import numpy as np

MOD = 11

NULLVALUE = -32768

class Node:

def \_\_init\_\_(data\_value = None):

self.data\_value = data\_value

self.data\_node = None

def hash(value):

return key % MOD

def insert\_hash(hash\_table, value):

add = hash(value)

hash\_table[add] = Node(value)

def search\_hash(hash\_table, value):

add = hash(value)

if hash\_table[add] == NULLVALUE:

print(‘value %d not found’ % value)

return add

# data = np.array([3, -1, -6, 5, 2, -4, 8, 11, 13])

data = [13, 5, 12, 1, 8, 11, 9, 6, 4, 16]

sum = 17

# pos\_hash\_table = np.ones((1, 100)) \* NULLKEY

node\_init = Node(NULLKEY)

pos\_hash\_table = [node\_init] \* 10

proper\_pairs = [ ]

for idata in data:

factor\_another = sum – idata

add = hash(factor\_another)

node\_add = pos\_hash\_table[add]

# empty node

if node\_add.data\_value == NULLKEY:

insert\_hash(pos\_hash\_table, idata)

else:

Flag\_Found = False

while node\_add.data\_value != NULLKEY:

if node\_add.data\_value == factor\_another:

proper\_pairs.append([idata, factor\_another])

Flag\_Found = True

break

else:

if node\_add.data\_node is not None:

node\_add = node\_add.data\_node

else:

break

if not Flag\_Found:

insert\_hash(pos\_hash\_table, idata)

print(proper\_pairs)