i) Vector class, assuming others is set of uncollected coins. And self as agent pos.

Manhattan:

```
def __manhattan__(self, other):
    return sum(list(map(abs, self.__sub__(other))))
```

Manhattan max:

```
def __manhattan_max__(self, others):
    return max(list(map(self.__manhattan__, others)))
```

ii)

Manhattan Sum

```python
def __manhattan_sum__(self, others):
    return sum(list(map(self.__manhattan__, others)))
```

iii) Manhattan Ordered Sum:

```python
def __manhattan_ordered_sum__(self, others):
    if len(others)<=0:
        return 0
    ordered = sorted(others, key=lambda x: self.__manhattan__(x))
    total = self.__manhattan__(ordered[0])
    for index, val in enumerate(ordered[:-1]):
        total += ordered[index].__manhattan__(ordered[index + 1])
    return total
```


A
3
2
1
Collected coin ids: 0, 1, 2, 3
Solution found!
Plan length: 128
States expanded: 3597
States visited: 3806
Total time: 74.098s

iv)
My Fancy Heuristic:
```
def __sub__(self, other):
   return self.x - other.x, self.y - other.y

def __l2_sq_(self, other):
   d = self.__sub__(other)
   sq = d[0] ** 2, d[1] ** 2
   return sum(sq)

def __l2_sq_min__(self, others):
   return min(list(map(self.__l2_sq_, others)))

def __l2_min_coins_penalty__(self, uncollected, all_coins, coins_collected, max_dist):
   if(len(uncollected)<=0):
       return 0
   return self.__l2_sq_min__(uncollected) + ((len(all_coins) - len(coins_collected)) *
max_dist)

Heuristic Call:
return agent_pos.__l2_min_coins_penalty__(uncollected, all_coins, collected,
                                         (state.grid.width ** 2 + state.grid.height **
2))
```

1)
Explanation for admissibility:
The heuristic is as follows:
Squared manhattan min (closest coin)+ num uncollected coins *max possible manhattan min distance
This shows that distance will never be over estimated as it is always at least:
num _coins * (grid.width**2+gird.height**2)
Hence it is impossible to underestimate the distance.

2)
Works arbitrarily on any state of this problem. Nothing is assumed.

3) Results: time: 12s, states visit: 915, states expand: 476: plan:124 (should be less)
Motivations: Provides penalty for states where coins are not collected, this way we just greedily collect the closest coin, and find a plan quickly, an identified plan can then be optimised. The tree could also be pruned to not re-explore states that have already been explored after collecting a coin, this would be faster as well.


A
3
2
1
Collected coin ids: 0, 1, 2, 3
Solution found!
Plan length: 124
States expanded: 476
States visited: 615
Total time: 11.630s