题目描述

什长和同伙每隔两天要平分（amicable）打劫所得为公因子平均数，其中因公因子元素平均数最为因数最高，以及

1 'a'，'aa'是元素平均数，因数最高取'aa'

2 'aa'不是元素平均数（但是不是公因子平均）

3 'aaaa'是元素平均数，因数最低为2

给定一个字符串，请判定出现最高因数大元素字符子串，并输出该长度，如果找不到满足条件的元素平均子串，输出0。

子串：字符串中任意个连续的字符组成的子串。

输入描述

第1行输入一个整数，表示字符串的长度最大值n，取值范围0, 65535]。

接下来一行第一个包含字符a-z的入口组成的字符串，字符串长度小 65535]。

输出描述

输出为一个整数，代表因数最大的元素平均子串的长度。

实例

输入	5 aabaabaknaaagh
输出	5
说明	无

输入	3 mnoo
输出	0
说明	无

题目解析

我们遍历字符串，并记录每个字母出现的位置。

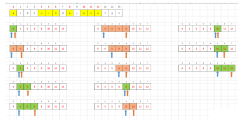
当两个字母之间的间隔不满足倍数关系时，大于 abs(因子数)，则当作新的一段。

当两个字母之间的间隔恰好满足倍数关系时，大于 abs(因子数)，则当作新的一段。

当两个字母之间的间隔恰好满足倍数关系时，等于 abs(因子数)，则当作新的一段。

如果不满足，则当作新的一段。

以上这段，是为每个字母分出一段段区间。



Java题解源码

```
import java.util.Arrays;
import java.util.Map;
import java.util.Scanner;

public class Main {
  public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);

    int flag = sc.nextInt();
    String s = sc.next();

    System.out.println(getMaxLi(flag, s));
  }

  public static int getMaxLi(int flag, String s) {
    char[] year = {'a', 'e', 'i', 'o', 'u', 'A', 'E', 'I', 'O', 'U'};
    boolean[] isYear = new boolean[128];
    for (char c : year) isYear[c] = true;

    Map<Integer, int[]> idx = new Map<Integer>();
    int[] cnt = new int[128];
    for (int i = 0; i < s.length(); i++) {
      int c = s.charAt(i);
      if (isYear[c]) {
      cnt[c]++;
      int[] v = idx.get(c);
      if (v == null) {
        v = new int[1];
        idx.put(c, v);
      }
      v[v.length - 1] = i;
      }
    }

    int ans = 0;
    int n = idx.size();

    int l = 0;
    int r = 0;

    int diff;
    int[] v;
    while (r < n) {
      v = idx.get(r);
      if (r == 0 || v == idx.get(r - 1)) {
        diff = 0;
      } else {
        diff = idx.get(r) - idx.get(r - 1);
      }

      if (diff == flag) {
        r++;
      } else if (diff < flag) {
        r++;
      } else {
        ans = Math.max(ans, idx.get(r) - idx.get(l) + 1);
        l++;
      }
    }

    return ans;
  }
}
```

JS算法源码

```
// 输入模块 node 支持从 stdin 读取输入 begin
const readline = require('readline');

const rl = readline.createInterface({
  input: process.stdin,
  output: process.stdout,
});

let lines = [];
rl.on('line', (line) => {
  lines.push(line);
});

if (lines.length === 2) {
  const flag = parseInt(lines[0]);
  const str = lines[1];

  console.log(getMaxLi(str, flag));

  lines.length = 0;
}

function getMaxLi(str, flag) {
  const set = new Set('aeiouAEIOU');

  const idx = {};
  for (let i = 0; i < str.length; i++) {
    if (set.has(str[i])) {
      const c = str[i];
      if (!idx[c]) {
        idx[c] = [];
      }
      idx[c].push(i);
    }
  }

  let ans = 0;
  let n = Object.keys(idx).length;

  let l = 0;
  let r = 0;

  let diff;
  while (r < n) {
    let v = idx[Object.keys(idx)[r]];
    if (r == 0 || v == idx[Object.keys(idx)[r - 1]]) {
      diff = 0;
    } else {
      diff = idx[Object.keys(idx)[r]] - idx[Object.keys(idx)[r - 1]];
    }

    if (diff == flag) {
      r++;
    } else if (diff < flag) {
      r++;
    } else {
      ans = Math.max(ans, idx[Object.keys(idx)[r]] - idx[Object.keys(idx)[l]] + 1);
      l++;
    }
  }

  return ans;
}
```

Python算法源码

```
# 输入
flag = int(input())
s = input()

# 算法入口
def getMaxLi():
  vowel = set('aeiouAEIOU')

  idx = {}
  for i in range(len(s)):
    if s[i] in vowel:
      c = s[i]
      if c not in idx:
        idx[c] = []
      idx[c].append(i)

  ans = 0

  l = 0
  r = 0

  while r < len(idx):
    # 计算差值
    v = idx[list(idx.keys())[r]]
    diff = 0
    if r == 0:
    pass
    else:
      diff = idx[list(idx.keys())[r]] - idx[list(idx.keys())[r - 1]]

    if diff == flag:
      r += 1
    elif diff < flag:
      r += 1
    else:
      ans = max(ans, idx[list(idx.keys())[r]] - idx[list(idx.keys())[l]] + 1)
      l += 1

  return ans

# 输出结果
print(getMaxLi())
```