CSE 321 – Introduction to Algorithm Design

Homework 05

Deadline: 30 December 23:55

PLEASE DO NOT FORGET TO READ THE NOTES CAREFULLY

1. **(35 points)** There are two unsigned integer lists *X* and *Y* with the same number of elements (N). The following relation exists among their i^{th} elements:

$$1 \leq X_i \leq Y_i$$

The *cost (S)* of a list *X* is defined as:

$$S = \sum_{i=2}^{N} |X_i - X_{i-1}|$$

Given a list Y*,* design and implement a dynamic programming algorithm in Python 3 that returns the maximum possible cost of list *X*.

Explain your algorithm and do worst case analysis as a comment block at the beginning of your implementation file, maximum_cost_StudentID.py

Your code will be tested with many different inputs as follows:

```
Y = [14,1,14,1,14]
cost = find_maximum_cost(Y)
print(cost)
#Output: 52
Y = [1,9,11,7,3]
cost = find_maximum_cost(Y)
print(cost)
#Output: 28
Y = [50,28,1,1,13,7]
cost = find_maximum_cost(Y)
print(cost)
#Output: 78
```

2. **(35 points)** A large amount of money was given to you and to protect, you split and buried it in a land with n*m cells (dimension is n row m column). Each cell in the land contains a positive integer which is the number of Turkish liras buried. A thief has learned your secret and going to steal maximum amount of money before the cops arrive. Initially the thief is at first column but can be at any row. He can move only 1 cell right, right-up or right-down from a given cell. When the thief arrives a new cell, he steals the money. Design and implement a dynamic programming algorithm in Python 3 to find out maximum amount of money he can steal when he reaches to the end (last column). Explain your algorithm and do worst case analysis as a comment block at the beginning of your implementation file, theft_StudentID.py. Your code will be tested with many different inputs as follows:

```
amountOfMoneyInLand= [[1,3,1,5], [2,2,4,1], [5,0,2,3], [0,6,1,2]]
res = theft(amountOfMoneyInLand)
print(res)
#Output: 16
amountOfMoneyInLand= [[10,33,13,15], [22,21,4,1], [5,0,2,3], [0,6,14,2]]
res = theft(amountOfMoneyInLand)
print(res)
#Output: 83
```

3. (**30 points**) A *Decent Number* having N digits has the following properties:
   1. Its digits can only be *3*'s and/or *5*'s.
   2. The number of *3*'s it contains is divisible by *5*.
   3. The number of *5*'s it contains is divisible by *3*.
   4. If there is more than one such number, we pick the largest one.

   Design and implement a greedy algorithm to find the decent number having N digits. If there is not any decent number with N digits, return -1. Explain your algorithm and your greedy choice as a comment block at the beginning of your implementation file, decentNumber_StudentID.py. Your code will be tested with many different inputs as follows:

```
dn = decentNumber(1)
print(dn)
#Output: -1
dn = decentNumber(3)
print(dn)
#Output: 555
dn = decentNumber(5)
print(dn)
#Output: 33333
dn = decentNumber(11)
print(dn)
#Output: 55555533333
```

Explanation

For $N = 1$, there is no decent number having 1 digit (so we print -1).

For $N = 3$, 555 is the only possible number. The number 5 appears three times in this number, so our count of 5's is evenly divisible by 3 (*Decent Number Property 3*).

For $N = 5$, 33333 is the only possible number. The number 3 appears five times in this number, so our count of 3's is evenly divisible by 5 (*Decent Number Property 2*).

For $N = 11$, 55555533333 and all permutations of these digits are valid numbers; among them, the given number is the largest one.