一. 采用深度优先搜索的策略，在搜索过程中通过优先选取面值较大的硬币来决定优先搜索对象实现搜索树剪枝。

具体实现中，维护一个最小堆，将 $amount$ 减去各个面值的硬币后所剩金额压入最小堆中，弹出顶层元素，再减去不大于其的所有面值并压入最小堆，直至顶层元素为 0（或者未找到结果），同时使用一个字典来记录硬币个数。

Algorithm 1 min_nums

Input: an array of coins, the total amount

Output: the min numbers used to match the amount if found, -1 if not found

```
Min_heap heap
Map path
heap.push(amount)
path.put(amount, 1)
while heap is not empty do
    top = heap.pop()
    len = path.get(top)
    if top == 0 then
        return len
    end if
    if top < min(coins) then
        continue
    end if
    for coin in coins do
        if top − coin ≥ 0 then
            heap.push(top − coin)
            path.put(top − coin, len + 1)
        end if
    end for
end while
return -1
```

二:

(1)

$g(n)$ 为从起点到节点 n 的代价。

$h*(n)$ 为从节点 n 回到起点（不经过已经走过的节点）的优化路径代价。

(2)

记

$h(1) = 0,$

$h(2) = 2,$

$h(3) = 3,$

$h(4) = 6.$

搜索树如下所示:

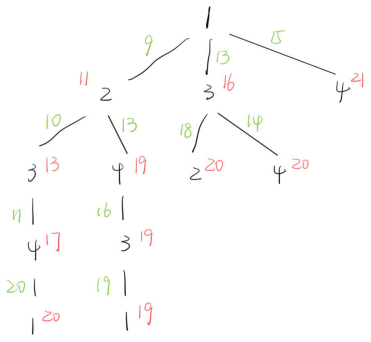

Figure 1: serch tree