**19CSE302 – Design and Analysis of algorithms**

**Assignment – 10.11.2021**

* **R.Abhinav**
* **CB.EN.U4CSE19453**

1. Stella is hosting a Party. Stella invited few guests for the dinner and the party is arranged in the garden with few dining tables. Suppose that the host wants her friends to know each other (There can be a situation where Stella’s friends may not know each other), so she wants her friends to get acquainted with each other. Vertices of the graph represent the guests who attend the party and edges means that they know each other. So Stella plans to follow a certain seating arrangements, such that no two guests in same table are known to each other. Use Branch and Bound method to solve the problem, also implement the solution.
2. **Puzzle Pegs :**
3. **S**ay that a pattern P of length m is a circular substring of a text T of length n if there is an index 0 ≤ i < m, such that P = T[n − m + i..n − 1] + T[0..i − 1], that is, if P is a substring of T or P is equal to the concatenation of a suffix of T and a prefix of T. Give an O(n +m)-time algorithm for determining whether P is a circular substring of T.
4. A crime has been committed in the city and the forensics person is able to identify some unknown DNA sample ‘UD’ on the crime scene. On the other hand, police has identified ‘N=5’ suspects who have the highest probability to be the murderer. The probability of a suspect to be a murderer is obtained by matching his DNA with unknown DNA ‘UD’. Arrange the suspects in the order of their decreasing probability to be a murderer.

**Code:**

def SlidingWindow(main\_text, sub\_text):  
  
 main\_len = len(main\_text)  
 if main\_len < len(sub\_text):  
 return -1  
 main\_occ = [0] \* 256  
   
 start = 0  
 result = main\_len + 1  
 count = 0  
  
 for i in sub\_text:  
 main\_occ[ord(i)] += 1  
 if main\_occ[ord(i)] == 1:  
 count += 1  
  
 j = 0  
 i = 0  
  
 while(j < main\_len):  
 main\_occ[ord(main\_text[j])] -= 1  
 if main\_occ[ord(main\_text[j])] == 0:  
 count -= 1  
  
 while count == 0:  
 if result > j - i + 1:  
 result = j - i + 1  
 start = i  
 main\_occ[ord(main\_text[i])] += 1  
 if main\_occ[ord(main\_text[i])] > 0:  
 count += 1  
 i += 1  
 j += 1  
 if result > main\_len:  
 return "-1"  
 return main\_text[start:start+result]  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 main\_text = input()  
 sub\_text = input()  
 result = SlidingWindow(main\_text, sub\_text)  
 print("Smallest window that contain all character :", result)