**算法训练 字串统计**

时间限制：1.0s   内存限制：512.0MB

问题描述

　　给定一个长度为n的字符串S，还有一个数字L，统计长度大于等于L的出现次数最多的子串（不同的出现可以相交），如果有多个，输出最长的，如果仍然有多个，输出第一次出现最早的。

输入格式

　　第一行一个数字L。  
　　第二行是字符串S。  
　　L大于0，且不超过S的长度。

输出格式

　　一行，题目要求的字符串。  
  
　　输入样例1：  
　　4  
　　bbaabbaaaaa  
  
　　输出样例1：  
　　bbaa  
  
　　输入样例2：  
　　2  
　　bbaabbaaaaa  
  
　　输出样例2：  
　　aa

数据规模和约定

　　n<=60  
　　S中所有字符都是小写英文字母。

提示

　　枚举所有可能的子串，统计出现次数，找出符合条件的那个

本题的C++参考代码如下：

#include<stdlib.h>

#include<stdio.h>

#include<bitset>

#include<iostream>

#include<string>

using namespace std;

int main()

{

string s;

int l;

cin>>l;

cin>>s;

int map[60][60]={0};

for(int i=0;i<s.length();i++)

{

for(int j=0;j<s.length();j++)

{

if(i==j)

{

map[i][j]=0;

}

else

{

if(s[i]==s[j])

{

if(i>0 && j>0)

map[i][j]=map[i-1][j-1]+1;

else

map[i][j]=1;

}

}

}

}

int m[60][60]={0};//计算每一行某个数值出现的次数只计算大于l的数值

int max=0;

for(int i=0;i<60;i++)

{

for(int j=0;j<60;j++)

{

int val=map[i][j];//取出数值

if(val>=l)

{

m[i][val]++;//累加

if(m[i][val]>max)

{

max=m[i][val];

}

}

}

}

for(int j=59;j>=0;j--)//从后向前找保证字符是最长的

{

for(int i=0;i<60;i++)//从前向后找保证字符是最早出现的

{

if(m[i][j]==max)

{

cout<<s.substr(i-j+1,j)<<endl;

return 0;

}

}

}

return 0;

}

本题的C参考代码如下：

#include<stdio.h>

#include<string.h>

char b[60][60];

int c[60];

int main()

{

int l,i1,max=1,now=0,maxn=1,i2,weizhi=0,i3,i4,weizhi1,changdu1=-1;

char a[10000];

scanf("%d",&l);

getchar();

gets(a);

int n=strlen(a);

for(i1=n;i1>=l;i1--)//长度

{

weizhi=0;

for(i3=0;i3<n-i1;i3++)

c[i3]=0;

for(i2=0;i2<n;i2++)//开始位置

{

if(i2+i1>n)

break;

int ok=1;

for(i3=0;i3<weizhi;i3++)//b的位置

{ int ko=1;

for(i4=0;i4<i1;i4++)

{

if(b[i3][i4]!=a[i2+i4])

ko=0;

}

if(ko)

{ ok=0;

c[i3]++;

if(c[i3]>max)

{max=c[i3];

changdu1=i1;

weizhi1=i2;

}

break;

}

}

if(ok)

{

for(i4=0;i4<i1;i4++)

{

b[weizhi][i4]=a[i2+i4];

}

c[weizhi]=1;

weizhi++;

}

}

}

for(i1=weizhi1;i1<weizhi1+changdu1;i1++)

printf("%c",a[i1]);

printf("\n");

return 0;

}

本题的Java参考代码如下：

import java.util.Scanner;

public class Main

{

public static void main(String[] args)

{

Scanner sc=new Scanner(System.in);

int L=sc.nextInt();

String s=sc.next();

Main a=new Main();

a.childs(L,s);

}

public void childs(int L,String s)

{

String s1[][]=new String[60][60];

int in[][]=new int[60][60];

int max=0;

int td=0;

int tj=0;

for(int i=L;i<s.length();i++)

for(int j=0;j<=s.length()-i;j++)

{

s1[i][j]=s.substring(j,j+i);

}

for(int i=L;i<s.length();i++)

for(int j=0;j<=s.length()-i;j++)

for(int k=j;k<=s.length()-i;k++)

if(s1[i][k].equals(s1[i][j]))

{

in[i][j]=in[i][j]+1;

}

for(int i=L;i<=s.length();i++)

for(int j=0;j<=s.length()-i;j++)

{

if(in[i][j]>=max && td!=i)

{

max=in[i][j];

td=i;

tj=j;

}else if(in[i][j]>max)

{

max=in[i][j];

td=i;

tj=j;

}

}

System.out.println(s1[td][tj]);

}

}