Client invocation:

1. ./bvDHT\_client
   1. Creates a new, empty DHT
2. ./bvDHT\_client peerAddress peerPort
   1. ./bvDHT\_client 10.92.21.15 33333
   2. Connects to an existing DHT through specified peer

Data maintained by each node:

* myAddr
* successorAddr
* predecessorAddr
* fingerTable of addresses
* data in key-val form

Data Definitions:

* key : 40-byte SHA1, hex digest string
  + import hashlib
  + hashlib.sha1(msg.encode()).hexdigest()
* sz : 4-byte unsigned integer in little-endian format
* nodeAddress : [sz][ipAddress:port]
  + Example: [17][10.92.21.15:33333]
* [T/F]: A 1-byte “T” or “F”

Protocols:

* [x] Connect
* [x] Disconnect
* [x] Insert(key, val)
* [x] Get(key)
* [x] Remove(key)
* [x] Closest Peer (key)
* [x] Contains
* [x] Predecessor Update // telling new succ we are pred

**Closest Peer**

Client: [CLOP][key]

Peer: [nodeAddress]

**Get Value**

Client: [GETV][key]

Peer: [T/F] // Acknowledge ownership of range

If ‘T’ continue

Peer: [T/F] // Confirm possession of item

If ‘T’ continue

Peer: [sz][value]

**Insert**

Client: [INST][Key]

Peer: [T/F] // Acknowledge ownership of range

If ‘T’ continue

Client: [sz][value]

Peer: [T] // Confirm insertion of data

**Remove**

Client: [RMVE][Key]

Peer: [T/F] // Acknowledge ownership of range

If ‘T’ continue

Peer: [T/F] // Acknowledge removal of key-value pair

**Contains**

Client: [CONT][key]

Peer: [T/F] // Acknowledge ownership of range

If ‘T’ continue

Peer: [T/F] // Confirm possession of item

If ‘T’ continue

**Connect**

Client: [CONN][myAddr]

Peer: [T/F] // Acknowledge ownership of range

If ‘T’ continue

Peer: [Successor nodeAddr]

Peer: [sz #files]

For each file (sz)

Peer: [key][sz][val]

// Client makes PRUP request to Peer successor

Client: [T]

**Predecessor Update**

Client: [PRUP][myAddr]

Peer: [T]

**Disconnect**

Client: [DISC]

Client: [successorAddr]

Client: [sz #files]

For each file (sz)

Client: [key][sz][val]

// Peer makes PRUP request to our Successor

Peer: [T]