Formatting Submissions for a USENIX Conference: An (Incomplete) Example

XƎLaTeX

Ruohong Jiao
First Institution

Second Name
Second Institution

Third Name
Third Institution

Fourth Name
Fourth Institution

Fifth Name
Fifth Institution

2020-03-28

摘要

Your abstract text goes here. Just a few facts. Whet our appetites. Not more than 200 words, if possible, and preferably closer to 150.

1 引言

Some of the **greatest 排版引擎**
discoveries in science 排版引擎
were made by ***accident 排版引擎***.

XƎTEX 是一种使用 Unicode 的 TEX 排版引擎，并支持一些现代字体技术，例如 OpenType、Graphite 和 Apple Advanced Typography（AAT），可以直接利用其高级特性，例如额外的字形，花体，合字，可变的文本粗细等等。

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

2 关于数学部分

数学、中英文皆可以混排。You can intersperse math, Chinese and English (Latin script) without adding extra environments.

3 Footnotes, Verbatim, and Citations

Footnotes should be places after punctuation characters, without any spaces between said characters and footnotes, like so.[1] And some embedded literal code may look as follows.

```
int main(int argc, char *argv[])
{
    return 0;
}
```

Now we're going to cite somebody. Watch for the cite tag. Here it comes. Arpachi-Dusseau and Arpachi-Dusseau co-authored an excellent OS book, which is also really funny [1], and Waldspurger got into the SIGOPS hall-of-fame due to his seminal paper about resource management in the ESX hypervisor [2].

The tilde character (˜) in the tex source means a non-breaking space. This way, your reference will always be attached to the word that preceded it, instead of going to the next line.

And the 'cite' package sorts your citations by their numerical order of the corresponding references at the end of the paper, ridding you from the need to notice that, e.g, "Waldspurger" appears after "Arpachi-Dusseau" when sorting references alphabetically [1, 2].

It'd be nice and thoughtful of you to include a suitable link in each and every bibtex entry that you use in your submission, to allow reviewers (and other readers) to easily get to the cited work, as is done in all entries found in the References section of this document.

Now we're going take a look at Section 4, but not before observing that refs to sections and citations and such are colored and clickable in the PDF because of the packages we've included.

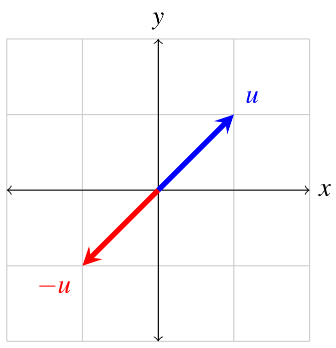
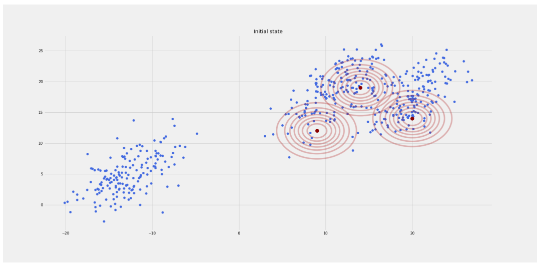
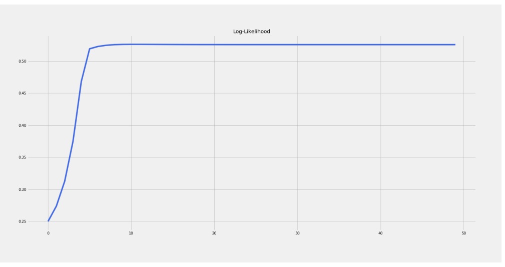


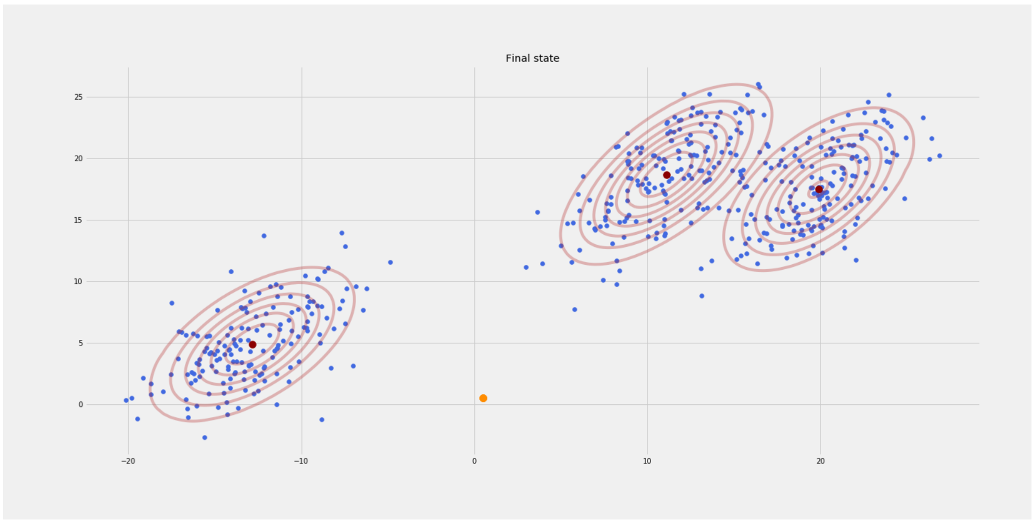
图 1: Text size inside figure should be as big as caption's text. Text size inside figure should be as big as caption's text. Text size inside figure should be as big as caption's text. Text size inside figure should be as big as caption's text. Text size inside figure should be as big as caption's text.



(a) GMM init



(b) GMM likehood



(c) GMM ans

图 2: GMM

4 Floating Figures and Lists

Here's a typical reference to a floating figure: Figure 1. Floats should usually be placed where latex wants then. Figure1 is centered, and has a caption that instructs you to make sure that the size of the text within the figures that you use is as big as (or bigger than) the size of the text in the caption of the figures. Please do. Really.

In our case, we've explicitly drawn the figure inlined in latex, to allow this tex file to cleanly compile. But usually, your figures will reside in some file.pdf, and you'd include them in your document with, say, \includegraphics.

Lists are sometimes quite handy. If you want to itemize things, feel free:

fread a function that reads from a `stream` into the array `ptr` at most `nobj` objects of size `size`, returning returns the number of objects read.

Fred a person's name, e.g., there once was a dude named Fred who separated usenix.sty from this file to allow for easy inclusion.

The noindent at the start of this paragraph in its tex version makes it clear that it's a continuation of the preceding paragraph, as opposed to a new paragraph in its own right.

4.1 LaTeX-ing Your TeX File

People often use `pdflatex` these days for creating pdf-s from tex files via the shell. And `bibtex`, of course. Works for us.

Acknowledgments

The USENIX latex style is old and very tired, which is why there's no \acks command for you to use when acknowledging. Sorry.

Availability

USENIX program committees give extra points to submissions that are backed by artifacts that are publicly available. If you made your code or data available, it's worth mentioning this fact in a dedicated section.

参考文献

[1] Remzi H. Arpaci-Dusseau and Arpaci-Dusseau Andrea C. *Operating Systems: Three Easy Pieces*. Arpaci-Dusseau Books, LLC, 1.00 edition, 2015. `http://pages.cs.wisc.edu/~remzi/OSTEP/`.

[2] Carl A. Waldspurger. Memory resource management in VMware ESX server. In *USENIX Symposium on Operating System Design and Implementation (OSDI)*, pages 181–194, 2002. `https://www.usenix.org/legacy/event/osdi02/tech/waldspurger/waldspurger.pdf`.

附录 A Some Codes

some text

[1]Remember that USENIX format stopped using endnotes and is now using regular footnotes.

附录 B　Codes

```python
#%%
import numpy as np
import scipy.stats as st

import matplotlib.pyplot as plt
import seaborn as sns

sns.set_palette("Paired")
```

```python
import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None #to become the incidence matrix
    VT = np.zeros((n*m,1), int)  #dummy variable

    #compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m-1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

                if M is None:
                    M = np.copy(VT)
                else:
                    M = np.concatenate((M, VT), 1)

                VT = np.zeros((n*m,1), int)

    return M
```

附录 C　Problem List

C.1　Leetcode 1

Problem Description:

两数之和

给定一个整数数组 *nums* 和一个目标值 *target*，请你在该数组中找出和为目标值的那两个整数，并返回他们的数组下标。

你可以假设每种输入只会对应一个答案。但是，数组中同一个元素不能使用两遍。

Sample:

input:

```
给定 nums = [2, 7, 11, 15], target = 9
```

otput:

```
因为 nums[0] + nums[1] = 2 + 7 = 9
所以返回 [0, 1]
```

Solution (Codes at D.1):

没有说明输入数字一定是正整数，不能先排序后提取小于 *target* 的数进行求解。

懒一点，n^2 循环。勤快一点用红黑树、堆进行存储然后查询 $nlogn$ 。

C.2 Leetcode 2

Problem Description:

两数相加

给出两个非空的链表用来表示两个非负的整数。其中，它们各自的位数是按照逆序的方式存储的，并且它们的每个节点只能存储一位数字。

如果，我们将这两个数相加起来，则会返回一个新的链表来表示它们的和。

您可以假设除了数字 0 之外，这两个数都不会以 0 开头。

Sample:

input:

```
输入：(2 -> 4 -> 3) + (5 -> 6 -> 4)
```

otput:

```
输出：7 -> 0 -> 8
原因：342 + 465 = 807
```

Solution (Codes at D.2):

大数加法，注意指针/引用转换。

C.3 Leetcode 19

Problem Description:

删除链表的倒数第 N 个节点

给定一个链表，删除链表的倒数第 n 个节点，并且返回链表的头结点。

Sample:

input:

```
给定一个链表：1->2->3->4->5，和 n = 2.
```

otput:

```
当删除了倒数第二个节点后，链表变为 1->2->3->5.
```

Solution (Codes at D.3):

递归记录个数。

C.4 Leetcode 25

Problem Description:

K 个一组翻转链表

给你一个链表，每 k 个节点一组进行翻转，请你返回翻转后的链表。

k 是一个正整数，它的值小于或等于链表的长度。

如果节点总数不是 k 的整数倍，那么请将最后剩余的节点保持原有顺序。

说明：

你的算法只能使用常数的额外空间。

你不能只是单纯的改变节点内部的值，而是需要实际进行节点交换。

Sample:

input:

```
给你这个链表：1->2->3->4->5
```

otput:

```
当 k = 2 时, 应当返回: 2->1->4->3->5

当 k = 3 时, 应当返回: 3->2->1->4->5
```

Solution (Codes at D.4):
标记子链表首尾，化简为链表反转问题，递归解决。

C.5 Leetcode 56

Problem Description:
合并区间
给出一个区间的集合，请合并所有重叠的区间。。
Sample:
input:

```
输入: intervals = [[1,3],[2,6],[8,10],[15,18]]

输入: intervals = [[1,4],[4,5]]
```

otput:

```
输出: [[1,6],[8,10],[15,18]]
解释: 区间 [1,3] 和 [2,6] 重叠, 将它们合并为 [1,6].

输出: [[1,5]]
解释: 区间 [1,4] 和 [4,5] 可被视为重叠区间。
```

Solution (Codes at D.5):
自定义排序之后合并。需要注意对是 C++ 快排默认比较顺序是挨个从小到大，如果可以使用原始比较函数就不要自己重新写，还是用初始的更快。
注意 java 的自定义比较函数。

C.6 Leetcode 61

Problem Description:
旋转链表
给定一个链表，旋转链表，将链表每个节点向右移动 k 个位置，其中 k 是非负数。
Sample:
input:

```
输入: 1->2->3->4->5->NULL, k = 2

输入: 0->1->2->NULL, k = 4
```

otput:

```
输出: 4->5->1->2->3->NULL
解释:
向右旋转 1 步: 5->1->2->3->4->NULL
向右旋转 2 步: 4->5->1->2->3->NULL

输出: 2->0->1->NULL
解释:
向右旋转 1 步: 2->0->1->NULL
向右旋转 2 步: 1->2->0->NULL
向右旋转 3 步: 0->1->2->NULL
向右旋转 4 步: 2->0->1->NULL
```

Solution (Codes at D.6):
对 k 取模得到 mk，之后更改倒数第 mk 个 node 开始第子链表顺序到首位。

C.7 Leetcode 138

Problem Description:

复制带随机指针的链表

给定一个链表，每个节点包含一个额外增加的随机指针，该指针可以指向链表中的任何节点或空节点。要求返回这个链表的深拷贝。

我们用一个由 n 个节点组成的链表来表示输入/输出中的链表。每个节点用一个 $[val, random_index]$ 表示：

val：一个表示 Node.val 的整数。

$random_index$：随机指针指向的节点索引（范围从 0 到 n-1）；如果不指向任何节点，则为 null 。

$-10000 <= Node.val <= 10000$

$Node.random$ 为空（null）或指向链表中的节点。

节点数目不超过 1000。

Sample:

input:

```
输入: head = [[7,null],[13,0],[11,4],[10,2],[1,0]]

输入: head = [[1,1],[2,1]]
输入: head = [[3,null],[3,0],[3,null]]
输入: head = []
```

otput:

```
输出: [[7,null],[13,0],[11,4],[10,2],[1,0]]
输出: [[1,1],[2,1]]
输出: [[3,null],[3,0],[3,null]]
输出: []
解释: 给定的链表为空（空指针），因此返回 null。
```

Solution (Codes at D.7):

map 存储节点 pair。

C.8 Leetcode 206

Problem Description:

反转链表

反转一个单链表。

Sample:

input:

```
输入: 1->2->3->4->5->NULL
```

otput:

```
输出: 5->4->3->2->1->NULL
```

Solution (Codes at D.8):

递归或先找到首尾节点之后 while 循环更新 next。

C.9 Leetcode 387

Problem Description:

字符串中的第一个唯一字符

给定一个字符串，找到它的第一个不重复的字符，并返回它的索引。如果不存在，则返回 -1 。

Sample:

input:

```
leetcode
loveleetcode
```

otput:

```
0
2
```

Solution (Codes at D.9):
找到字符出现的首位和末位进行判断，然后取最早出现的。

附录 D　Code List

D.1　Leetcode 1

C++

```cpp
#include <algorithm>
#include <cstdio>
#include <cstdlib>
#include <cstring>
#include <iostream>
#include <queue>
#include <stack>
#include <string>
#include <vector>

using namespace std;

class Solution {
 public:
  vector<int> twoSum(vector<int>& nums, int target) {
    vector<int> ans;
    for (int i = 0; i < nums.size(); i++)
      for (int j = i + 1; j < nums.size(); j++)
        if (nums[i] + nums[j] == target) {
          // u = i;
          // v = j;
          ans.push_back(i);
          ans.push_back(j);
          break;
        }
    return ans;
  }
  void input(void) {
    while (~scanf("%d %d", &n, &m))
      for (int i = 0; i < n; i++) {
        scanf("%d", &t);
        numbers.push_back(t);
      }
  }
  void solve(void) { twoSum(numbers, m); }
  void otput(void) { printf("%d %d\n", u, v); }

 private:
  int n, m, t;
  int u, v;
  vector<int> numbers;
};

int main() {
  freopen("./assets/fipt.txt", "r", stdin);
  freopen("./assets/fopt.txt", "w", stdout);

  Solution sol;

  sol.input();
  sol.solve();
  sol.otput();

  return 0;
}
```

Java

```java
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.PrintStream;
import java.util.Scanner;

class Solution {
  public int[] twoSum(int[] nums, int target) {
    int[] ans = new int[2];
    for (int i = 0; i < nums.length; i++)
      for (int j = i + 1; j < nums.length; j++)
        if (nums[i] + nums[j] == target) {
          u = i;
          v = j;
          ans[0] = i;
          ans[1] = j;
          break;
        }
    return ans;
  }

  public static void main(String[] args) throws FileNotFoundException {
    FileInputStream fin = new FileInputStream("./assets/fipt.txt");
    PrintStream fot = new PrintStream(new FileOutputStream("./assets/fopt.txt"));

    System.setIn(fin);
    System.setOut(fot);

    Solution sol = new Solution();

    sol.input();
    sol.solve();
    sol.otput();
  }

  public void input() {
    Scanner in = new Scanner(System.in);
    while (in.hasNext()) {
      n = in.nextInt();
      m = in.nextInt();
      for (int i = 0; i < n; i++)
        numbers[i] = in.nextInt();
    }

    in.close();
  }

  public void solve() {
    twoSum(numbers, m);
  }

  public void otput() {
    System.out.println(u + " " + v);
  }

  private int n, m, t;
  private int u, v;
  private int[] numbers = new int[10000];
}
```

D.2 Leetcode 2

C++

```cpp
#include <algorithm>
#include <cstdio>
#include <cstdlib>
#include <cstring>
#include <iostream>
#include <map>
#include <queue>
#include <stack>
#include <string>
#include <vector>

using namespace std;

typedef struct ListNode {
  int val;
  ListNode* next;
  ListNode(int x) : val(x), next(NULL) {}
} ListNode;

class Solution {
 public:
  ListNode* addTwoNumbers(ListNode* l1, ListNode* l2) {
    ListNode* res = new ListNode(0);
    ListNode* u = l1;
    ListNode* v = l2;
    ListNode* cur = NULL;
    int t = 0;

    while (u != NULL || v != NULL || t != 0) {
      if (cur == NULL) {
        cur = res;
      } else {
        cur->next = new ListNode(0);
        cur = cur->next;
      }
      cur->next = NULL;

      cur->val = t;
      if (u != NULL) {
        cur->val += u->val;
        u = u->next;
      }
      if (v != NULL) {
        cur->val += v->val;
        v = v->next;
      }
      t = cur->val / 10;
      cur->val %= 10;
    }

    return res;
  }
  void input(void) {
    while (~scanf("%d %d", &n, &m)) {
      a = new ListNode(0);
      b = new ListNode(0);
      ListNode* u = a;
      ListNode* v = b;

      for (int i = 0; i < n; i++) {
        u->next = new ListNode(0);
        u = u->next;
        u->next = NULL;

        scanf("%d", &u->val);
      }
      for (int i = 0; i < m; i++) {
        v->next = new ListNode(0);
        v = v->next;
        v->next = NULL;

        scanf("%d", &v->val);
      }

      solve(a, b);
    }
  }
  void show(ListNode* x) {
    ListNode* cur = x;
    while (cur != NULL) {
      printf("%d", cur->val);
      cur = cur->next;
    }
  }
  void solve(ListNode* x, ListNode* y) {
    ListNode* res;
    res = addTwoNumbers(x, y);
    otput(res);
  }
  void otput(ListNode* x) {
    ListNode* cur = x->next;
    while (cur != NULL) {
      printf("%d", cur->val);
      cur = cur->next;
    }
    cout << endl;
  }

 private:
  int n, m;
  ListNode *a, *b;
};

int main() {
  freopen("./assets/fipt.txt", "r", stdin);
  freopen("./assets/fopt.txt", "w", stdout);

  Solution sol;

  sol.input();

  return 0;
}
```

Java

```java
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.PrintStream;
import java.util.Scanner;

class Solution {
  public class ListNode {
    int val;
    ListNode next;

    ListNode(int x) {
      val = x;
    }
  }

  public ListNode addTwoNumbers(ListNode l1, ListNode l2) {
    ListNode res = new ListNode(0);
    ListNode u = l1;
    ListNode v = l2;
    ListNode cur = null;
    int t = 0;

    while (u != null || v != null || t != 0) {
      if (cur == null) {
        cur = res;
      } else {
        cur.next = new ListNode(0);
        cur = cur.next;
      }
      cur.next = null;

      cur.val = t;
      if (u != null) {
        cur.val += u.val;
        u = u.next;
      }
      if (v != null) {
        cur.val += v.val;
        v = v.next;
      }
      t = cur.val / 10;
      cur.val %= 10;
    }

    return res;
  }

  public static void main(String[] args) throws FileNotFoundException {
    FileInputStream fin = new FileInputStream("./assets/fipt.txt");
    PrintStream fot = new PrintStream(new FileOutputStream("./assets/fopt.txt"));

    System.setIn(fin);
    System.setOut(fot);

    Solution sol = new Solution();

    sol.input();
  }

  public void input() {
    Scanner in = new Scanner(System.in);
    while (in.hasNext()) {
      n = in.nextInt();
      m = in.nextInt();

      ListNode a = new ListNode(0);
      ListNode b = new ListNode(0);
      ListNode u = a;
      ListNode v = b;

      for (int i = 0; i < n; i++) {
        if (i != 0) {
          u.next = new ListNode(0);
          u = u.next;
        }
        u.val = in.nextInt();
      }
      for (int i = 0; i < m; i++) {
        if (i != 0) {
          v.next = new ListNode(0);
          v = v.next;
        }
        v.val = in.nextInt();
      }

      solve(a, b);
    }

    in.close();
  }

  public void solve(ListNode x, ListNode y) {
    ListNode res;
    res = addTwoNumbers(x, y);
    otput(res);
  }

  public void otput(ListNode x) {
    while (x != null) {
      System.out.print(x.val);
      x = x.next;
    }
    System.out.println("");
  }

  private int n, m;
}
```

D.3 Leetcode 19

C++

```
#include <algorithm>
#include <cstdio>
#include <cstdlib>
#include <cstring>
#include <iostream>
#include <map>
#include <queue>
#include <stack>
#include <string>
#include <vector>

using namespace std;

typedef struct ListNode {
  int val;
  ListNode* next;
  ListNode(int x) : val(x), next(NULL) {}
} ListNode;

class Solution {
 public:
  int deleteNode(ListNode* x, int target) {
    if (x == NULL) return 0;

    int u = deleteNode(x->next, target);

    if (u != -1) {
      if (u == target) {
        ListNode* y = x->next->next;
        delete x->next;
        x->next = y;
      } else {
        return u + 1;
      }
    }

    return -1;
  }
  ListNode* removeNthFromEnd(ListNode* head, int n) {
    int u = deleteNode(head, n);
    if (u != -1) {
      ListNode* t = head;
      head = head->next;
      delete t;
    }

    return head;
  }

  void input(void) {
    while (~scanf("%d %d", &n, &m)) {
      a = new ListNode(0);
      ListNode* u = a;

      for (int i = 0; i < m; i++) {
        if (i != 0) {
          u->next = new ListNode(0);
          u = u->next;
        }

        scanf("%d", &u->val);
      }

      solve(a, n);
    }
  }
  void solve(ListNode* x, int y) {
    ListNode* res;
    res = removeNthFromEnd(x, y);
    otput(res);
  }
  void otput(ListNode* x) {
    ListNode* cur = x;
    while (cur != NULL) {
      printf("%d", cur->val);
      cur = cur->next;
    }
  }

 private:
  int n, m;
  ListNode* a;
};

int main() {
  freopen("./assets/fipt.txt", "r", stdin);
  freopen("./assets/fopt.txt", "w", stdout);

  Solution sol;

  sol.input();

  return 0;
}
```

Java

```
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.PrintStream;
import java.util.Scanner;

class Solution {
  public class ListNode {
    int val;
    ListNode next;

    ListNode(int x) {
      val = x;
```

```java
    }
  }

  public int deleteNode(ListNode x, int target) {
    if (x == null)
      return 0;

    int u = deleteNode(x.next, target);

    if (u != -1) {
      if (u == target) {
        ListNode y = x.next.next;
        x.next = y;
      } else {
        return u + 1;
      }
    }

    return -1;
  }

  public ListNode removeNthFromEnd(ListNode head, int n) {
    int u = deleteNode(head, n);
    if (u != -1) {
      head = head.next;
    }
    return head;
  }

  public static void main(String[] args) throws FileNotFoundException {
    FileInputStream fin = new FileInputStream("./assets/fipt.txt");
    PrintStream fot = new PrintStream(new FileOutputStream("./assets/fopt.txt"));

    System.setIn(fin);
    System.setOut(fot);

    Solution sol = new Solution();

    sol.input();
  }

  public void input() {
    Scanner in = new Scanner(System.in);
    while (in.hasNext()) {
      n = in.nextInt();
      m = in.nextInt();

      ListNode a = new ListNode(0);
      ListNode u = a;

      for (int i = 0; i < m; i++) {
        if (i != 0) {
          u.next = new ListNode(0);
          u = u.next;
        }
        u.val = in.nextInt();
      }

      solve(a, n);
    }

    in.close();
  }

  public void solve(ListNode x, int y) {
    ListNode res;
    res = removeNthFromEnd(x, y);
    otput(res);
  }

  public void otput(ListNode x) {
    while (x != null) {
      System.out.print(x.val);
      x = x.next;
    }
    System.out.println("");
  }

  private int n, m;
}
```

D.4 Leetcode 25

C++

```
#include <algorithm>
#include <cstdio>
#include <cstdlib>
#include <cstring>
#include <iostream>
#include <map>
#include <queue>
#include <stack>
#include <string>
#include <vector>

using namespace std;

typedef struct ListNode {
  int val;
  ListNode* next;
  ListNode(int x) : val(x), next(NULL) {}
} ListNode;

class Solution {
 public:
  void reverseGroup(ListNode* u, ListNode* v) {
    if (u != v) {
      if (u->next == v) {
        v->next = u;
      } else {
        ListNode* t = u->next;
        reverseGroup(u->next, v);
        t->next = u;
      }
    }
  }
  ListNode* reverseKGroup(ListNode* head, int k) {
    if (k == 1) return head;

    ListNode* fakeHead = new ListNode(0);
    fakeHead->next = head;
    ListNode* pre = fakeHead;
    ListNode* cur = head;
    ListNode* u = NULL;
    ListNode* t = NULL;
    ListNode* v = NULL;
    int cnt = 0;
    while (cur != NULL) {
      cnt++;
      if (cnt == k) {
        u = pre->next;
        v = cur;
        t = cur->next;
        reverseGroup(u, v);
        pre->next = v;
```

```cpp
        u->next = t;
        pre = u;
        cur = u;
        cnt = 0;
      }
      cur = cur->next;
    }
    return fakeHead->next;
  }
  void input(void) {
    while (~scanf("%d %d", &n, &m)) {
      a = new ListNode(0);
      ListNode* u = a;

      for (int i = 0; i < m; i++) {
        if (i != 0) {
          u->next = new ListNode(0);
          u = u->next;
        }

        scanf("%d", &u->val);
      }

      solve(a, n);
    }
  }
  void solve(ListNode* x, int y) {
    ListNode* res;
    res = reverseKGroup(x, y);
    otput(res);
  }
  void otput(ListNode* x) {
    ListNode* cur = x;
    while (cur != NULL) {
      printf("%d", cur->val);
      cur = cur->next;
    }
    cout << endl;
  }

 private:
  int n, m;
  ListNode* a;
};

int main() {
  freopen("./assets/fipt.txt", "r", stdin);
  freopen("./assets/fopt.txt", "w", stdout);

  Solution sol;

  sol.input();

  return 0;
}
```

Java

```java
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.PrintStream;
import java.util.Scanner;

class Solution {
  public class ListNode {
    int val;
    ListNode next;

    ListNode(int x) {
      val = x;
    }
  }

  void reverseGroup(ListNode u, ListNode v) {
    if (u != v) {
      if (u.next == v) {
        v.next = u;
      } else {
        ListNode t = u.next;
        reverseGroup(u.next, v);
        t.next = u;
      }
    }
  }

  public ListNode reverseKGroup(ListNode head, int k) {
    if (k == 1)
      return head;
    ListNode fakeHead = new ListNode(0);
    fakeHead.next = head;
    ListNode pre = fakeHead;
    ListNode cur = head;
    ListNode u = null;
    ListNode t = null;
    ListNode v = null;
    int cnt = 0;
    while (cur != null) {
      cnt++;
      if (cnt == k) {
        u = pre.next;
        v = cur;
        t = cur.next;
        reverseGroup(u, v);
        pre.next = v;
        u.next = t;
        pre = u;
        cur = u;
        cnt = 0;
      }
      cur = cur.next;
    }
    return fakeHead.next;
  }

  public static void main(String[] args) throws FileNotFoundException {
    FileInputStream fin = new FileInputStream("./assets/fipt.txt");
    PrintStream fot = new PrintStream(new FileOutputStream("./assets/fopt.txt"));

    System.setIn(fin);
    System.setOut(fot);

    Solution sol = new Solution();

    sol.input();
  }

  public void input() {
    Scanner in = new Scanner(System.in);
    while (in.hasNext()) {
      n = in.nextInt();
      m = in.nextInt();

      ListNode a = new ListNode(0);
      ListNode u = a;

      for (int i = 0; i < m; i++) {
        if (i != 0) {
          u.next = new ListNode(0);
          u = u.next;
        }
        u.val = in.nextInt();
      }

      solve(a, n);
    }

    in.close();
  }

  public void solve(ListNode x, int y) {
    ListNode res;
    res = reverseKGroup(x, y);
    otput(res);
  }

  public void otput(ListNode x) {
    while (x != null) {
      System.out.print(x.val);
      x = x.next;
    }
    System.out.println("");
  }

  private int n, m;
}
```

D.5 Leetcode 56

C++

```cpp
#include <algorithm>
#include <cstdio>
#include <cstdlib>
#include <cstring>
#include <iostream>
#include <map>
#include <queue>
#include <stack>
#include <string>
#include <vector>

using namespace std;

class Solution {
 public:

  vector<vector<int> > merge(vector<vector<int> >& intervals) {
    vector<vector<int> > res;
    sort(intervals.begin(), intervals.end());

    for (int i = 0; i < intervals.size(); i++) {
      if (res.empty()) {
        res.push_back(intervals[i]);
      } else {
        if (res.back()[1] < intervals[i][0]) {
          res.push_back(intervals[i]);
        } else {
          if (res.back()[1] < intervals[i][1]) res.back()[1] = intervals[i][1];
        }
      }
    }

    return res;
  }
  void input(void) {
    while (~scanf("%d", &n)) {
      int u, v;
      for (int i = 0; i < n; i++) {
        vector<int> t;
        scanf("%d %d", &u, &v);

        t.push_back(u);
        t.push_back(v);
        a.push_back(t);
      }

      solve(a);
      a.clear();
    }
  }
  void solve(vector<vector<int> >& x) {
    vector<vector<int> > res;
    res = merge(x);
    otput(res);
  }
  void otput(vector<vector<int> >& x) {
    vector<vector<int> > cur = x;
    for (int i = 0; i < cur.size(); i++)
      printf("%d %d\n", cur[i][0], cur[i][1]);
  }

 private:
  int n;
  vector<vector<int> > a;
};

int main() {
  freopen("./assets/fipt.txt", "r", stdin);
  freopen("./assets/fopt.txt", "w", stdout);

  Solution sol;

  sol.input();

  return 0;
}
```

Java

```
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.PrintStream;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.Comparator;
import java.util.Scanner;

class Solution {
  public int[][] merge(int[][] intervals) {
    ArrayList<int[]> res = new ArrayList<int[]>();
    Arrays.sort(intervals, new Comparator<int[]>() {
      @Override
      public int compare(int[] l, int[] r) {
        return l[0] - r[0];
      }
    });
    int sz = intervals.length;
    for (int i = 0; i < sz; i++) {
      int l = intervals[i][0];
      int r = intervals[i][1];
      while (i < sz - 1 && intervals[i + 1][0] <= r) {
        r = Math.max(r, intervals[i + 1][1]);
```

```java
        i++;
      }
      res.add(new int[] { l, r });
    }

    return res.toArray(new int[res.size()][2]);
  }

  public static void main(String[] args) throws FileNotFoundException {
    FileInputStream fin = new FileInputStream("./assets/fipt.txt");
    PrintStream fot = new PrintStream(new FileOutputStream("./assets/fopt.txt"));

    System.setIn(fin);
    System.setOut(fot);

    Solution sol = new Solution();

    sol.input();
  }

  public void input() {
    Scanner in = new Scanner(System.in);
    while (in.hasNext()) {
      n = in.nextInt();

      vals = new int[n][2];

      for (int i = 0; i < n; i++) {
        vals[i][0] = in.nextInt();
        vals[i][1] = in.nextInt();
      }

      solve(vals);
    }

    in.close();
  }

  public void solve(int[][] x) {
    int[][] res;
    res = merge(x);
    otput(res);

  }

  public void otput(int[][] x) {
    int sz = x.length;
    for (int i = 0; i < sz; i++)
      System.out.println(x[i][0] + " " + x[i][1]);
  }

  private int n;
  private int[][] vals;
}
```

D.6 Leetcode 61

C++

```cpp
#include <algorithm>
#include <cstdio>
#include <cstdlib>
#include <cstring>
#include <iostream>
#include <map>
#include <queue>
#include <stack>
```

```
#include <string>
#include <vector>

using namespace std;

typedef struct ListNode {
  int val;
  ListNode* next;
  ListNode(int x) : val(x), next(NULL) {}
} ListNode;

class Solution {
 public:
  ListNode* rotateRight(ListNode* head, int k) {
    if (head == NULL) return head;
    ListNode* res = head;
    ListNode* cur = head;
    ListNode* pre = NULL;
    ListNode* st = NULL;
    ListNode* en = NULL;
    vector<ListNode*> v;
    while (cur != NULL) {
      v.push_back(cur);

      if (cur->next == NULL) en = cur;
      cur = cur->next;
    }

    int lenOfList = v.size();
    int mk = k % lenOfList;
    if (mk != 0) {
      pre = v[lenOfList - mk - 1];
      st = v[lenOfList - mk];
      pre->next = NULL;
      en->next = res;
      res = st;
    }

    return res;
  }

  void input(void) {
    while (~scanf("%d %d", &n, &m)) {
      a = new ListNode(0);
      ListNode* u = a;

      for (int i = 0; i < m; i++) {
        if (i != 0) {
          u->next = new ListNode(0);
          u = u->next;
        }

        scanf("%d", &u->val);
      }

      solve(a, n);
    }
  }
  void solve(ListNode* x, int y) {
    ListNode* res;
    res = rotateRight(x, y);
    otput(res);
  }
  void otput(ListNode* x) {
    ListNode* cur = x;
    while (cur != NULL) {
      printf("%d", cur->val);
      cur = cur->next;
    }
    cout << endl;
  }

 private:
  int n, m;
  ListNode* a;
};

int main() {
  freopen("./assets/fipt.txt", "r", stdin);
  freopen("./assets/fopt.txt", "w", stdout);

  Solution sol;

  sol.input();

  return 0;
}
```

Java

```
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.PrintStream;
import java.util.LinkedList;
import java.util.Scanner;

class Solution {
  public class ListNode {
    int val;
    ListNode next;

    ListNode(int x) {
      val = x;
    }
  }

  public ListNode rotateRight(ListNode head, int k) {
    if (head == null)
      return head;
    ListNode res = head;
    ListNode cur = head;
    ListNode pre = null;
    ListNode st = null;
    ListNode en = null;
    LinkedList<ListNode> v = new LinkedList<ListNode>();
    while (cur != null) {
      v.add(cur);

      if (cur.next == null)
        en = cur;
      cur = cur.next;
    }

    int lenOfList = v.size();
    int mk = k % lenOfList;
    if (mk != 0) {
      pre = v.get(lenOfList - mk - 1);
      st = v.get(lenOfList - mk);
      pre.next = null;
      en.next = res;
      res = st;
    }

    return res;
  }

  public static void main(String[] args) throws FileNotFoundException {
```

```java
    FileInputStream fin = new FileInputStream("./assets/fipt.txt");
    PrintStream fot = new PrintStream(new FileOutputStream("./assets/fopt.txt"));

    System.setIn(fin);
    System.setOut(fot);

    Solution sol = new Solution();

    sol.input();
  }

  public void input() {
    Scanner in = new Scanner(System.in);
    while (in.hasNext()) {
      n = in.nextInt();
      m = in.nextInt();

      ListNode a = new ListNode(0);
      ListNode u = a;

      for (int i = 0; i < m; i++) {
        if (i != 0) {
          u.next = new ListNode(0);
          u = u.next;
        }
        u.val = in.nextInt();
      }

      solve(a, n);
    }

    in.close();
  }

  public void solve(ListNode x, int y) {
    ListNode res;
    res = rotateRight(x, y);
    otput(res);
  }

  public void otput(ListNode x) {
    while (x != null) {
      System.out.print(x.val);
      x = x.next;
    }
    System.out.println("");
  }

  private int n, m;
}
```

D.7 Leetcode 138

C++

```cpp
#include <algorithm>
#include <cstdio>
#include <cstdlib>
#include <cstring>
#include <iostream>
#include <map>
#include <queue>
#include <stack>
#include <string>
#include <vector>

using namespace std;
```

```

class Node {
 public:
  int val;
  Node* next;
  Node* random;

  Node(int _val) {
    val = _val;
    next = NULL;
    random = NULL;
  }
};
class Solution {
 public:
  Node* copyRandomList(Node* head) {
    if (head == NULL) return NULL;

    Node* res = new Node(head->val);
    Node* cur_h = head;
    Node* cur_r = res;
    map<Node*, Node*> mp;
    mp.insert(pair<Node*, Node*>(NULL, NULL));

    while (cur_h != NULL) {
      if (cur_h != head) {
        cur_r->next = new Node(cur_h->val);
        cur_r = cur_r->next;
      }
      mp.insert(pair<Node*, Node*>(cur_h, cur_r));
      cur_h = cur_h->next;
    }

    cur_h = head;
    cur_r = res;
    while (cur_h != NULL) {
      cur_r->random = mp[cur_h->random];
      cur_r = cur_r->next;

      cur_h = cur_h->next;
    }

    return res;
  }
  void input(void) {
    while (~scanf("%d", &n)) {
      a = new Node(0);
      Node* u = a;
      int x;
      vector<int> v;
      vector<Node*> record;

      for (int i = 0; i < n; i++) {
        if (i != 0) {
          u->next = new Node(0);
          u = u->next;
        }

        scanf("%d %d", &u->val, &x);
        v.push_back(x);
        record.push_back(u);
      }
      for (int i = 0; i < n; i++) {
        if (i + 1 < n) {
          record[i]->next = record[i + 1];
        }
        if (v[i] == 11111) {
          continue;
        }
```

```cpp
        record[i]->random = record[v[i]];
      }

      solve(a);
    }
  }
  void solve(Node* x) {
    Node* res;
    res = copyRandomList(x);
    otput(res);
  }
  void otput(Node* x) {
    Node* cur = x;
    int cnt = 0;
    map<Node*, int> mp;
    while (cur != NULL) {
      mp.insert(pair<Node*, int>(cur, cnt));
      cnt++;
      cur = cur->next;
    }
    cur = x;
    while (cur != NULL) {
      printf("%d ", cur->val);
      if (cur->random == NULL)
        printf("null\n");
      else
        printf("%d\n", mp[cur->random]);

      cur = cur->next;
    }
  }

 private:
  int n;
  Node* a;
};

int main() {
  freopen("./assets/fipt.txt", "r", stdin);
  freopen("./assets/fopt.txt", "w", stdout);

  Solution sol;

  sol.input();

  return 0;
}
```

Java

```java
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.PrintStream;
import java.util.HashMap;
import java.util.LinkedList;
import java.util.Map;
import java.util.Scanner;

class Node {
  int val;
  Node next;
  Node random;

  public Node(int val) {
    this.val = val;
    this.next = null;
    this.random = null;
  }
```

```java
}

class Solution {
  public Node copyRandomList(Node head) {
    if (head == null)
      return null;

    Node res = new Node(head.val);
    Node cur_h = head;
    Node cur_r = res;
    Map<Node, Node> mp = new HashMap<Node, Node>();
    mp.put(null, null);

    while (cur_h != null) {
      if (cur_h != head) {
        cur_r.next = new Node(cur_h.val);
        cur_r = cur_r.next;
      }
      mp.put(cur_h, cur_r);
      cur_h = cur_h.next;
    }

    cur_h = head;
    cur_r = res;
    while (cur_h != null) {
      cur_r.random = mp.get(cur_h.random);
      cur_r = cur_r.next;

      cur_h = cur_h.next;
    }

    return res;
  }

  public static void main(String[] args) throws FileNotFoundException {
    FileInputStream fin = new FileInputStream("./assets/fipt.txt");
    PrintStream fot = new PrintStream(new FileOutputStream("./assets/fopt.txt"));

    System.setIn(fin);
    System.setOut(fot);

    Solution sol = new Solution();

    sol.input();
  }

  public void input() {
    Scanner in = new Scanner(System.in);
    while (in.hasNext()) {
      n = in.nextInt();

      Node a = new Node(0);
      Node u = a;
      LinkedList<Integer> v = new LinkedList<Integer>();
      LinkedList<Node> record = new LinkedList<Node>();

      for (int i = 0; i < n; i++) {
        if (i != 0) {
          u.next = new Node(0);
          u = u.next;
        }
        u.val = in.nextInt();

        v.add(in.nextInt());
        record.add(u);
      }
      for (int i = 0; i < n; i++) {
        if (i + 1 < n) {
          record.get(i).next = record.get(i + 1);
        }
        if (v.get(i) == 11111) {
          continue;
        }
        record.get(i).random = record.get(v.get(i));
      }

      solve(a);
    }

    in.close();
  }

  public void solve(Node x) {
    Node res;
    res = copyRandomList(x);
    otput(res);

  }

  public void otput(Node x) {
    Node cur = x;
    int cnt = 0;
    Map<Node, Integer> mp = new HashMap<Node, Integer>();
    while (cur != null) {
      mp.put(cur, cnt);
      cnt++;
      cur = cur.next;
    }
    cur = x;
    while (cur != null) {
      System.out.print(cur.val + " ");
      if (cur.random == null)
        System.out.println("null");
      else
        System.out.println(mp.get(cur.random));

      cur = cur.next;
    }

  }

  private int n, m;
}
```

D.8 Leetcode 206

C++

```
#include <algorithm>
#include <cstdio>
#include <cstdlib>
#include <cstring>
#include <iostream>
#include <map>
#include <queue>
#include <stack>
#include <string>
#include <vector>

using namespace std;

typedef struct ListNode {
  int val;
  ListNode* next;
  ListNode(int x) : val(x), next(NULL) {}
} ListNode;
```

```

class Solution {
 public:
  ListNode* reverseList(ListNode* head) {
    if (head == NULL) return NULL;

    ListNode* st = head;
    ListNode* en = NULL;
    ListNode* cur = head;
    while (cur->next != NULL) {
      cur = cur->next;
    }
    en = cur;
    while (st != en) {
      cur = st->next;
      st->next = en->next;
      en->next = st;
      st = cur;
    }
    return en;
  }
  void input(void) {
    while (~scanf("%d", &n)) {
      a = new ListNode(0);
      ListNode* u = a;
      int x;

      for (int i = 0; i < n; i++) {
        if (i != 0) {
          u->next = new ListNode(0);
          u = u->next;
        }

        scanf("%d", &u->val);
      }

      solve(a);
    }
  }
  void solve(ListNode* x) {
    ListNode* res;
    res = reverseList(x);
    otput(res);
  }
  void otput(ListNode* x) {
    ListNode* cur = x;
    while (cur != NULL) {
      printf("%d\n", cur->val);
      cur = cur->next;
    }
  }

 private:
  int n;
  ListNode* a;
};

int main() {
  freopen("./assets/fipt.txt", "r", stdin);
  freopen("./assets/fopt.txt", "w", stdout);

  Solution sol;

  sol.input();

  return 0;
}
```

Java

```java
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.PrintStream;
import java.util.Scanner;

class Solution {
  public class ListNode {
    int val;
    ListNode next;

    ListNode(int x) {
      val = x;
    }
  }

  public ListNode reverseList(ListNode head) {
    if (head == null)
      return null;

    ListNode st = head;
    ListNode en = null;
    ListNode cur = head;
    while (cur.next != null) {
      cur = cur.next;
    }
    en = cur;
    while (st != en) {
      cur = st.next;
      st.next = en.next;
      en.next = st;
      st = cur;
    }
    return en;
  }

  public static void main(String[] args) throws FileNotFoundException {
    FileInputStream fin = new FileInputStream("./assets/fipt.txt");
    PrintStream fot = new PrintStream(new FileOutputStream("./assets/fopt.txt"));

    System.setIn(fin);
    System.setOut(fot);

    Solution sol = new Solution();

    sol.input();
  }

  public void input() {
    Scanner in = new Scanner(System.in);
    while (in.hasNext()) {
      n = in.nextInt();

      ListNode a = new ListNode(0);
      ListNode u = a;

      for (int i = 0; i < n; i++) {
        if (i != 0) {
          u.next = new ListNode(0);
          u = u.next;
        }
        u.val = in.nextInt();
      }

      solve(a);
    }

    in.close();
  }

  public void solve(ListNode x) {
    ListNode res;
    res = reverseList(x);
    otput(res);

  }

  public void otput(ListNode x) {
    ListNode cur = x;
    while (cur != null) {
      System.out.println(cur.val);
      cur = cur.next;
    }

  }

  private int n, m;
}
```

D.9 Leetcode 387

C++

```cpp
#include <algorithm>
#include <cstdio>
#include <cstdlib>
#include <cstring>
#include <iostream>
#include <map>
#include <queue>
#include <stack>
#include <string>
#include <vector>

using namespace std;

class Solution {
 public:
  int firstUniqChar(string s) {
    int ans = -1;
    int u, v;
    for (char ch = 'a'; ch <= 'z'; ch++) {
      u = s.find(ch);
      v = s.rfind(ch);
      if (u == v && u != -1) {
        if (ans == -1) {
          ans = u;
        } else {
          if (u < ans) ans = u;
        }
      }
    }
    return ans;
  }
  void input(void) {
    while (cin >> str) {
      solve(str);
    }
  }
  void solve(string s) {
    int res;
    res = firstUniqChar(s);
    otput(res);
  }
  void otput(int id) { printf("%d\n", id); }

 private:
  int n, m, t;
  string str;
};

int main() {
  freopen("./assets/fipt.txt", "r", stdin);
  freopen("./assets/fopt.txt", "w", stdout);

  Solution sol;

  sol.input();

  return 0;
}
```

Java

```java
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.PrintStream;
import java.util.Scanner;

class Solution {
  public int firstUniqChar(String s) {
    int ans = -1;
    int u, v;
    for (char ch = 'a'; ch <= 'z'; ch++) {
      u = s.indexOf(ch);
      v = s.lastIndexOf(ch);
      if (u == v && u != -1) {
        if (ans == -1) {
          ans = u;
        } else {
          if (u < ans)
            ans = u;
        }
      }
    }
    return ans;
  }

  public static void main(String[] args) throws FileNotFoundException {
    FileInputStream fin = new FileInputStream("./assets/fipt.txt");
    PrintStream fot = new PrintStream(new FileOutputStream("./assets/fopt.txt"));

    System.setIn(fin);
    System.setOut(fot);

    Solution sol = new Solution();

    sol.input();
  }

  public void input() {
    Scanner in = new Scanner(System.in);
    while (in.hasNext()) {
      String str = in.nextLine();
      solve(str);
    }

    in.close();
  }

  public void solve(String s) {
    int res;
    res = firstUniqChar(s);
```

```java
      otput(res);
    }

    public void otput(int idx) {
      System.out.println(idx);
    }

    private int n, m, t;
    private int u, v;
    private int[] numbers = new int[10000];
}
```