**算法训练 乘积最大**

时间限制：1.0s   内存限制：256.0MB

**问题描述**  
  
　　今年是国际数学联盟确定的“2000——世界数学年”，又恰逢我国著名数学家华罗庚先生诞辰90周年。在华罗庚先生的家乡江苏金坛，组织了一场别开生面的数学智力竞赛的活动，你的一个好朋友XZ也有幸得以参加。活动中，主持人给所有参加活动的选手出了这样一道题目：  
  
　　设有一个长度为N的数字串，要求选手使用K个乘号将它分成K+1个部分，找出一种分法，使得这K+1个部分的乘积能够为最大。  
  
　　同时，为了帮助选手能够正确理解题意，主持人还举了如下的一个例子：  
  
　　有一个数字串：312， 当N=3，K=1时会有以下两种分法：  
  
　　3\*12=36  
　　31\*2=62  
  
　　这时，符合题目要求的结果是：31\*2=62  
  
　　现在，请你帮助你的好朋友XZ设计一个程序，求得正确的答案。  
  
**输入格式**  
  
　　程序的输入共有两行：  
　　第一行共有2个自然数N，K（6≤N≤40，1≤K≤6）  
　　第二行是一个长度为N的数字串。  
  
  
**输出格式**  
  
　　输出所求得的最大乘积（一个自然数）。  
  
　　**样例**输入  
  
　　4 2  
　　1231

样例输出

62

锦囊1

动态规划。

锦囊2

用F[i,j]表示前i位分j个部分最大值是多少，则F[i,j]=F[k,j-1]\*V(k+1,i)。其中V(k+1,i)表示从第k+1位到第i位的数。

本题的C++参考代码如下：

#include <iostream>

#include <string>

#include <cmath>

#include <cstring>

using namespace std;

int N,K;

string n;

int g[41] = { 0 };

long long int F[41][31] = { 0 };

inline long long int MAX(long long int a, long long int b)

{

return a>b?a:b;

}

long long int aget(int b, int c)

{

long long int temp = 0;

for(int i = b; i <= c; i++)

{

temp = (temp\*10 + g[i]);

}

return temp;

}

int main()

{

cin >> N >> K >> n;

K++;

char buf[N];

strcpy(buf,n.c\_str());

for(int i = 0; i< N; i++) g[i+1] = buf[i] - '0';

for(int i = 1; i <= N; i ++) F[i][1] = aget(1, i );

for(int j = 2; j <= K; j++)

for(int i = 1; i <= N; i++)

for(int k = 1; k < i; k++)

F[i][j] = MAX( F[i][j] , F[k][j-1]\*aget(k+1,i) );

cout << F[N][K];

return 0;

}

本题的C参考代码如下：

#include <stdio.h>

#define maxN 41

#define maxK 7

#define InfiniteMin -999999999

int main()

{

int N,K;

int i,j,k,m;

int A[maxN][maxK]; /\*A[i][j]表示前i个数有j个乘号能达到的最大乘积\*/

int s[maxN];

char num[maxN];

int temp,max;

scanf("%d%d%s",&N,&K,num);

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

s[i+1]=num[i]-'0';

for(i=1;i<=N;i++)

{

temp=0;

for(j=1;j<=i;j++)

temp=temp\*10+s[j];

A[i][0]=temp;

}

for(j=1;j<=K;j++)

{

for(i=j+1;i<=N;i++)

{

max=InfiniteMin;

for(k=i;k-1>j-1;k--)

{

temp=0;

for(m=k;m<=i;m++)

temp=temp\*10+s[m];

temp\*=A[k-1][j-1];

max=max>temp?max:temp;

}

A[i][j]=max;

}

}

printf("%d",A[N][K]);

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 {

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

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

st.nextToken();

int N = (int) st.nval;

st.nextToken();

int K = (int) st.nval;

st.nextToken();

long M = (long) st.nval;

String str = String.valueOf(M);

long dp[][] = new long[K + 1][N + 1];

for (int i = 1; i <= N; i++) {

dp[0][i] = Long.parseLong(str.substring(0, i));

}

for (int i = 1; i <= K; i++) {

for (int j = 1 + i; j <= N; j++) {

for (int k = i; k <= N; k++) {

int font = 0;

for (int l = k; l < j; l++) {

font = str.charAt(l) - '0' + font \* 10;

}

if (dp[i][j] < dp[i - 1][k] \* font)

dp[i][j] = dp[i - 1][k] \* font;

}

}

}

System.out.println(dp[K][N]);

}

}