CS6102– Computational Thinking

Week -11

01– Feb-2023

I. Ciphering a Sentence

Activity – 1:

Use the rules below to create ciphers.

Create two rules for mapping the alphabet.

Sample rules:

- The characters are divided into two groups: (1) characters for which the image of their uppercase form has an enclosed area (such as P or O) and (2) characters that do not have an enclosed area in in the image of their uppercase form (such as I or Z).
- Sort the two groups alphabetically, with group 1 first and then group 2.

Apply the rules.

Applying rule 1:

Group 1: {A, B, D, O, P, Q, R}

Group 2: {C, E, F, G, H, I, J, K, L, M, N, S, T, U, V, W, X, Y, Z}

Applying rule 2:

A, B, D, O, P, Q, R, C, E, F, G, H, I, J, K, L, M, N, S, T, U, V, W, X, Y, Z

Place the letter sequence that resulted from Rule 2 in the second row of the following table beneath the number 1 to 26 (example below).

Alphabet Mapping #1.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26
A	B	D	O	P	Q	R	C	E	F	G	H	I	J	K	L	M	N	S	T	U	V	W	X	Y	Z

Now, invent a third rule and apply it to the above mapping to come up with another one-to-one alphabet mapping table. A third rule could be to place the even-number-mapped letters in alphabetical order followed by the odd-number-mapped letters in alphabetical order. And let it be Alphabet Mapping #2.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26

Activity 2: Encode Sentences

Encode sentences using the mapping developed in Activity 1.

a. Think of a simple message you would like to send to your friend.
b. Encode it three times: first use just Alphabet Mapping #1, then use just Alphabet Mapping #2
c. If you have done a mapping on your own, use that for encoding.

For example, if the sentence is:

We drove to the gym

Using the Alphabet Mapping #1, we find each letter of the sentence in the second row of the table created in Activity 1. We replace the letter with the corresponding number in the first row.

Please note that one must place letter breaks (the character "_") and separate each word by a blank space. In this example, the encoded sentence is:

23_9 3_7_4_22_9 20_4 20_12_9 11_25_17

1. Perform deciphering the following sentence.
   - **a.** 10_6_20_23 22_18_23_23_14_19_18 25_14_23 25_17_20_10_10_18_9 20_9 4_20_16_6_18_17 **(Using Alphabet mapping #2)**
   - **b.** 20_14_22 14_1_8_18 18_2 4_20_16_6_18_17 10_6_20_23.
2. Invent a *fourth rule* and apply it to the alphabet mapping#3 to come up with another one-to-one alphabet mapping table. A fourth rule could be placing the vowels at the first few positions followed by even-number-mapped letters in alphabetical order and then the odd-number-mapped letters in alphabetical order. And let it be Alphabet Mapping #3
   - ***a.*** *This message was written in cipher*
3. Invent a *fifth rule* and apply it to the alphabet mapping#3 to come up with another one-to-one alphabet mapping table. A fifh rule could be placing the odd-number-mapped letters in alphabetical order at the first followed by even-number-mapped letters in alphabetical order. And let it be Alphabet Mapping #4
   - ***a.*** *We had the best experience of the nuances of computational world.*
4. Answer the following
   - a. Imagine that another person gives you an encoded message and you do not know the mapping tables. How would you go about deciphering or decoding the sentences? What patterns would you look for to help you discover the one-to-one mapping?
   - b. Write a procedure and draw a flowchart for implementing any version of the cipher in a computer program.

II Debugging: spot the difference

****

The following is a correct fragment of code (in Java) for sorting an array.

```
public static void Sort(int[]a, int n)
      {
      for (int p=1; p <= n-1; p++)
            {
            for (int i=0; i < n-p; i++)
                  {
                  if (a[i] > a[i+1])
                        {
                        swap(a, i, i+1);
                        }
                  }
            }
      }
```

****

The following version has mistakes. Can you spot 16 differences?

```
public static Sort(int{}a; int m)
      {
      for (int p=0, p < n-1, p++)
            {
            For (int i; i < n-q; i++)
                  {
                  if (a[i] < a[i+1];
                        {
                        Swap(a, i, i+l)
                        }
                  }
            }
****************************************************************
```

Pondering over a problem of complexity

Identify a solution to the problem, and show how it might be computed by hand.

A hospital supervisor needs to create a weekly schedule for four nurses, subject To the following conditions

1. Each day is divided into three 8-hour shifts.
2. On each day, all nurses are assigned to different shifts and one nurse has the day off.
3. Each nurse works five or six days a week.
4. No shift is staffed by more than two different nurses in a week.
5. If a nurse works shifts 2 or 3 on a given day, he must also work the same shift either the previous day or the following day.

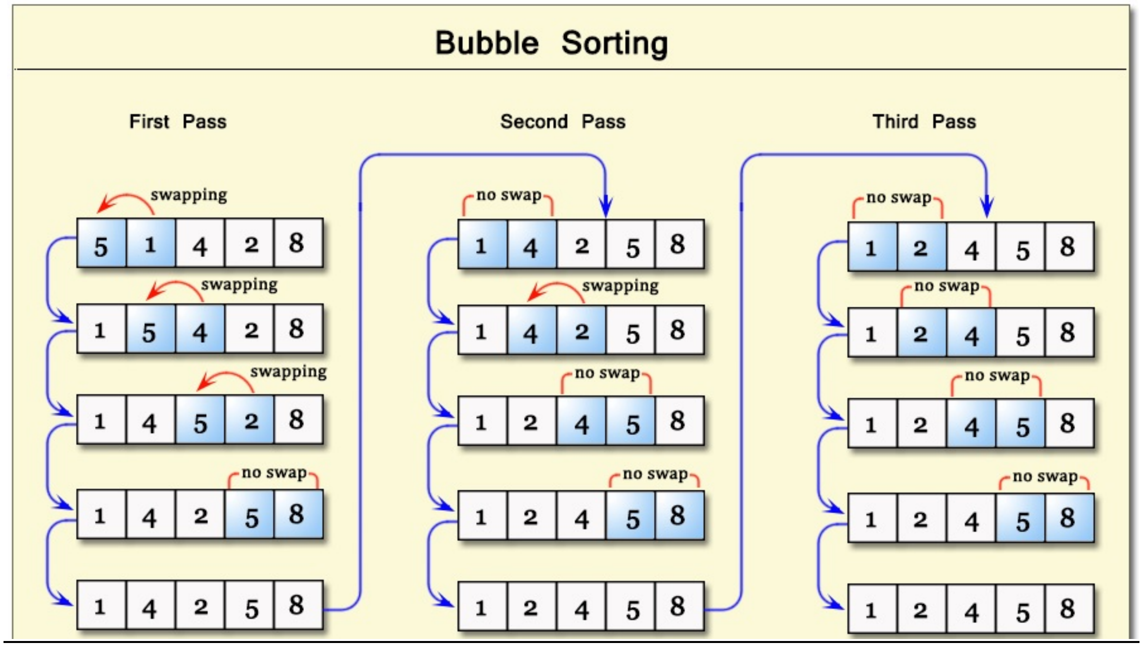
Two ways to formulate the problem

- ✓ Assign nurses to shifts
- ✓ Assign shifts to nurses

UNDERSTANDING THE ESSENSE OF CT through:

(References: slides-bubblesort.ppt, bubblesort-explained.pdf)

- ➢ Sorting algorithms (Bubble sort)
- ➢ Efficiency of algorithms



FUN-TIME

1. https://play2048.co/ or https://mgarciaisaia.github.io/2048/ or Android app 2048
2. https://sudharshank-10.github.io/Algobot