```
1  ##################################################################
   ##################################################################
   ###
2  #Question 1
3  def powerN(m,n):
4      if n == 0:
5          return 1
6      elif n < 0:
7          return 1/powerN(m,-n)
8      else:
9          return m * powerN(m, n - 1)
10 #print(power(3,-2))
11 ##################################################################
   ##################################################################
   ###
12 #Question 3
13 def hocBuilder(height,count=0):
14     if height==0:
15         return 0
16     elif count==0:
17         return 3 + hocBuilder(height, count+1)
18     elif count<height:
19         return 5 + hocBuilder(height, count+1)
20     else:
21         return 5
22 #print(hocBuilder(4))
23 ##################################################################
   ##################################################################
   ###
24 #Question 5 - (a)
25 def patternA(length,count=0):
26     if count<length:
27         count+=1
28         return subLoopA1(count) + "\n" + patternA(length,
   count)
29     else:
30         return ""
31 def subLoopA1(length,count=1):
32     if count<length:
33         return str(count) + " " + subLoopA1(length, count
   +1)
34     else:
35         return str(count)
36 #print(patternA(5))
37 ##################################################################
   ##################################################################
   ###
38 #Question 5 - (b)
39 def patternB(length,count=0):
40     if count < length:
41         count += 1
```

```
42          return subLoopB1(length, count) + "\n" + patternB
    (length, count)
43      else:
44          return ""
45  def subLoopB1(loop, length, count=1):
46      if count == 1 :
47          if length>1:
48              return subLoopB2((loop-length)*2) + str(count
    ) + " " + subLoopB1(loop, length, count + 1)
49          else:
50              return subLoopB2((loop-length)*2) + str(count
    )
51      elif count<length:
52          return str(count) + " " + subLoopB1(loop, length
    , count+1)
53      else:
54          return str(count)
55  def subLoopB2(count):
56      if count!=0:
57          return subLoopB2(count-1) + " "
58      else:
59          return ""
60  #print(patternB(5))
61  ##########################################################
    ##########################################################
    ###
62  #Question 6
63  class FinalQ:
64      def print(self,array,idx):
65          if(idx<len(array)):
66              profit = self.calcProfit(array[idx])
67              if idx == 0:
68                  print("{}. Investment: {}; Profit: {}".
    format(idx + 1, array[idx], profit) + "\n" + self.print(
    array,idx + 1))
69              else:
70                  return "{}. Investment: {}; Profit: {}"
    .format(idx + 1, array[idx], profit) + "\n" + self.print(
    array,idx + 1)
71          else:
72              return ""
73
74      def calcProfit(self,investment,curr=0,profit=4.5,
    increment=100):
75          investment-=investment%100
76          if curr==0:
77              if investment>100000:
78                  x=investment-100000
79                  return float(self.calcProfit(75000,1,45,
    1000)+self.calcProfit(x,1,800,10000)+self.calcProfit(x%
    10000,1,8))
```

```
            else:
                return float(self.calcProfit(investment-25000,1))
        elif curr<investment:
            return profit+self.calcProfit(investment, curr+increment,profit,increment)
        else:
            return 0

# Tester
array=[25000,100000,250000,350000,10010000]
# Last input is for limit testing, and is the highest value this algorithm can take in
# I could just remove the limit, that would allow me to enter higher values, but it's completely unnecessary because the values I can reach is already pretty high.
f = FinalQ()
f.print(array,0)

# Expected -
# 1. Investment: 25000; Profit: 0.0
# 2. Investment: 100000; Profit: 3375.0
# 3. Investment: 250000; Profit: 15375.0
# 4. Investment: 350000; Profit: 23375.0

# Result -
# 1. Investment: 25000; Profit: 0.0
# 2. Investment: 100000; Profit: 3375.0
# 3. Investment: 250000; Profit: 15375.0
# 4. Investment: 350000; Profit: 23375.0

```