

线上课程可用BILIBILI客户端扫二维码或者单击URL链接

[https://www.bilibili.com/video/BV12K4y1X7qQ?p=1](https://www.bilibili.com/video/BV12K4y1X7qQ?p=1%0c)

# 穷举

韩信点兵。

韩信有一队兵，他想知道有多少人，便让士兵排队报数。

按从1至 5报数，最末一个士兵报的数为1；

按从1至6报数，最末一个士兵报的数为5；

按从 1至 7报数，最末一个士兵报的数为 4；

按从 1至 11报数，最末一个士兵报的数为 10。

你知道韩信至少有多少兵吗？（请特别注意是“至少”）

---------------------------------------------------------------------------------

#include <iostream>

using namespace std;

int main(){

int ans = -1;

for (int i = 0 ; ; i ++){

if (i % 5 == 1 && i % 6 == 5 && i % 7 == 4 && i % 11 == 10){

ans = i;

break;

}

}

cout << ans << endl;

return 0;

}

---------------------------------------------------------------------------------

输入一个大于2的正整数n，输出其所有的素数，用判断对2~sqrt(n)是否整除法，会减一半分。

#include <iostream>

using namespace std;

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

{

return 0;

}

---------------------------------------------------------------------------------

#include <iostream>

using namespace std;

const int N = 1e5 + 10;

int v[N], prime[N], cnt;

void get\_prime(int n){

// <= n

for (int i = 2; i \* i <= n ;i ++){

if (v[i]) continue;

for (int j = i \* i; j <= n ;j += i){

v[j] = 1;

}

}

}

int main(){

int n;

cin >> n;

get\_prime(n);

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

if (!v[i]){

cout << i << endl;

}

}

return 0;

}

---------------------------------------------------------------------------------

# 递推：

猴子摘桃（代码实现）

有一猴子第1天摘下若干个桃子，当即吃了一半，还不过瘾，又多吃了1个。第2天早上又将剩下的桃子吃掉一半，又多吃了1个。以后每天早上都吃了前一天剩下的一半后又多吃1个。到第d天早上想再吃时，见只剩下1个桃子了。

求第1天共摘了多少个桃子。

粒子裂变（代码实现）

#include <iostream>

using namespace std;

int main(){

int t;

cin >> t;

int a = 1;

for (int i = 1; i <= t - 1; i ++){

a = (a + 1) \* 2;

}

cout << a << endl;

return 0;

}

---------------------------------------------------------------------------------

核反应堆中有α和β两种粒子，每秒钟内一个α粒子可以裂变为3个β粒子，而一个β粒子可以裂变为1个α粒子和2个β粒子。若在t=0时刻的反应堆中只有一个α粒子，求在t秒时反应堆裂变产生的α粒子和β粒子数。

---------------------------------------------------------------------------------

#include <iostream>

using namespace std;

int main(){

int t;

cin >> t;

int a = 1, b = 0;

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

int ta = 0, tb = 0;

tb += 3 \* a;

ta += b, tb += 2 \* b;

a = ta, b = tb;

}

cout << b << endl << a;

return 0;

}---------------------------------------------------------------------------------

# 分治法：

1. **汉诺塔问题（10分）**

|  |  |
| --- | --- |
| 得分 |  |

完成汉诺塔递归调用需要的函数，n盘片从A柱经过B，搬到C柱

输入：A柱上的盘子数量n（n>=1）

输出：移动标示

样例输入

3

样例输出

A--C

A--B

C--B

A--C

B--A

B--C

A--C

代码：将汉诺塔函数部分填写完成，用递归实现。

#include <iostream>

#include <ctime>

using namespace std;

void print(char A,char C)

{

//cout<<A<<"--"<<C<<endl;

}

void hanoi(int n,char A,char B,char C)

{

}

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

{

int n;

cout<<"请输入A座上的盘子数目：";

cin>>n;

hanoi(n,'A','B','C');

return 0;

}

#include <iostream>

#include <ctime>

using namespace std;

void print(char A,char C)

{

cout<<A<<"--"<<C<<endl;

}

void hanoi(int n,char A,char B,char C)

{

if (n == 1) print(A, C);

else {

hanoi(n - 1, A, C, B);

print(A, C);

hanoi(n - 1, B, A, C);

}

}

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

{

int n;

cout<<"请输入A座上的盘子数目：";

cin>>n;

hanoi(n,'A','B','C');

return 0;

}

---------------------------------------------------------------------------------

1. **折半查找问题（10分）**

|  |  |
| --- | --- |
| 得分 |  |

折半查找，需要用递归实现

题目描述：既书中所谓的二分查找法，给定一个数据类型为DataType的顺序数组（数据元素没有重复），完成函数，查找数据x是否存在，存在则返回该数据的下标，不存在则返回-1。

完成函数体代码：

int BinarySearch(DataType a[],int L,int R,DataType x)

//a 代表顺序数组，L代表左边界，R代表右边界， x代表要查找的元素

//存在则返回该数据的下标，不存在则返回-1

{

}

---------------------------------------------------------------------------------

#include <iostream>

using namespace std;

typedef int DataType;

int BinarySearch(DataType a[],int L,int R,DataType x)

//a 代表顺序数组，L代表左边界，R代表右边界， x代表要查找的元素

//存在则返回该数据的下标，不存在则返回-1

{

if (L > R) return -1;

int mid = (L + R) / 2;

if (a[mid] == x) return mid;

if (a[mid] < x) return BinarySearch(a, mid + 1, R, x);

return BinarySearch(a, L, mid - 1, x); // a[mid] > x

}

int main(){

int a[] = {1, 3, 5, 8, 12, 25, 67, 100};

int x;

while (cin >> x){

cout << BinarySearch(a, 0, 7, x) << endl;

}

return 0;

}

---------------------------------------------------------------------------------

1. **委员会问题（10分）**

|  |  |
| --- | --- |
| 得分 |  |

**题目描述**

设计求解委员会问题，从一个有n个人的团体中抽出k(k<=n)个人组成一个委员会，计算有多少种组成方法

**输入**

n和k(k<=n)，表示总人数和抽出的专家人数

**输出**

多少种组成方法

**样例输入**

5 3

**样例输出**

10

代码：委员会问题递归函数部分填写完成。

#include <stdio.h>

int Comm(int n,int k);

int main()

{

int a,b,s;

scanf("%d %d",&a,&b);

s=Comm(a,b);

printf("%d",s);

printf("\n");

return 0;

}

**int Comm(int n,int k)**

**{**

**if (k > n || n < 1 || k < 0) return 0;**

**if (n == 1 || n == k || k == 0) return 1;**

**return Comm(n - 1, k) + Comm(n - 1, k - 1);**

**}**

1. **快速排序(10分)**

|  |  |
| --- | --- |
| 得分 |  |

将给定的一组数字序列按照快速排序的方式排序，写出每次排序的中间结果，如果有缺漏，甚至直接写最后结果的，酌情扣分。（注意按升序排）

25 16 54 86 56 123 91 13 8 34 *i j*

第0趟 (25 16 54 86 56 123 91 13 8 34) 1 10

第1趟 (25 16 8 86 56 123 91 13 54 34) 3 9

第2趟 (25 16 8 13 56 123 91 86 54 34) 4 8

第3趟 (13 16 8 25 56 123 91 86 54 34) 5 4

第4趟 (13 16 8 ) 25 56 123 91 86 54 34 1 3

第5趟 (13 8 16 ) 25 56 123 91 86 54 34 2 3

第6趟 (8 13 16 ) 25 56 123 91 86 54 34 3 2

第7趟 8 13 16 25 (56 123 91 86 54 34) 5 9

第8趟 8 13 16 25 (56 34 91 86 54 123) 6 10

第9趟 8 13 16 25 (56 34 54 86 91 123) 7 9

第10趟 8 13 16 25 (54 34 56 86 91 123) 8 7

第11趟 8 13 16 25 (54 34 ) 56 86 91 123 5 6

第12趟 8 13 16 25 ( 34 54) 56 86 91 123 6 6

第13趟 8 13 16 25 34 54 56 (86 91 123) 8 10

第14趟 8 13 16 25 34 54 56 (86 91 123) 9 8

第15趟 8 13 16 25 34 54 56 86 (91 123) 9 10

第16趟 8 13 16 25 34 54 56 86 (91 123) 10 9

结果 8 13 16 25 34 54 56 86 91 123

TIPS:整行标红的表示选择新的基准数，没有发生元素交换，因此可以不写。

1. **归并排序(10分)**

|  |  |
| --- | --- |
| 得分 |  |

将给定的一组数字序列按照归并排序的方式排序，写出每次排序的中间结果，如果有缺漏，甚至直接写最后结果的，酌情扣分。（注意按升序排）

25 16 54 86 56 123 91 13 8 34

第0趟：25||||16||| 54 || 86 56|123 91 13 8 34

第1趟：16 25 ||| 54 || 86 56 | 123 91 13 8 34

第2趟：16 25 54 || 86 ||| 56 | 123 91 13 8 34

第3趟：16 25 54 || 56 86 | 123 91 13 8 34

第4趟：16 25 54 56 86 | 123 |||| 91 ||| 13 || 8 34

第5趟：16 25 54 56 86 | 91 123 ||| 13 || 8 34

第6趟：16 25 54 56 86 | 13 91 123 || 8 ||| 34

第7趟：16 25 54 56 86 | 13 91 123 || 8 34

第8趟：16 25 54 56 86 | 8 13 34 91 123

第9趟：8 13 16 25 34 54 56 86 91 123

动态规划

**用动态规划方式优化递归实现的Fabonacci数列（15分）**

|  |  |
| --- | --- |
| 得分 |  |

优化下列代码，要求用表（数组）记录中间结果，减少重复计算运算量。

#include <stdio.h>

int fibo(int n){

if(n==0) return 0;

if(n==1) return 1;

return fibo(n-1)+fibo(n-2);

}

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

{

printf("%d\n",fib(40));

return 0;

}

----------------------------------------------------------

#include <stdio.h>

int f[200];

int fibo(int n){

if(n==0) return 0;

if(n==1) return 1;

if (f[n]) return f[n];

f[n] = fibo(n-1)+fibo(n-2);

return f[n];

}

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

{

printf("%d\n",fibo(40));

return 0;

}

**最长公共子序列（15分）**

给定两个字符串

X = “JDKGLDJGLA”

Y = “SJFLSDJG”

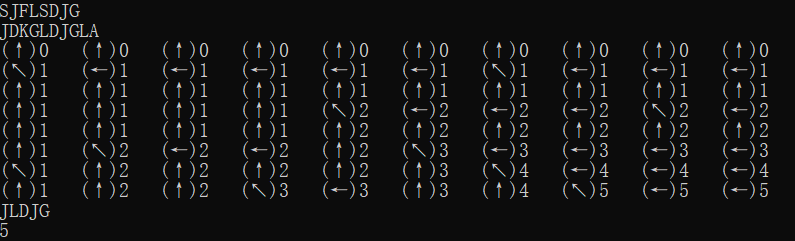
求X和Y最长公共子序列

LCS

长度为5

JLDJG

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | J | D | K | G | L | D | J | G | L | A |
|  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| S | 0 | ↑0 | ↑0 | ↑0 | ↑0 | ↑0 | ↑0 | ↑0 | ↑0 | ↑0 | ↑0 |
| J | 0 | ↖1 | ←1 | ←1 | ←1 | ←1 | ←1 | ↖1 | ←1 | ←1 | ←1 |
| F | 0 | ↑1 | ↑1 | ↑1 | ↑1 | ↑1 | ↑1 | ↑1 | ↑1 | ↑1 | ↑1 |
| L | 0 | ↑1 | ↑1 | ↑1 | ↑1 | ↖2 | ←2 | ←2 | ←2 | ↖2 | ←2 |
| S | 0 | ↑1 | ↑1 | ↑1 | ↑1 | ↑2 | ↑2 | ↑2 | ↑2 | ↑2 | ↑2 |
| D | 0 | ↑1 | ↖2 | ←2 | ←2 | ↑2 | ↖3 | ←3 | ←3 | ←3 | ←3 |
| J | 0 | ↖1 | ↑2 | ↑2 | ↑2 | ↑2 | ↑3 | ↖4 | ←4 | ←4 | ←4 |
| G | 0 | ↑1 | ↑2 | ↑2 | ↖3 | ←3 | ←3 | ↑4 | ↖5 | ←5 | ←5 |



**矩阵链乘问题（15分）**

题目描述：矩阵A和B可乘的条件是矩阵A的列数等于矩阵B的行数。

给定n个矩阵｛A1,A2,…,An｝，其中Ai与Ai+1是可乘的，i=1,2 ,…,n-1。如何确定计算矩阵连乘积的计算次序，使得依此次序计算矩阵连乘积需要的数乘次数最少，并求出最少需要的乘次数。这些矩阵的维数分别是：

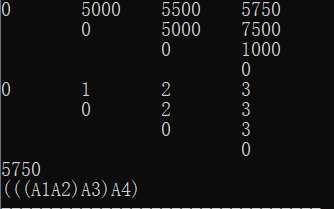
{5x50,50x20,20x5,5x10}、要求

1）按动态规划解决最少需要的乘次数，2）求出乘的先后秩序

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 |
| 1 | 0 | 5000 | 5500 | 5750 |
| 2 | / | 0 | 5000 | 7500 |
| 3 | / | / | 0 | 1000 |
| 4 | / | / | / | 0 |
|  | 1 | 2 | 3 | 4 |
| 1 | 0 | 1 | 2 | 3 |
| 2 | / | 0 | 2 | 3 |
| 3 | / | / | 0 | 3 |
| 4 | / | / | / | 0 |

最少需要：5750

乘法次序为：(((A1A2)A3)A4)



**0-1背包问题（15分）**

题目描述：一个旅行者有一个最多能用15公斤的背包，现在有5件物品，它们的重量分别是{3，2，1，5，10},它们的价值分别为{4，5，1，7，9}若每种物品只有一件求旅行者能获得最大总价值，并求出包含的是哪几样物品。

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | V | | | | | | | | | | | | | | | | |
| i |  | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| 1 | 0 | 0 | 0 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 |
| 2 | 0 | 0 | 5 | 5 | 5 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 |
| 3 | 0 | 1 | 5 | 6 | 6 | 9 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 |
| 4 | 0 | 1 | 5 | 6 | 6 | 9 | 10 | 12 | 13 | 13 | 16 | 17 | 17 | 17 | 17 | 17 |
| 5 | 0 | 1 | 5 | 6 | 6 | 9 | 10 | 12 | 13 | 13 | 16 | 17 | 17 | 17 | 17 | 18 |

最大总价值为18，购买了1、2、5件物品

**贪心**

**活动安排问题（15分）**

题目描述：假设要在同一会场里安排一批活动，活动安排尽可能多。以下这组数据代表了11个活动的开始时间和结束时间，请你安排下。

1 6

3 5

0 4

5 7

3 8

5 9

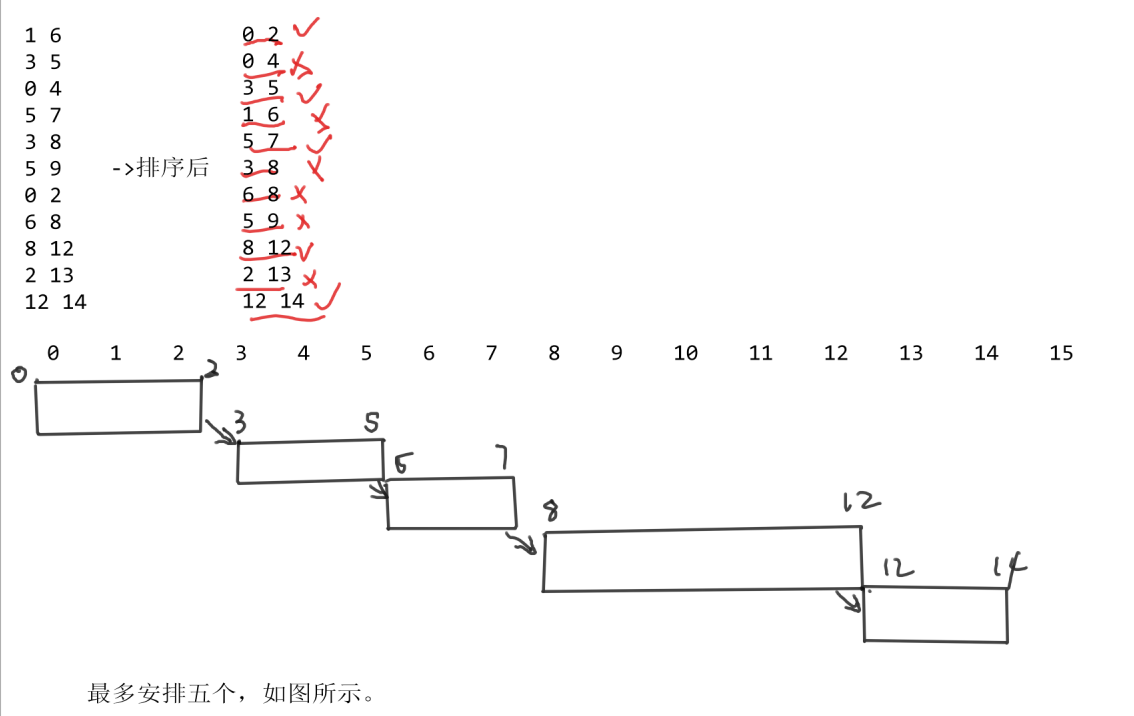
0 2

6 8

8 12

2 13

12 14

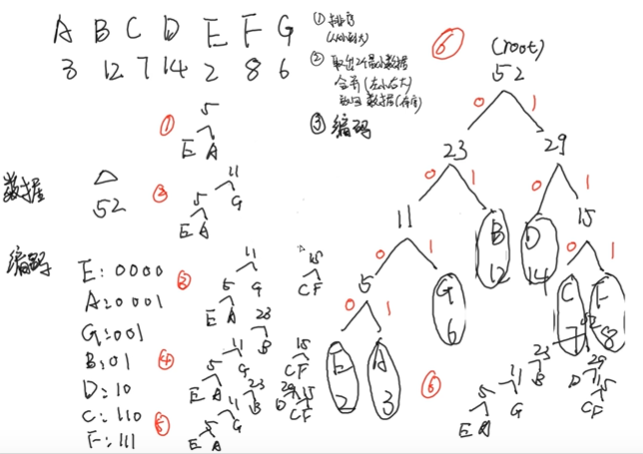


**哈夫曼编码（15分）**

已知下列字符A、B、C、D、E、F、G的出现频率分别为3、12、7、14、2、8，6

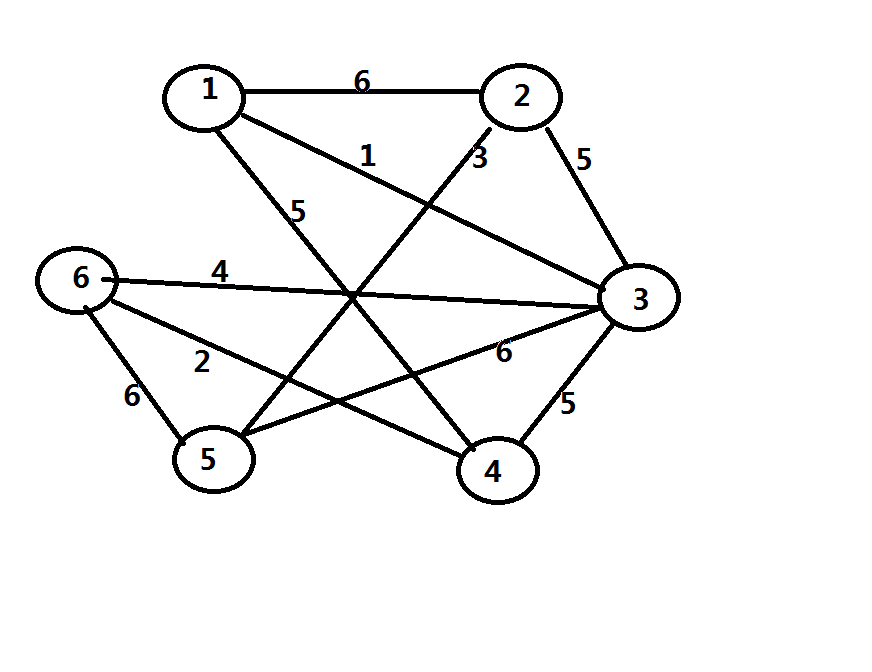
（其中哈夫曼树建立时，默认左小右大，左0右1，输入结果也应按字母出现的先后顺序）

要求1）哈夫曼生成的详细步骤，2）每个字符的编码



**最小生成树（15分）**

有向网如下图所示，试求出1）最小生成树（详细步骤），2）最小生成树的权值和，方法不限。



6 10

1 2 6

1 3 1

1 4 5

2 3 5

2 5 3

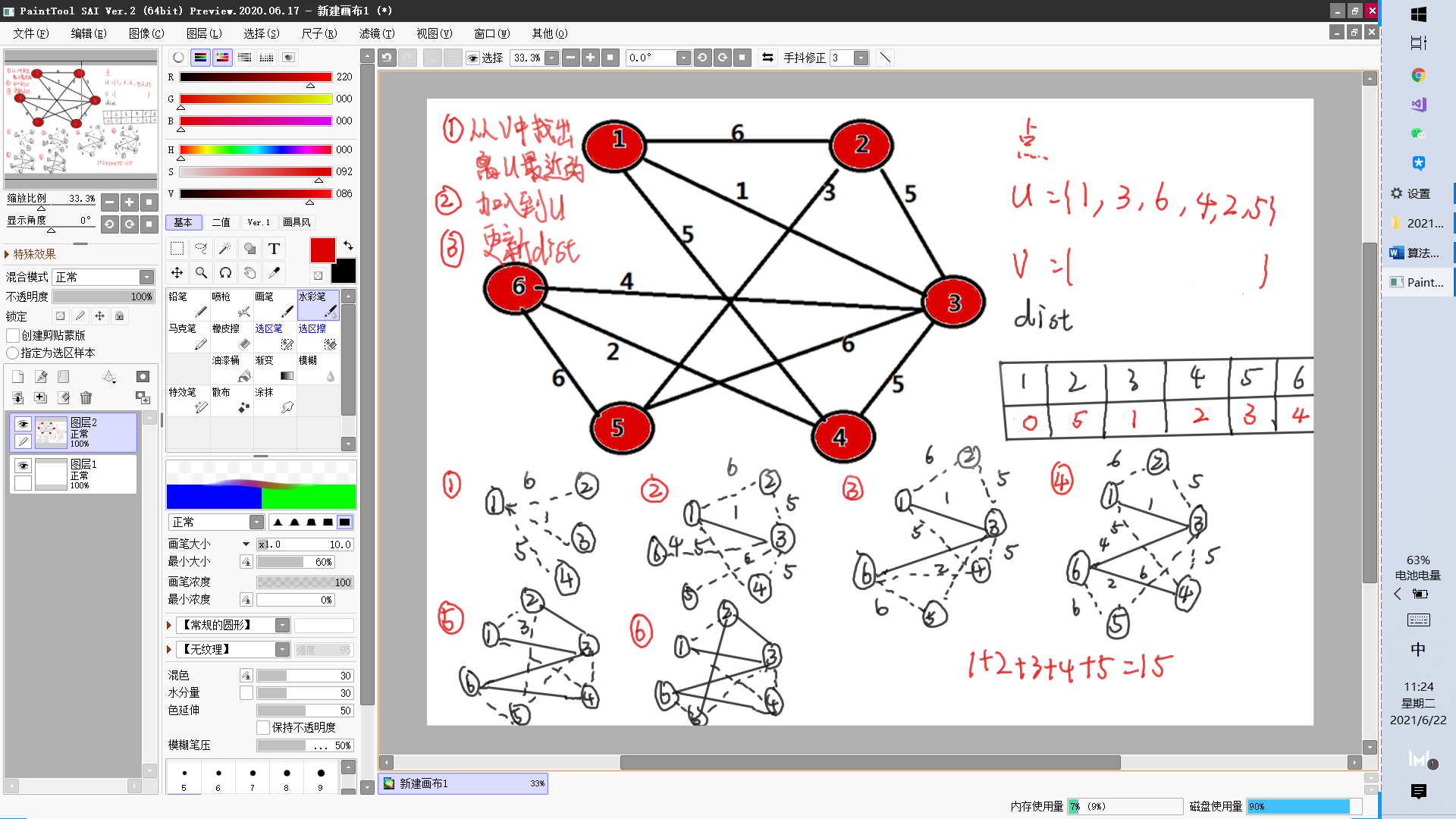
3 4 5

3 5 6

3 6 4

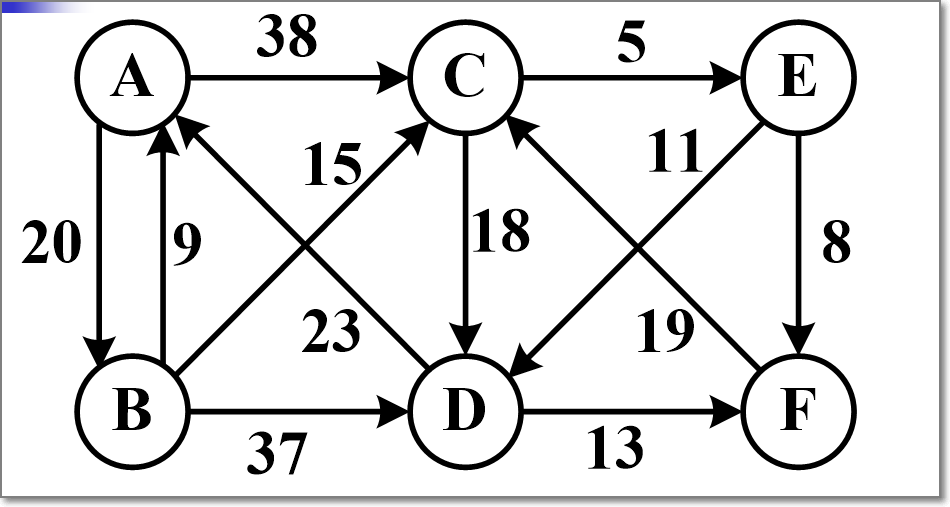
4 6 2

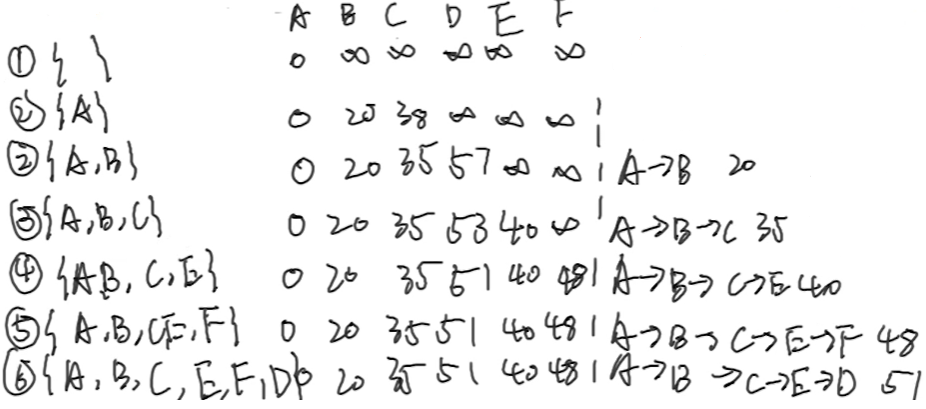
5 6 6



**单源最短路径（15分）**

有向网如下图所示，试用迪杰斯特拉算法求出从顶点1到其他各顶点间的最短路径。





**n皇后问题，代码实现（20分）**

**题目描述**

在n×n格的棋盘上放置彼此不受攻击的n个皇后。按照国际象棋的规则，皇后可以攻击与之处在同一行或同一列或同一斜线上的棋子。n后问题等价于再n×n的棋盘上放置n个皇后，任何2个皇后不妨在同一行或同一列或同一斜线上。

**输入**

输入给定棋盘的大小n（n>=4）

**输出**

输出共有多少种摆放方法

**样例输入**

8

**样例输出**

92

---------------------------------------------------------------------------------

#include <iostream>

using namespace std;

int v[20];

bool place(int k) {

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

if (v[i] == v[k]) return false;

if (abs(i - k) == abs(v[i] - v[k])) return false;

}

return true;

}

int Nqueue(int n) {

int ans = 0;

int k = 1;

v[1] = 0;

while (k) {

v[k]++;

while (!place(k)) v[k]++;

// 一定是已经放下的

if (v[k] <= n) {

if (k == n) ans++; // 最后一行已经合法

else {

k++;

v[k] = 0;

}

}

else k--;

}

return ans;

}

int main() {

int n;

cin >> n;

cout << Nqueue(n) << endl;

return 0;

}

---------------------------------------------------------------------------------

**子集和数问题，代码实现（20分）**

**题目描述**

子集和问题的一个实例为〈S,t〉。其中，S={ 1 x ， 2 x ，…， n x }是一个正整数的集合，c是一个正整数。子集和问题判定是否存在S的一个子集S1，使得s1中的各元素之和等于c。

**输入**

第一行有2个正整数n和c，n表示S的大小，c是子集子集和的目标值。接下来的1行中，有n个正整数，表示集合S中的元素。

**输出**

输出子集和问题的解，非下标，同时在n个数为升序的前提下，所出的子集顺序是唯一的。并输出共有几种组合，如果无解，则输出0

注意，每个数字之间空2个空格，每行最后的数字也要空两个空格

**样例输入**

10 60

30 46 86 5 11 66 84 49 69 55

**样例输出**

5 55

11 49

2

---------------------------------------------------------------------------------

#include <iostream>

#include <algorithm>

using namespace std;

const int N = 1e5 + 10;

int w[N], vis[N];

int n, M;

int ans;

void SumOfSet(int s, int k) {

if (k >= n) return;

vis[k] = 1;

if (s + w[k] == M) {

for (int i = 0; i <= k; i++) {

if (vis[i]) cout << w[i] << " ";

}

cout << endl;

ans++;

}

else if (s + w[k] <= M) {

SumOfSet(s + w[k], k + 1);

}

if (s + w[k + 1] <= M) {

vis[k] = 0;

SumOfSet(s, k + 1);

}

}

int main() {

cin >> n >> M;

for (int i = 0; i < n; i++) cin >> w[i];

sort(w, w + n);

SumOfSet(0, 0);

cout << ans << endl;

return 0;

}

代码已通过评测