**Queue**

Bài 1: Tạo một queue. Ưu tiên dãy số chẵn giảm dần đứng đầu hàng và dãy số lẻ giảm dần đứng ở cuối queue.

Bài 2: Sử dụng queue thực hiện hành động đảo ngược dữ liệu trong queue.

Bài 3: Tạo một queue chứa giá trị của dãy số nguyên được nhập từ bàn phím, nhiệm vụ của bạn đếm xem có bao cặp liền nhau bằng giá trị K cho trước. In ra số lượng cặp t/m.

**Examples**

**input**

10 3

5 2 6 9 7 2 4 7 10

**output**

4

**input**

6 2

1 3 1 3 1 3

**output**

5

**input**

8 4

1 1 1 1 1 1 1 1

**output**

0

**Bài 1:** Cho một chuỗi string s chỉ bao gồm các toán tử và toán hạng.

**Trong đó:**

- Toán tử chỉ bao gồm các phép: + -  \* và  /

- Toán hạng là các số nguyên  a ( 0<= a <=999).

hiệm vụ của bạn là tính giá trị của biểu thức đã cho.

**Examples**

**input**

2\*3-4\*5

**output**

-14

**input**

2\*9-5\*2+1-6

**output**

-3

**input**

20\*99-5+4\*999/70/50

**output**

1976

**Bài 2:** Bạn được cho một chuỗi s có chẵn lần độ dài để đảm bảo rằng số lượng dấu ngoặc đóng luôn bằng số lượng dấu ngoặc mở.

Trong một lần di chuyển, bạn có thể chọn **chính xác một dấu ngoặc** và di chuyển nó đến đầu hoặc cuối chuỗi.

**Trong đó:**

- chuỗi chỉ gồm ‘(‘ và ‘)’.

- nếu s là chuỗi ngoặc thường thì “(“ + s + “)” cũng là chuỗi thường.

- nếu s và t là chuỗi thường thì s + t cũng là chuỗi thường.

Ví dụ: “( ) ( )”, “( ( ) ) ( )”, “( ( ) )” và “ ( ) ” là chuỗi thường nhưng “ ) (“, “( ) (“ , “) ) ) ” thì không phải.

Bạn có cần trả lời T trường hợp.

**Input**

Dòng đầu tiên của input chứa 1 số nguyên t ( 1<= t <= 2000) \_\_\_ số trường hợp. Sau đó t dòng.

Dòng đầu tiên của từng trường hợp chứa 1 số nguyên n ( 2<= n <= 50 ) \_\_\_\_\_ độ dài của chuỗi S. Dòng thứ 2 là chuỗi S.

**Output**

Cho mỗi trường hợp, in ra câu trả lời là số nguyên nhỏ nhất cần thực hiện di chuyển để có được chuỗi thường từ chuỗi S đã cho. Đề bài đảm bảo sẽ luôn tồn tại câu trả lời.

**Example**

**input**

4

2

)(

4

()()

8

())()()(

10

)))((((())

**output**

1

0

1

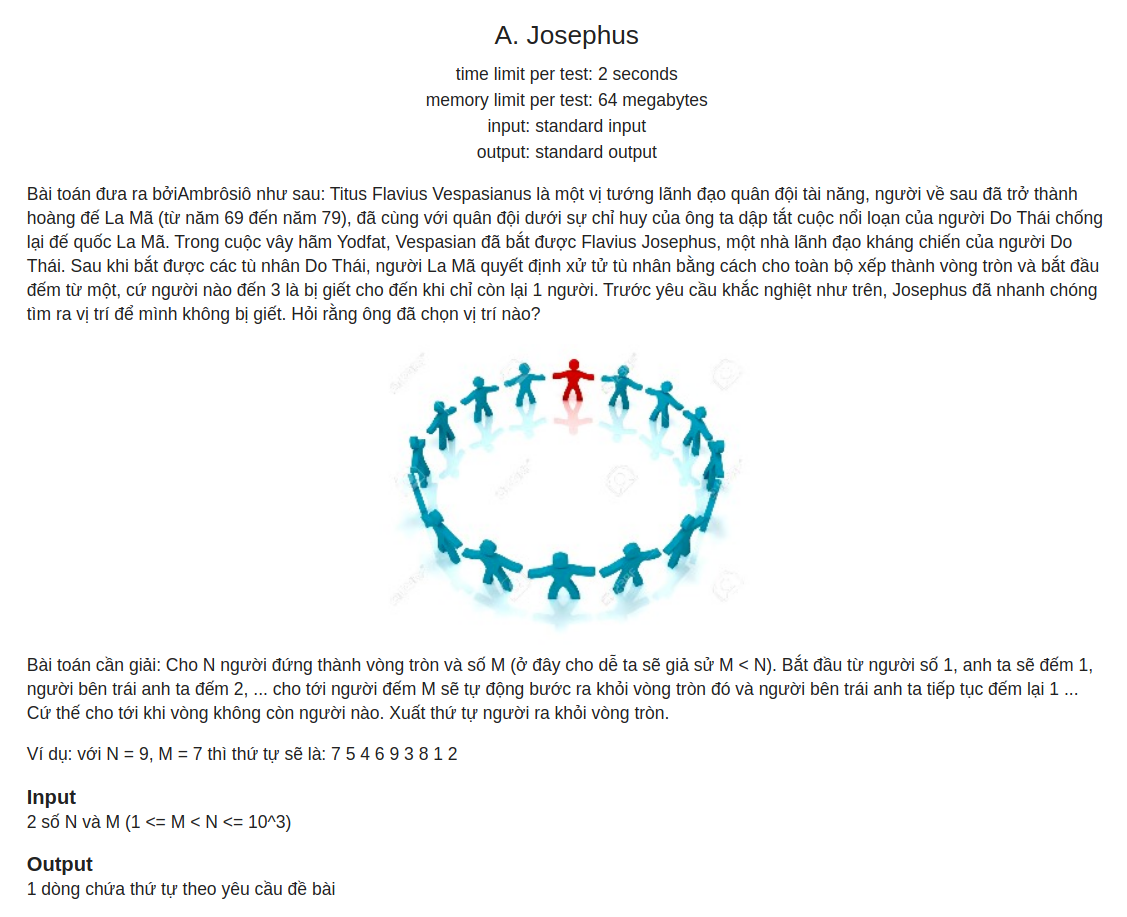
3

CHÚ Ý:

Trong trường hợp đầu tiên của ví dụ, cách tối ưu nhất là di chuyển dấu ngoặc đầu tiên xuống vị trí cuối cùng của chuỗi s.

Trong trường hợp thứ 3 của ví dụ, cách tối ưu nhất là di chuyển dấu ngoặc cuối chuỗi về vị trí đầu của chuỗi.

Trong trường hợp thứ 4 của ví dụ, chúng ta có thể chọn 3 hoặc động. Chuyển 3 dấu ngoặc để thu được chuỗi có kết quả như "((()))(())".



**Examples**

**input**

10 5

**output**

5 10 6 2 9 8 1 4 7 3

**input**

9 7

**output**

7 5 4 6 9 3 8 1 2