# Decode

## Problem J

Luo Tsz Fung {pepper1208}

The 1st Buddhist Sin Tak College Computer Club Programming Contest

October 18, 2024

stpc();

# Background

Problem Idea by `pepper1208`
Preparation by `pepper1208`, `rina__owo`

# Problem Restatement

Given two string, $S$ and $T$.

Check if the number of occurrence of $T$ in $S$ is **exactly** $n$ or not.

If yes, output `Yes`. Otherwise, output `No`.

# Statistics

Points are given per checkpoint in this problem. There are 20 checkpoints in this question.

Attempts: 26

| | | | | | |
|---|---|---|---|---|---|
| 0 points | 9 | + | 0 | = | 9 |
| 40 points | 0 | + | 1 | = | 1 |
| 45 points | 0 | + | 1 | = | 1 |
| 70 points | 0 | + | 2 | = | 2 |
| 100 points | 0 | + | 2 | = | 2 |

First solved by **Chan Tsz Hang** at **1h 20m 27s**

# Solution

This is a very classic **string processing** problem.

Treat $S$ and $T$ as two arrays. (Session 3!)

Your mission is to find $T$ in $S$. If found, add the counter by 1.

Finally, check if the counter equals to $n$. Then, output an appropriate answer.

# Solution

Set up a for loop to iterate every character in *S*.

For each character *i*, compare `S.substr(i, T.size())` with *T*.
If those two are the same, process the increment of the counter.

**Score: 100!**

# Not using C++ STL

How to compare two strings, not using C++ STL?

## Observation 1

If the size of $S$ and $T$ are not the same, then $S \neq T$.

## Observation 2

Assume the size of $S$ and $T$ is $n$. If $S_i = T_i$ for $1 \leq i \leq n$, then $S = T$.

# Not using C++ STL

You can facilitate the two observations by adding one more `for loop` inside.

Check every **part of string** in $S$ with different starting characters.

For every part of string $S'$, check if $S'[i+m] = T[m]$, for $0 \leq m \leq$ `T.size()`.

# Sometimes, you can be smart!

As points are given per checkpoints, you could observe that the answer could be only `Yes` or `No`.

Try to only output them!

Score of only output `Yes`: 70
Score of only output `No`: 30

# Takeaways

1. Be familiar with C++ STL and its facilitation.
2. Sometimes be smart, rather than working on the question with a long time.