题目描述

一个整数可以由连续的自然数之和来表示。

给定一个整数，计算该整数有几种连续自然数之和的表达式，且打印出每种表达式

输入描述

一个目标整数T (1 <=T<= 1000)

输出描述

该整数的所有表达式和表达式的个数。

如果有多种表达式，输出要求为：自然数个数最少的表达式优先输出，每个表达式中按自然数递增的顺序输出，具体的格式参见样例。

在每个测试数据结束时，输出一行"Result:X"，其中X是最终的表达式个数。

用例

输入	9
输出	9=9 9=4+5 9=2+3+4 Result:3
说明	整数 9 有三种表示方法，第1个表达式只有1个自然数，最先输出。 第2个表达式有2个自然数，第2次序输出。 第3个表达式有3个自然数，最后输出。 每个表达式中的自然数都是按递增次序输出的。 数字与符号之间无空格

输入	10
输出	10=10 10=1+2+3+4 Result:2
说明	无

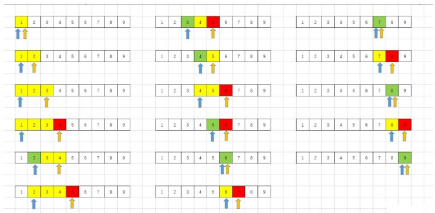
题目解析

本题可以考虑使用 滑动窗口🔍。

比如输入9，则生成一个数组arr = [1,2,3,4,5,6,7,8,9]。

然后用left、right指针同时指向arr的索引0，并取arr[0]为初始sum值

left指针和right指针的移动逻辑如下：



right指针从左向右开始移动，每移动一次就计算left~right之间的子数组和赋值给sum，并且判断：

- sum > target 若true，则left++，sum -= arr[left]
- sum === target，若true，则保存此时的子数组到res中，然后left++且right++，计算sum += arr[right] - arr[left]。此步有可能right指针会处于越界位置，因此注意越界检查。
- sum < target，若true，则right++，计算sum+=arr[right]

当left都移动到数组尾部时，结束。

Java算法源码

```java
import java.util.ArrayList;
import java.util.Arrays;
import java.util.Scanner;
import java.util.StringJoiner;

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

  public static void getResult(int t) {
    int[] arr = new int[t];
    for (int i = 0; i < t; i++) arr[i] = i + 1;

    ArrayList<int[]> ans = new ArrayList<>();

    int l = 0;
    int r = 1;

    int sum = arr[l];
    while (l < t) {
      if (sum > t) {
        sum -= arr[l++];
      } else if (sum == t) {
        ans.add(Arrays.copyOfRange(arr, l, r));
        sum -= arr[l++];
        if (r >= t) break;
        sum += arr[r++];
      } else {
        sum += arr[r++];
      }
    }

    ans.sort((a, b) -> a.length - b.length);

    for (int[] an : ans) {
      StringJoiner sj = new StringJoiner("+");
      for (int v : an) sj.add(v + "");
      System.out.println(t + "=" + sj);
    }

    System.out.println("Result:" + ans.size());
  }
}
```

JS算法源码

```javascript
/* JavaScript Node ACM模式 控制台输入获取 */
const readline = require("readline");

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

rl.on("line", (line) => {
  getResult(parseInt(line));
});

function getResult(t) {
  const arr = new Array(t).fill(0).map((_, i) => i + 1);

  const ans = [];

  let l = 0;
  let r = 1;

  let sum = arr[l];

  while (l < t) {
    if (sum > t) {
      sum -= arr[l++];
    } else if (sum == t) {
      ans.push(arr.slice(l, r));
      sum -= arr[l++];
      if (r >= t) break;
      sum += arr[r++];
    } else {
      sum += arr[r++];
    }
  }

  ans
    .sort((a, b) => a.length - b.length)
    .forEach((sub) => {
      console.log(`${t}=${sub.join("+")}`);
    });

  console.log(`Result:${ans.length}`);
}
```

Python算法源码

```python
# 输入获取
t = int(input())


# 算法入口
def getResult():
    arr = [i + 1 for i in range(t)]

    ans = []

    l = 0
    r = 1
    total = arr[l]

    while l < t:
        if total > t:
            total -= arr[l]
            l += 1
        elif total == t:
            ans.append(arr[l:r])
            total -= arr[l]
            l += 1
            if r >= t:
                break
            else:
                total += arr[r]
                r += 1
        else:
            total += arr[r]
            r += 1

    ans.sort(key=lambda x: len(x))

    for an in ans:
        print(f"{t}={'+'.join(map(str, an))}")

    print(f"Result:{len(ans)}")


# 算法调用
getResult()
```