## <u>**Question**</u>: [Huffman Code]

## <u>**Content**</u>:

If we store every char with 8bits, the memory usage will be very inefficient. However, we can save lots of space if we encode every alphabet according to its frequency of use. Huffman code is thus a born for this situation.

## <u>**Input**</u>:

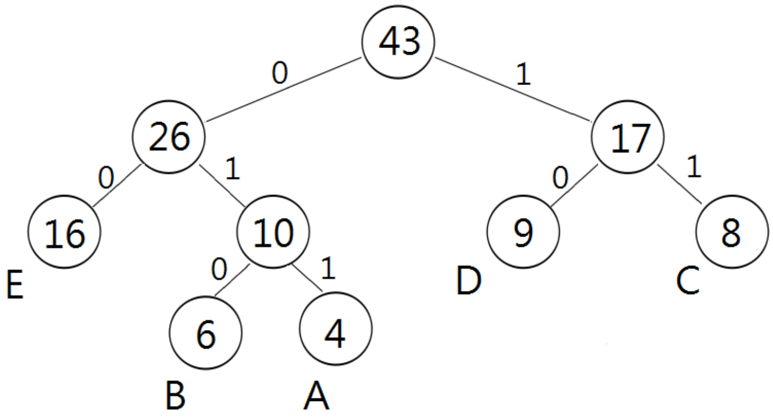
1. The first line is the number (<10) of the set(s).
2. Every set has three lines. The first line comes **n** alphabet; the second line comes their frequency (<1000) of use respectively; the third line is -1 which means the end of the set. Every number is positive integer separated with space. We guarantee that there are no duplicate numbers.
3. Note that the size of the frequency is going to affect the result, such as, (sample input) the frequency of an input "B A" is "6 4", then "B" must put on the left side of "A".

## <u>**Output**</u>:

1. You have to output each coding of the alphabet. The output order is in descending number according to the alphabet's coding.
2. An alphabet and it's coding must be separated by a space.

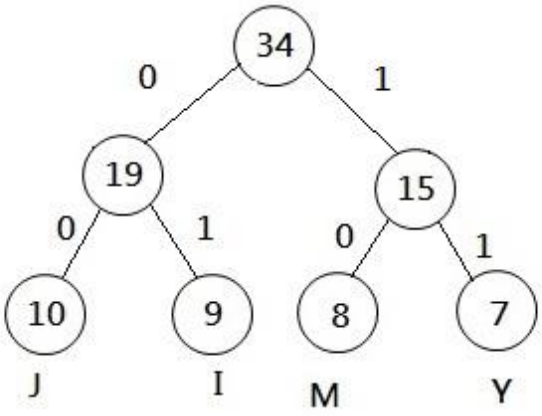
**Sample Input:**

```
2
E D C B A
16 9 8 6 4
-1
J I M Y
10 9 8 7
-1
```



**Sample Output:**

```
A 011

B 010

C 11

D 10

E 00

Y 11

M 10

I 01

J 00
```


34
0
1
19
15
0
1
0
1
10
9
8
7
J
I
M
Y