Naivee String Matching Algorithm

Refernce : https://medium.com/@krupa_110/the-naive-string-matching-algorithm-be7992ebbd1d

```
M = P.length  //length of the pattern
N = T.length  //length of the text
for s = 0 to N - M:
  if [1,2,..,M] == T[s+1, ..., s+M]
    print "pattern occurs with a shift"
```

Worst Case Time complexity = O((n-m+1) * m)

Best Case Time complexity = O(n)

- Naive pattern searching is the simplest method among other pattern searching algorithms.
- It checks for all character of the main string to the pattern
- It is an **Exact string matching algorithm**
- It is helpful for smaller texts.
- Does not need any pre-processing phases
- We can find substring by checking once for the string
- It doesnot occupy extra space to perform the operation
- The naive approach tests all the possible placement of Pattern P[1, .., m] relative to text T[1, .., n]. We try shift **s = 0,1,...., n-m ,** successively and for each shift s, compare T[s+1, ..., s+m] to P[1, .., m]. It returns all the valid shifts found

Advantages:

1. No Pre-processing phase required because the running time of Naive-String-Matcher is equal to its matching time.

2. No extra space are needed.
3. Also,the comparisons can be done in any order.

Disadvantage:

Naive method is inefficient because information from a shift is not used again.