**6 kyu**

**String subpattern recognition I**

14295% of 6536 of142[GiacomoSorbi](https://www.codewars.com/users/GiacomoSorbi)

C++

* [TRAIN AGAIN](https://www.codewars.com/kata/string-subpattern-recognition-i/train/cpp)
* [NEXT KATA](https://www.codewars.com/trainer/cpp)

Details

[Solutions](https://www.codewars.com/kata/string-subpattern-recognition-i/solutions/cpp)

[Discourse (19)](https://www.codewars.com/kata/string-subpattern-recognition-i/discuss/cpp)

* Add to Collection
* |
* Share this kata:

In this kata you need to build a function to return either true/True or false/False if a string can be seen as the repetition of a simpler/shorter subpattern or not.

For example:

hasSubpattern("a") == false; //no repeated pattern

hasSubpattern("aaaa") == true; //created repeating "a"

hasSubpattern("abcd") == false; //no repeated pattern

hasSubpattern("abababab") == true; //created repeating "ab"

hasSubpattern("ababababa") == false; //cannot be entirely reproduced repeating a pattern

Strings will never be empty and can be composed of any character (just consider upper- and lowercase letters as different entities) and can be pretty long (keep an eye on performances!).

If you liked it, go for the [next kata](https://www.codewars.com/kata/string-subpattern-recognition-ii/" \t "_blank) of the series!

<https://www.codewars.com/kata/string-subpattern-recognition-i/cpp>

#include <iostream>

#include <stdio.h>

#include <string>

using namespace std;

bool hasSubpattern( std::string& s){

  //your code here

  //your code here

    string subs = "";

    for(int len =1; len <s.length(); len++)

    {

        subs = s.substr(0, len);

        //Console.WriteLine(subs);

        bool flag = true;

        int j = 0;

        while (j < s.length())

        {

            string s2 = "";

            if (j + len > s.length())

            {

                s2 = s.substr(j);

            }

            else

            {

                s2 = s.substr(j, len);

            }

            if(subs != s2)

            {

                flag = false;

                break;

            }

            j += len;

        }

        if (flag)

        {

            return true;

        }

    }

    return false;

}

int main() {

    string s = "abababab";

    //string s =    "aa";

    cout << hasSubpattern(s) << endl;

    return 0;

}