**LIST.**

**Bài 1:**  Lấy hai lists, ví dụ như sau:

**a = [1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89]**

**b = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13]**

Viết chương trình cho ra một list chỉ chứa những phần tử chung giữa các list đã cho (không được trùng nhau). Đảm bảo rằng chương trình có thể hoạt động trên hai lists có kích thước khác nhau.

**Bài 2:** Nhập vào 1 list chứa các số nguyên dương **a[i]**. Đề bài đảm bảo luôn tồn tại cặp **i, j** thỏa mãn điều kiện. **a[i] + a[j] = k.**

Nhiệm vụ của bạn là tìm tất các các cặp **i, j** sao cho thỏa mãn 2 điều kiện.

**→ I < j.**

**→ a[i] + a[j] = k**

**Input:**

Dòng đầu chứa hai số **n, k (2 <= n <= 1000)**. và **k( 0<= k <=10^9).**

Dòng tiếp theo bao gồm n số nguyên là các giá trị của danh sách.

**Output:**

Dòng đầu tiên in ra số lượng các cặp thỏa mãn.

Các dòng tiếp theo in ra 2 số i,j lần lượt là số lượng các cặp tương ứng.

**Example:**

|  |  |
| --- | --- |
| **Input :**  5 10  2 9 1 7 3 5 5 4 | **Output:**  3  2 3  4 5  6 7 |

**MATH**

**Bài 1:** Viết chương trình tính số tiền thực của một tài khoản ngân hàng dựa trên nhật ký giao dịch được nhập vào từ giao diện điều khiển.  Định dạng nhật ký được hiển thị như sau:

D 100 ( D là tiền gửi).   
W 200 ( W là tiền rút ra).

|  |  |
| --- | --- |
| **Input:**  D 300  D 300  W 200  D 100 | **Output:**  500 |

**Bài 2:** Dãy Fibonacci được tính dựa trên công thức sau:

f(n)=0 nếu n=0

f(n)=1 nếu n=1

f(n)=f(n-1)+f(n-2) nếu n>1

Hãy viết chương trình tính giá trị của f(n) với n là số được người dùng nhập vào. Ví dụ: Nếu n được nhập vào là 7 thì đầu ra của chương trình sẽ là 13.

**DICTIONARY**

**Bài 1:** Với số nguyên n nhất định, hãy viết chương trình để tạo ra một dictionary chứa (i, i\*i) như là số nguyên từ 1 đến n (bao gồm cả 1 và n) sau đó in ra dictionary này. Ví dụ: Giả sử số n là 8 thì đầu ra sẽ là: {1: 1, 2: 4, 3: 9, 4: 16, 5: 25, 6: 36, 7: 49, 8: 64}.

**Bài 2:** Định nghĩa một hàm có thể in dictionary chứa key là các số từ 1 đến 10 và values là các giá trị bình phương của chúng.

**Bài 3:** Cho dictionary **dict = {1:'a', 5:'h', 3:'x', 2:'l'}**

sắp xếp dict theo thứ tự giảm dần **của các phần tử trong dict.**

**Bài 4:** Cho dictionary **dict = {1:'a', 5:'h', 3:'x', 2:'l'}**

sắp xếp dict theo keys giảm dần và tăng dần.

**Bài 5:** Nhập vào một chuỗi mật khẩu với T là số lần nhập.

Sau đó T dòng mỗi dòng bao gồm 1 chuỗi string nhập từ bàn phím.

- Nếu Chuỗi mật khẩu chưa tồn tại hãy in ra “OKE”

- Nếu chuỗi mật khẩu đã tồn tại hãy đưa ra gợi ý ở cuối mật khẩu.

|  |  |
| --- | --- |
| **Input.**  **6**  iloveyou  iloveyou  ihateyou  abcd  abcd  abcd  abcd1  abcd12  abcd1234 | **Output.**  OKE  iloveyou1  OKE  OKE  abcd1  abcd2  abcd12  abcd123  OKE. |

**SET**

**Bài 1.**Cô giáo Thảo thống kê chiều cao của các học sinh trong lớp 12A được kết quả như sau:  
161 182 161 154 176 170 167 171 170 174 150 142 148 165 170 178 156 145 149 163 162 159 165 165 170 180 155 159 155 153 152 162 180 168 169 168 167 170  
a**.** Hỏi lớp có bao nhiêu học sinh?  
b**.** Tính chiều cao trung bình của các học sinh trong lớp.  
c**.**Liệt kê các chiều cao khác nhau của học sinh trong lớp. Tính giá trị trung bình của chúng.

**Bài 2.**Liệt kê tập hợp S gồm các kí tự của câu dưới đây, tính cả dấu cách. Viết chương trình in ra tập hợp S có bao nhiêu phần tử. Thêm vào tập S các phần tử là các kí tự xuất hiện trong câu sau: ***“Anh duy dep zai va học gioi ahihi”***

**Bài 3.**Cho tập hợp A gồm có các phần tử sau, mỗi phần tử cách nhau bởi dấu cách trắng.  
**A = “1 a 5 7 f 0 89 g I 11 88 3 p”.**  
Viết chương trình liệt kê các tập con có hai phần tử trở lên của tập hợp A. In ra tất cả những tập con như vậy.

**Tuple**

**Bài 1:** Viết một chương trình để tạo ra và in tuple chứa các số chẵn được lấy từ tuple (1,2,3,4,5,6,7,8,9,10).

**Bài 2:** Cho một dãy số nguyên từ 1 đến n. Sử dụng tuple hiển thị ra màn hình nửa dãy đầu tăng dần, nửa dãy sau giảm dần.

[**https://codeforces.com/gym/221491/problem/D**](https://codeforces.com/gym/221491/problem/D)

**D. Trông xe (Dict)**

time limit per test

4 seconds

Một bãi đỗ xe nhận trông xe trong vòng một tháng. Mỗi xe sẽ được gắn một số hiệu là một số nguyên dương T (10102010 ≤ T ≤ 10109999). Hai xe khác nhau sẽ được gắn hai số hiệu khác nhau. Một xe có thể ra vào bãi đỗ xe nhiều lần, mỗi lần vào bãi đỗ xe, người trông xe sẽ ghi vào sổ sách số hiệu của chiếc xe đó.



Cuối tháng dựa vào sổ ghi chép, người trông xe làm thống kê về số lần vào bãi đỗ xe của từng chiếc xe để tiến hành thu phí. Nếu một chiếc xe vào bãi đỗ xe p lần, cuối tháng chủ xe phải trả một lượng phí được tính như sau:

- Nếu p ≤ 5, chi phí là: 100. - Nếu p > 5, chi phí là: 100 + (p-5)

Yêu cầu: Tính tổng số phí người trông xe thu được vào cuối tháng.

Input

- Dòng đầu chứa một số nguyên dương K (0 < K ≤ 10^6)

- K dòng tiếp theo, mỗi dòng chứa số hiệu một chiếc xe.

Output

- Là tổng số phí thu được.

|  |  |
| --- | --- |
| **Example**  **input**  7  10102010  10108888  10102010  10102010  10102010  10102010  10102010 | **output**  201 |

**Valid Parentheses (Dict + string)**

Given a string s containing just the characters '(', ')', '{', '}', '[' and ']', determine if the input string is valid.

An input string is valid if:

1. Open brackets must be closed by the same type of brackets.
2. Open brackets must be closed in the correct order.

**Example 1:**

**Input:** s = "()"

**Output:** true

**Example 2:**

**Input:** s = "()[]{}"

**Output:** true

**Example 3:**

**Input:** s = "(]"

**Output:** false

**Example 4:**

**Input:** s = "([)]"

**Output:** false

**Example 5:**

**Input:** s = "{[]}"

**Output:** true

**Constraints:**

* 1 <= s.length <= 104
* s consists of parentheses only '()[]{}'.

**A. Generous NAM (DICT + STRING)**

time limit per test

2 seconds

Nam có một chuỗi bóng bay nhiều màu có n quả bóng. bóng thứ i có màu b[i].

Nam có k bạn bè. Bạn bè của Nam khó chịu khi 1 người nhận được 2 quả bóng cùng màu nhau. Nam muốn tất cả bạn bè của mình hạnh phúc. Nhưng không may Nam ko học giỏi toán. Hãy giúp Nam trả lời câu hỏi có thể làm tất cả bạn bè của anh ấy hạnh phúc hay không.

Có hãy ra “YES” ngược lại in ra “NO”.

**Input:**

Dòng đầu tiên chứa 2 số nguyên n và k với (1<= n,k <= 100). Số lượng quả bóng và số lượng bạn bè của Nam.

Dòng tiếp theo chứa một chuỗi string s - màu của những quả bóng Nam có.

**Output:**

In ra “YES” or “NO” trên 1 dòng.

**Examples**

**input**

4 2

aabb

**output**

YES

**input**

6 3

aacaab

**output**

NO

**Note**

Trong ví dụ đầu tiên Nam có thể cho quả bóng thứ 1 và 3 cho người bạn đầu tiên và quả bóng 2,4 cho người bạn thứ 2. In ra “YES”.

Trong ví dụ thứ 2 bạn không thể chia được quả bóng thỏa mãn điều kiện in ra “NO”