String Matching Naïve Approach:

The naive string matching algorithm checks for a pattern in a text by comparing each character of the pattern with the corresponding character of the text.

Algorithm:

1. Iterate through the text with a loop variable **i** from 0 to **n - m**, where **n** is the length of the text and **m** is the length of the pattern.
2. For each **i**, check if the pattern matches the text starting from **i**.

Code:

```
#include <iostream>
#include <string>
using namespace std;

void naiveStringMatch(string text, string pattern) {
   int n = text.length();
   int m = pattern.length();

   for (int i = 0; i <= n - m; i++) {
      int j;
      for (j = 0; j < m; j++) {
         if (text[i + j] != pattern[j])
            break;
      }
      if (j == m)
         cout << "Pattern found at index " << i << endl;
   }
}

int main() {
   string text = "ababcabcabababcabc";
   string pattern = "abc";
   naiveStringMatch(text, pattern);
   return 0;
}
```

Output:

```
Pattern found at index 2
Pattern found at index 10
```

Space complexity: _____

Justification:_____
_____

Time complexity:

Best case time complexity: _____

Justification:_____
_____

Worst case time complexity: _____

Justification:_____