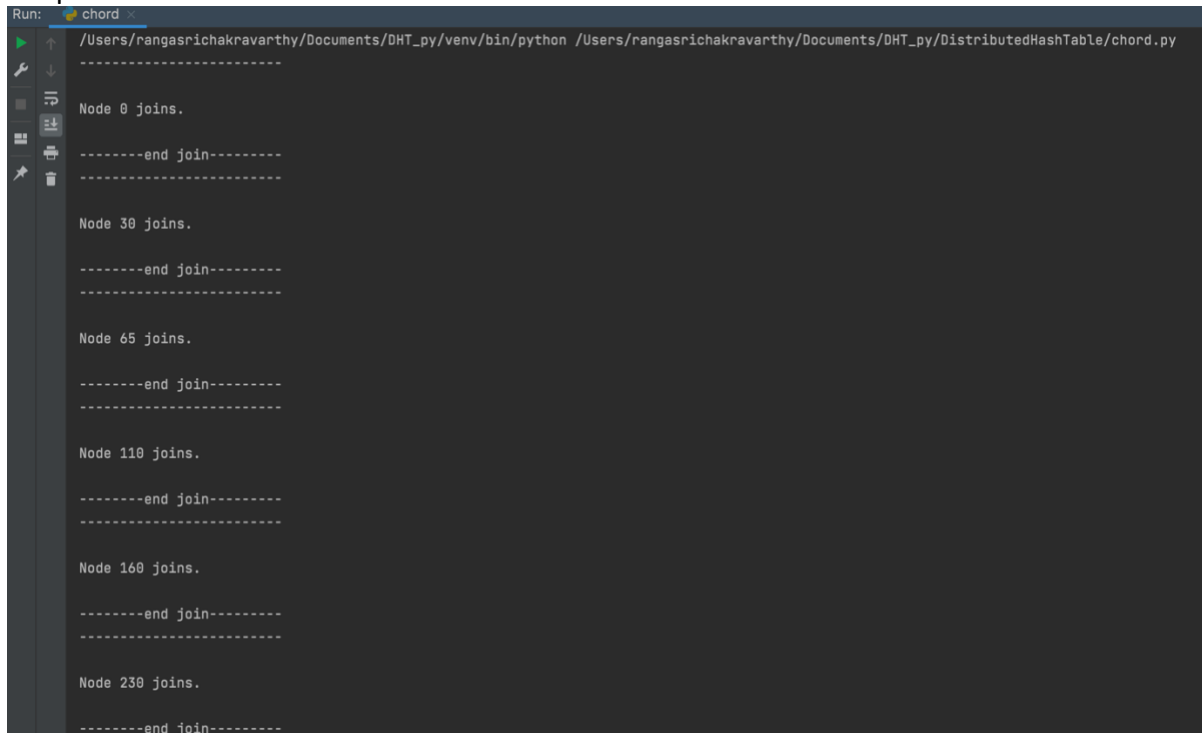
I have implemented the join, find, insert and remove functions and have included the commands to run all these functions in chord.py

The following screenshots show the output of those functions:

Join:

```
n0 = Node(0)
n1 = Node(30)
n2 = Node(65)
n3 = Node(110)
n4 = Node(160)
n5 = Node(230)
n0.join(None)
n1.join(n0)
n2.join(n1)
n3.join(n2)
n4.join(n3)
n5.join(n4)
```

Output:



```
Run: chord
/Users/rangasrichakravarthy/Documents/DHT_py/venv/bin/python /Users/rangasrichakravarthy/Documents/DHT_py/DistributedHashTable/chord.py
-----
Node 0 joins.
-----end join-----
-----
Node 30 joins.
-----end join-----
-----
Node 65 joins.
-----end join-----
-----
Node 110 joins.
-----end join-----
-----
Node 160 joins.
-----end join-----
-----
Node 230 joins.
-----end join-----
```

Printing finger tables of all nodes after all the nodes have joined the chord network:

```
print_all_finger_tables([n0, n1, n2, n3, n4, n5])
```

Output:

```
Finger table of node 0
i | FT[i]

1 | 30
2 | 30
3 | 30
4 | 30
5 | 30
6 | 65
7 | 65
8 | 160
```

```
Finger table of node 30
i | FT[i]

1 | 65
2 | 65
3 | 65
4 | 65
5 | 65
6 | 65
7 | 110
8 | 160
```

```
Finger table of node 65
i | FT[i]

1 | 110
2 | 110
3 | 110
4 | 110
5 | 110
6 | 110
7 | 160
8 | 230
```

```
Finger table of node 110
i | FT[i]

1 | 160
2 | 160
3 | 160
4 | 160
5 | 160
6 | 160
7 | 230
8 | 0
```

```
Finger table of node 160
i | FT[i]

1 | 230
2 | 230
3 | 230
4 | 230
5 | 230
6 | 230
7 | 230
8 | 65
```

```
Finger table of node 230
i | FT[i]

1 | 0
2 | 0
3 | 0
4 | 0
5 | 0
6 | 30
7 | 65
8 | 110
```

Insert:

```
n0.insert(3, 3)
n1.insert(200)
n2.insert(123)
n3.insert(45, 3)
n4.insert(99)
n2.insert(60, 10)
n0.insert(50, 8)
n3.insert(100, 5)
n3.insert(101, 4)
n3.insert(102, 6)
n5.insert(240, 8)
n5.insert(250, 10)
```

Output:

```
Key 3 was inserted into Node 30
Key 200 was inserted into Node 230
Key 123 was inserted into Node 160
Key 45 was inserted into Node 65
Key 99 was inserted into Node 110
Key 60 was inserted into Node 65
Key 50 was inserted into Node 65
Key 100 was inserted into Node 110
Key 101 was inserted into Node 110
Key 102 was inserted into Node 110
Key 240 was inserted into Node 0
Key 250 was inserted into Node 0
-----
```

New node joins:

```
n6 = Node(100)
n6.join(n5)
print_all_finger_tables([n0, n1, n2, n3, n4, n5, n6])
```

Output:

```
Node 100 joins.

Key 99 is moved from successor Node 110 to Node 100
Key 100 is moved from successor Node 110 to Node 100
-----end join-----
```

Finger tables of all nodes after new node join:

<p>Finger table of node 0</p> <p>i FT[i]</p> <p>1 30</p> <p>2 30</p> <p>3 30</p> <p>4 30</p> <p>5 30</p> <p>6 65</p> <p>7 65</p> <p>8 160</p>	<p>Finger table of node 30</p> <p>i FT[i]</p> <p>1 65</p> <p>2 65</p> <p>3 65</p> <p>4 65</p> <p>5 65</p> <p>6 65</p> <p>7 100</p> <p>8 160</p>	<p>Finger table of node 65</p> <p>i FT[i]</p> <p>1 100</p> <p>2 100</p> <p>3 100</p> <p>4 100</p> <p>5 100</p> <p>6 100</p> <p>7 160</p> <p>8 230</p>	<p>Finger table of node 110</p> <p>i FT[i]</p> <p>1 160</p> <p>2 160</p> <p>3 160</p> <p>4 160</p> <p>5 160</p> <p>6 160</p> <p>7 230</p> <p>8 0</p>
---	---	---	--

<p>Finger table of node 160</p> <p>i FT[i]</p> <p>1 230</p> <p>2 230</p> <p>3 230</p> <p>4 230</p> <p>5 230</p> <p>6 230</p> <p>7 230</p> <p>8 65</p>	<p>Finger table of node 230</p> <p>i FT[i]</p> <p>1 0</p> <p>2 0</p> <p>3 0</p> <p>4 0</p> <p>5 0</p> <p>6 30</p> <p>7 65</p> <p>8 110</p>	<p>Finger table of node 100</p> <p>i FT[i]</p> <p>1 110</p> <p>2 110</p> <p>3 110</p> <p>4 110</p> <p>5 160</p> <p>6 160</p> <p>7 230</p> <p>8 230</p>
---	--	--

Printing the key, value pair stored in all nodes:

```
n0.print_key_value_pairs()
n1.print_key_value_pairs()
n2.print_key_value_pairs()
n3.print_key_value_pairs()
n4.print_key_value_pairs()
n5.print_key_value_pairs()
n6.print_key_value_pairs()
```

Output:

```
-----Node id: 0-----
{240: 8, 250: 10}
-----Node id: 30-----
{3: 3}
-----Node id: 65-----
{45: 3, 50: 8, 60: 10}
-----Node id: 110-----
{101: 4, 102: 6}
-----Node id: 160-----
{123: None}
-----Node id: 230-----
{200: None}
-----Node id: 100-----
{99: None, 100: 5}
```

Find / Lookup:

Lookup all the keys inserted on nodes n0, n1, n2 and n6:

```
nodes = [n0, n1, n2, n6]
keys_to_find = [3, 200, 123, 45, 99, 60, 50, 100, 101, 102, 240, 250]
find all keys in all nodes(keys to find, nodes)
```

Output:

```
-----Node 0-----
Look-up result of key 3 from node 0 with path [0, 30] value is 3
Look-up result of key 200 from node 0 with path [0, 230] value is None
Look-up result of key 123 from node 0 with path [0, 160] value is None
Look-up result of key 45 from node 0 with path [0, 65] value is 3
Look-up result of key 99 from node 0 with path [0, 100] value is None
Look-up result of key 60 from node 0 with path [0, 65] value is 10
Look-up result of key 50 from node 0 with path [0, 65] value is 8
Look-up result of key 100 from node 0 with path [0, 100] value is 5
Look-up result of key 101 from node 0 with path [0, 110] value is 4
Look-up result of key 102 from node 0 with path [0, 110] value is 6
Look-up result of key 240 from node 0 with path [0] value is 8
Look-up result of key 250 from node 0 with path [0] value is 10
```

-----Node 30-----

Look-up result of key 3 from node 30 with path [30] value is 3
Look-up result of key 200 from node 30 with path [30, 230] value is None
Look-up result of key 123 from node 30 with path [30, 160] value is None
Look-up result of key 45 from node 30 with path [30, 65] value is 3
Look-up result of key 99 from node 30 with path [30, 100] value is None
Look-up result of key 60 from node 30 with path [30, 65] value is 10
Look-up result of key 50 from node 30 with path [30, 65] value is 8
Look-up result of key 100 from node 30 with path [30, 100] value is 5
Look-up result of key 101 from node 30 with path [30, 110] value is 4
Look-up result of key 102 from node 30 with path [30, 110] value is 6
Look-up result of key 240 from node 30 with path [30, 0] value is 8
Look-up result of key 250 from node 30 with path [30, 0] value is 10

-----Node 65-----

Look-up result of key 3 from node 65 with path [65, 30] value is 3
Look-up result of key 200 from node 65 with path [65, 230] value is None
Look-up result of key 123 from node 65 with path [65, 160] value is None
Look-up result of key 45 from node 65 with path [65] value is 3
Look-up result of key 99 from node 65 with path [65, 100] value is None
Look-up result of key 60 from node 65 with path [65] value is 10
Look-up result of key 50 from node 65 with path [65] value is 8
Look-up result of key 100 from node 65 with path [65, 100] value is 5
Look-up result of key 101 from node 65 with path [65, 110] value is 4
Look-up result of key 102 from node 65 with path [65, 110] value is 6
Look-up result of key 240 from node 65 with path [65, 0] value is 8
Look-up result of key 250 from node 65 with path [65, 0] value is 10

-----Node 100-----

Look-up result of key 3 from node 100 with path [100, 30] value is 3
Look-up result of key 200 from node 100 with path [100, 230] value is None
Look-up result of key 123 from node 100 with path [100, 160] value is None
Look-up result of key 45 from node 100 with path [100, 65] value is 3
Look-up result of key 99 from node 100 with path [100] value is None
Look-up result of key 60 from node 100 with path [100, 65] value is 10
Look-up result of key 50 from node 100 with path [100, 65] value is 8
Look-up result of key 100 from node 100 with path [100] value is 5
Look-up result of key 101 from node 100 with path [100, 110] value is 4
Look-up result of key 102 from node 100 with path [100, 110] value is 6
Look-up result of key 240 from node 100 with path [100, 0] value is 8
Look-up result of key 250 from node 100 with path [100, 0] value is 10

Remove:

```
n0.remove(3)
find all keys in all nodes(keys to find, nodes)
```

Output:

```
Key 3 has been removed from the Node 30
```

Print values of lookup for node 0 and node 1 for all key value pairs:

```
-----Node 0-----
Key: 3 is not present in node 30
Look-up result of key 200 from node 0 with path [0, 230] value is None
Look-up result of key 123 from node 0 with path [0, 160] value is None
Look-up result of key 45 from node 0 with path [0, 65] value is 3
Look-up result of key 99 from node 0 with path [0, 100] value is None
Look-up result of key 60 from node 0 with path [0, 65] value is 10
Look-up result of key 50 from node 0 with path [0, 65] value is 8
Look-up result of key 100 from node 0 with path [0, 100] value is 5
Look-up result of key 101 from node 0 with path [0, 110] value is 4
Look-up result of key 102 from node 0 with path [0, 110] value is 6
Look-up result of key 240 from node 0 with path [0] value is 8
Look-up result of key 250 from node 0 with path [0] value is 10
```

```
-----Node 30-----
Key: 3 is not present in node 30
Look-up result of key 200 from node 30 with path [30, 230] value is None
Look-up result of key 123 from node 30 with path [30, 160] value is None
Look-up result of key 45 from node 30 with path [30, 65] value is 3
Look-up result of key 99 from node 30 with path [30, 100] value is None
Look-up result of key 60 from node 30 with path [30, 65] value is 10
Look-up result of key 50 from node 30 with path [30, 65] value is 8
Look-up result of key 100 from node 30 with path [30, 100] value is 5
Look-up result of key 101 from node 30 with path [30, 110] value is 4
Look-up result of key 102 from node 30 with path [30, 110] value is 6
Look-up result of key 240 from node 30 with path [30, 0] value is 8
Look-up result of key 250 from node 30 with path [30, 0] value is 10
```

References:

1. Chord paper: <https://conferences.sigcomm.org/sigcomm/2001/p12-stoica.pdf>
2. Professor Chen Qian's lectures
3. Video explanation of the concepts of chord: <https://www.youtube.com/watch?v=qqv4OJ5Lc4E>
4. Lecture notes of Professor Steven Gordon:
<https://sandilands.info/sgordon/teaching/its413y12s2/unprotected/ITS413Y12S2L08-P2P.pdf>