1. **设计题目及要求**

题目1：实现若干个矩阵连乘，输出次数，以及计算的顺序

题目2：实现最长升序子序列的长度计算和输出

题目3：实现三角数的最大路径和的输出

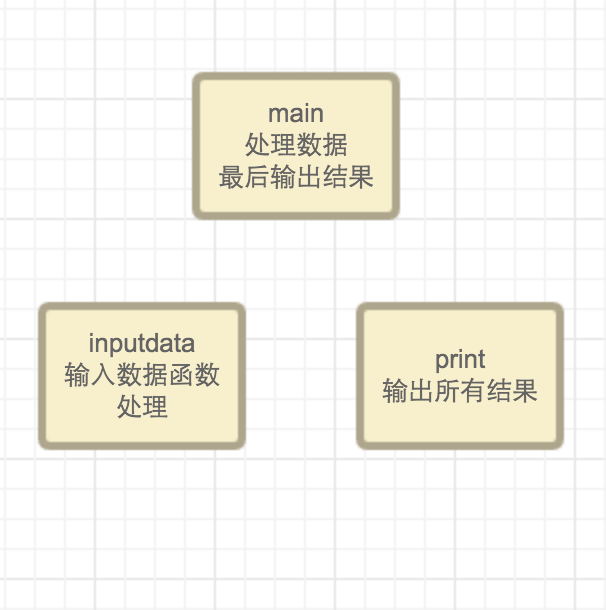
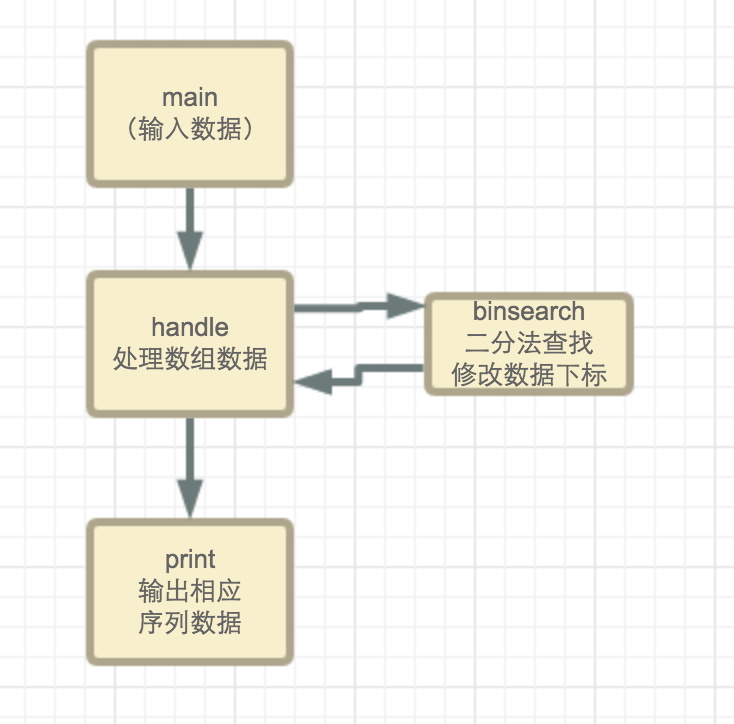
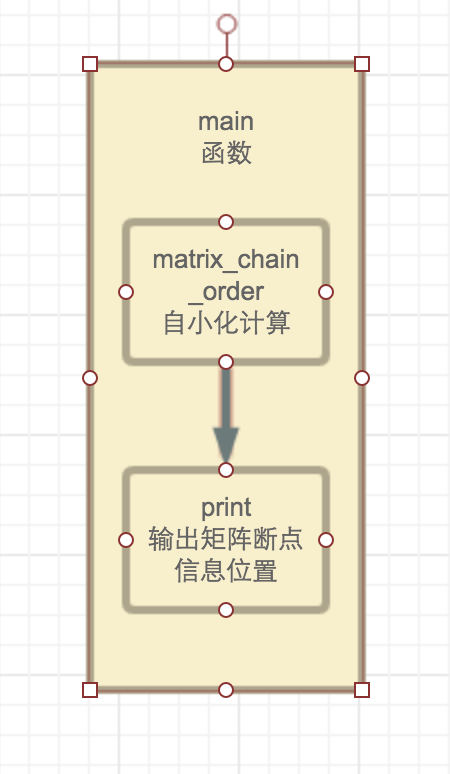
1. **解题思路**

题目1：自下而上的解题思路，先求解小问题，通过控制问题的规模（题目中用r）来由小问题，简化计算得到大问题的解。需要注意的是矩阵相乘的次数为两个矩阵的三个不同标之和。

题目2：顺序方式依次扫描数组，容器数组用于装升序数组，小数可覆盖，序列最长下标不会因小数更改，但是前面的数据会覆盖，只能纪录最长下标。二维数组遇小数，比当前行最大数小换行，在相应的位置纪录数据，用于找回最长子序列。

题目3:从低到上的方式，判断大小，累加数据。

1. **概要设计**
2. **功能模块图：**

题目1： 题目2: 题目3:

2、代码实现及结果：

题目1：

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

> File Name: 1.c

> Author: danneel

> Mail: 8119253203@qq.com

> Created Time: 三 4/13 16:11:12 2016

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <stdio.h>

#include <stdlib.h>

#define MAXVALUE 1000000

#define N 1024

int p[N+1] = {0};

int m[N+1][N+1]={0};

int s[N+1][N+1]={0};

void matrix\_chain\_order(int \*p,int len,int m[N+1][N+1],int s[N+1][N+1]);

void print(int s[N+1][N+1],int i,int j);

int main()

{

int n;

int i,j;

printf("请输入矩阵的个数：\n");

scanf("%d",&n);

printf("请输入你的矩阵数据:\n");

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

{

scanf("%d",&p[i]);

}

matrix\_chain\_order(p,n+1,m,s);

printf("the times to calculate is %d times\n", m[1][n]);

print(s,1,n);

return 0;

}

void matrix\_chain\_order(int \*p,int len,int m[N+1][N+1],int s[N+1][N+1])

{

int i,j,k,t;

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

{

m[i][i] = 0;

}

for(t=2;t<=len;t++) //当前链乘矩阵的长度

{

for(i=1;i<=len-t+1;i++) //从第一矩阵开始算起，计算长度为t的最少代价

{

j=i+t-1;//长度为t时候的最后一个元素

m[i][j] = MAXVALUE; //初始化为最大代价

for(k=i;k<=j-1;k++) //寻找最优的k值，使得分成两部分k在i与j-1之间

{

int temp = m[i][k]+m[k+1][j] + p[i-1]\*p[k]\*p[j];

if(temp < m[i][j])

{

m[i][j] = temp; //记录下当前的最小代价

s[i][j] = k; //记录当前的括号位置，即矩阵的编号

}

}

}

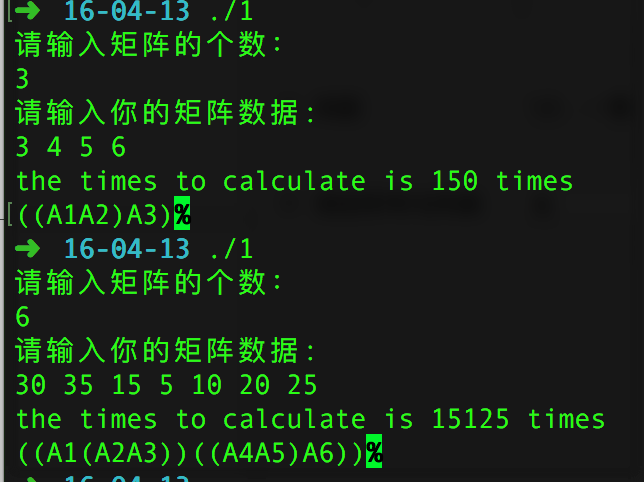
}

}

void print(int s[N+1][N+1],int i,int j)

{

if( i == j )

{

printf("A%d", i);

}

else

{

printf("(");

print(s,i,s[i][j]);

print(s,s[i][j]+1,j);

printf(")");

}

}

结果：

题目2：

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

> File Name: 2.c

> Author: danneel

> Mail: 8119253203@qq.com

> Created Time: 三 4/14 16:52:30 2016

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <stdio.h>

#include <stdlib.h>

const int N = 1024;

int a[N]; //a[i] 原始数组

int d[N]; //d[i] 装子序列，记录最长的长度

int s[N][N]; //记录子序列

int flag = 1;

void print(int s[][N], int flag, int l)

{

int c[l];

int i = 0;

int j = 0;

for (i = l; i > 0; i--)

{

for (j = flag; j > 0; j--)

{

if (s[j][i] != 0)

{

c[i] = s[j][i];

}

else

{

flag--;

}

}

}

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

{

printf("%d ",c[i]);

}

}

int BinSearch(int key, int\* d, int low, int high)

{

int mid = 0;

while(low<high)

{

mid = low/2 +high/2;

if(key>d[mid] && key<=d[mid+1])

{

return mid;

}

else if(key>d[mid])

{

low = mid+1;

}

else

{

high = mid-1;

}

}

return high;

}

int handle(int\* a, int n, int\* d)

{

int i = 0;

int j = 0;

int k = 1;

int re = 0;

int temp = 0;

int q = 1;

flag = 1;

d[1] = a[1];

s[k][1] = a[1];

int len = 1;

for(i = 2; i <= n; i++)

{

if(d[len]<a[i])

{

j = ++len;

d[j] = a[i];

q = j;

s[k][q] = a[i];

if (re == 1)

{

flag++;

}

re = 0;

}

else

{

temp = BinSearch(a[i],d,1,len) + 1;

j = temp;

d[j] = a[i];

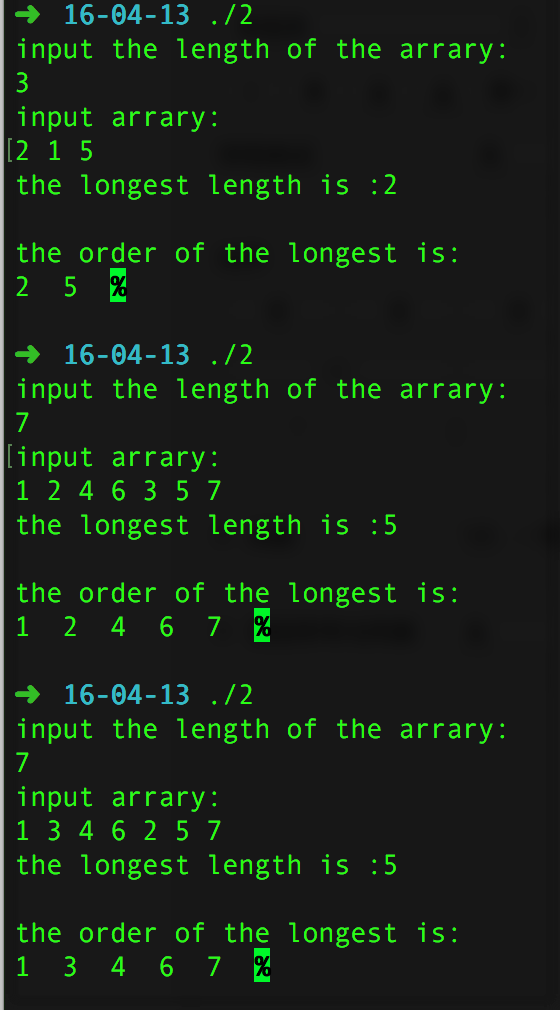
q = j;

if (s[k][q] > a[i])

{

k++;

re = 1;

}

s[k][q] = a[i];

}

}

return len;

}

int main(int args, char \*argv[])

{

int length = 0;

int l = 0;

printf("input the length of the arrary:\n");

scanf("%d",&length);

printf("input arrary:\n");

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

{

scanf("%d",&a[i]);

}

l = handle(a, length, d);

printf("the longest length is :%d\n\n", l);

printf("the order of the longest is:\n");

print(s, flag, l);

return 0;

}

结果：

题目3:

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

> File Name: 3.c

> Author: danneel

> Mail: 8119253203@qq.com

> Created Time: 一 4/18 15:04:26 2016

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include<stdio.h>

#include<stdlib.h>

int a[1024][1024] = {0};

int n = 0;

void inputdata()

{

int i = 0;

int j = 0;

printf("please input your lines:\n");

scanf("%d", &n);

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

{

printf("please input the number of %d line.\n", i);

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

{

scanf("%d", &a[i][j]);

}

}

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

{

a[n+1][i] = a[n][i];

}

}

void print()

{

int i = 0;

int j = 0;

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

{

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

{

printf("%d\t", a[i][j]);

}

printf("\n");

}

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

{

printf("%d\t", a[n+1][i]);

}

printf("\n\n");

}

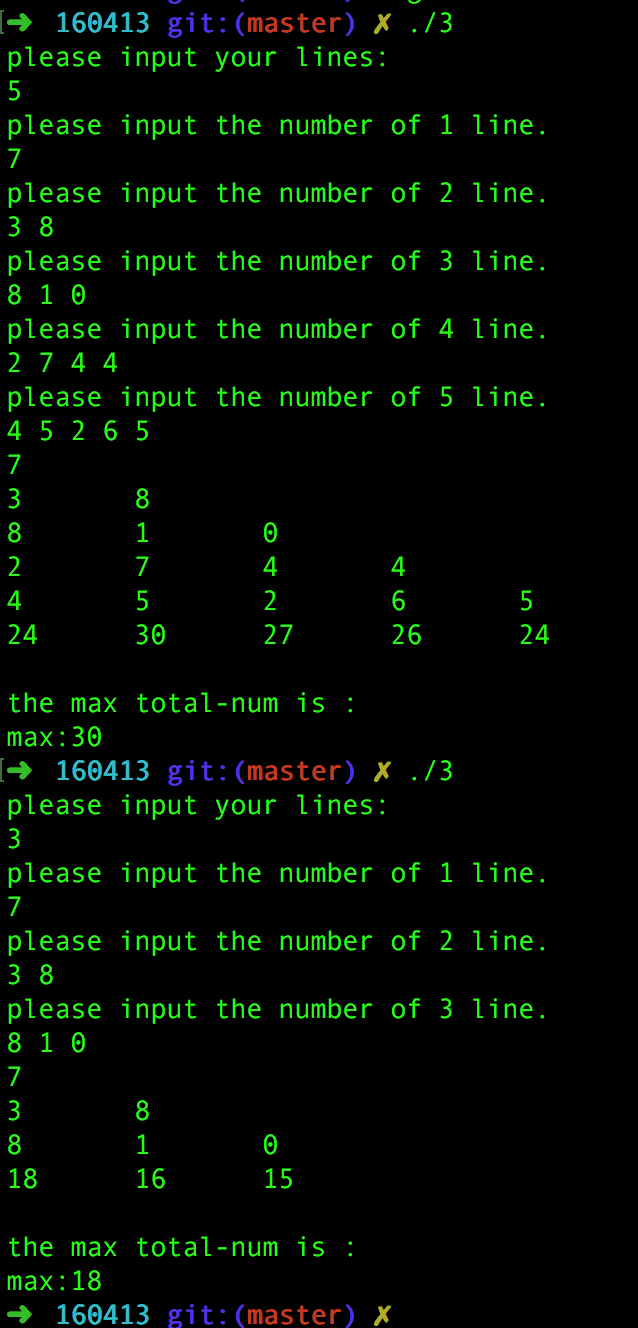
int main(int argc, char \*argv[])

{

int i = 0;

int x = 0;

int y = 0;

int max = 0;

inputdata();

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

{

y = i;

for (x = n; x > 0; x--)

{

if (a[x-1][y-1] >= a[x-1][y])

{

a[n+1][i] += a[x-1][y-1];

y = y-1;

}

else

{

a[n+1][i] += a[x-1][y];

y = y;

}

}

max = (max > a[n+1][i])? max:a[n+1][i];

}

print();

printf("the max total-num is :\n");

printf("max:%d\n", max);

}

结果：