A black screen with white text

Description automatically generated

class MinSumListHeap:

def \_\_init\_\_(self):

self.heap = []

def add\_list(self, new\_list):

self.heap.append(new\_list)

self.\_heapify\_up(len(self.heap) - 1)

def pop\_smallest(self):

if not self.heap:

return None

smallest\_list = self.heap[0]

if len(smallest\_list) > 1:

self.heap[0] = self.heap.pop()

self.\_heapify\_down(0)

else:

self.heap.pop(0)

return smallest\_list

def \_heapify\_up(self, index):

while index > 0:

parent\_index = (index - 1) // 2

if self.\_get\_sum(self.heap[index]) < self.\_get\_sum(self.heap[parent\_index]):

self.heap[index], self.heap[parent\_index] = self.heap[parent\_index], self.heap[index]

index = parent\_index

else:

break

def \_heapify\_down(self, index):

while True:

left\_child\_index = 2 \* index + 1

right\_child\_index = 2 \* index + 2

smallest = index

if (left\_child\_index < len(self.heap) and

self.\_get\_sum(self.heap[left\_child\_index]) < self.\_get\_sum(self.heap[smallest])):

smallest = left\_child\_index

if (right\_child\_index < len(self.heap) and

self.\_get\_sum(self.heap[right\_child\_index]) < self.\_get\_sum(self.heap[smallest])):

smallest = right\_child\_index

if smallest != index:

self.heap[index], self.heap[smallest] = self.heap[smallest], self.heap[index]

index = smallest

else:

break

def \_get\_sum(self, lst):

return sum(lst)

# Example usage:

min\_heap\_list = MinSumListHeap()

min\_heap\_list.add\_list([1, 3, 5])

min\_heap\_list.add\_list([2, 4, 6])

min\_heap\_list.add\_list([0, 7, 8])

print("Pop smallest:")

print(min\_heap\_list.pop\_smallest()) # [0, 7, 8]

print(min\_heap\_list.pop\_smallest()) # [1, 3, 5]

print(min\_heap\_list.pop\_smallest()) # [2, 4, 6]

print(min\_heap\_list.pop\_smallest()) # None, as the heap is empty

A screenshot of a computer

Description automatically generatedclass MinSumHeap:

def \_\_init\_\_(self):

self.data = []

def add(self, num\_list):

self.data.extend(num\_list)

self.data.sort()

def \_\_iter\_\_(self):

return iter(self.data) A screenshot of a computer

Description automatically generated