**算法训练 JAM计数法**

时间限制：1.0s   内存限制：256.0MB

问题描述

　　Jam是个喜欢标新立异的科学怪人。他不使用阿拉伯数字计数，而是使用**小写英文字母**计数，他觉得这样做，会使世界更加丰富多彩。在他的计数法中，每个数字的位数都是相同的（使用相同个数的字母），英文字母按原先的顺序，排在前面的字母小于排在它后面的字母。我们把这样的“数字”称为Jam数字。在Jam数字中，每个字母互不相同，而且**从左到右是严格递增**的。每次，Jam还指定使用字母的**范围**，例如，从2到10，表示只能使用{b,c,d,e,f,g,h,i,j}这些字母。如果再规定位数为5，那么，紧接在Jam数字“bdfij”之后的数字应该是“bdghi”。（如果我们用U、V依次表示Jam数字“bdfij”与“bdghi”，则U<V< span>，且不存在Jam数字P，使U<P<V< span>）。你的任务是：对于从文件读入的一个Jam数字，按顺序输出紧接在后面的5个Jam数字，如果后面没有那么多Jam数字，那么有几个就输出几个。

输入格式

　　有2行，第1行为3个正整数，用一个空格隔开：  
　　s t w  
　　（其中s为所使用的最小的字母的序号，t为所使用的最大的字母的序号。w为数字的位数，这3个数满足：1≤s<T≤26, 2≤w≤t-s ）  
　　第2行为具有w个小写字母的字符串，为一个符合要求的Jam数字。  
　　所给的数据都是正确的，不必验证。

输出格式

　　最多为5行，为紧接在输入的Jam数字后面的5个Jam数字，如果后面没有那么多Jam数字，那么有几个就输出几个。每行只输出一个Jam数字，是由w个小写字母组成的字符串，不要有多余的空格。

样例输入

2 10 5  
bdfij

样例输出

bdghi  
bdghj  
bdgij  
bdhij  
befgh

锦囊1

高精度，模拟。

锦囊2

类似使用高精度的方法要保存当前状态，每次在最后位增加一个，如果增加后的字符没使用过就OK，如果使用过再不断增加。如果增加到最大的字母，则往前进位，并把当前位设成最小的。按进位方法可能需要递归操作。

本题的C++参考代码如下：

#include<stdio.h>

int p[26], num;

void output()

{

for(int i = 0; i < num; i++)

printf("%c", p[i]+'a');

printf("\n");

}

int main()

{

int s, end;

scanf("%d%d%d", &s, &end, &num);

getchar();

for(int i = 0; i < num; i++)

{

char c = getchar();

p[i] = c - 'a';

}

int t = 5;

while(t--)

{

int crt = num - 1, tmp = end - 1;

while(crt >= 0 && p[crt--] == tmp--);

if(crt < 0)

return 0;

p[crt] = p[++crt]++;

while(++crt < num)

p[crt] = p[crt-1] + 1;

output();

}

return 0;

}

本题的C参考代码如下：

#include <stdio.h>

int main()

{

int s,t,w;

char in[26];

int ad[26],i,j,k,flag,st;

scanf("%d%d%d",&s,&t,&w);

getchar();

for(i=0;i<w;i++)

{

scanf("%c",&in[i]);

ad[i]=in[i]-'a'+1;

}

// for(i=0;i<w;i++)

// printf("%d ",ad[i]);//+'a'-1);

// printf("\n");

for(i=0;i<5;i++)

{

if(ad[0]==(t-w+1))

break;

// flag=0;

for(j=w-1;j>=0;j--)

{

if(ad[j]==t-w+j+1)

continue;

// if(!flag)

// {

// ad[j]++;

st=++ad[j];

// k=j;

for(k=j+1;k<w;k++)

ad[k]=++st;

for(k=0;k<w;k++)

printf("%c",ad[k]+'a'-1);

printf("\n");

flag=1;

break;

// }

}

}

return 0;

}

本题的Java参考代码如下：

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

import java.io.StreamTokenizer;

public class Main {

private static char chs[] = new char[27];

static {

for (int i = 1; i < chs.length; i++) {

chs[i] = (char) (96 + i);

}

}

private static int a, b, c;

private static String tmp = "";

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

StreamTokenizer st = new StreamTokenizer(new BufferedReader(new InputStreamReader(System.in)));

st.nextToken();

a = (int) st.nval;

st.nextToken();

b = (int) st.nval;

st.nextToken();

c = (int) st.nval;

st.nextToken();

String str = st.sval;

int len = str.length(), count = 0;

for (int i = len - 1; i >= 0; i--) {

if (chs[b] - str.charAt(i) >= (len - i)) {

tmp = str.substring(0, i);

int index = str.charAt(i) - 95;

for (int j = i; j < len; j++) {

tmp += chs[index++];

}

if(count==5)

break;

if (count < 5)

System.out.println(tmp);

count++;

str = tmp;

i = len;

}

}

}

}