完善程序

1. 1.(矩阵变幻)有一个奇幻的矩阵，在不停的变幻，其变幻方式为：数字0变成矩阵

00

01

数字1变成矩阵

1 1

1 0

最初该矩阵只有一个元素0,变幻n次后，矩阵会变成什么样?例如，矩阵最初为：[0];矩阵变幻1次后：

0 0

0 1

矩阵变幻2次后：

0 0 0 0

0 1 0 1

0 0 1 1

0 1 1 0

输入一行一个不超过10的正整数n。输出变幻n次后的矩阵。 试补全程序。提示：

“<<”表示二进制左移运算符，例如：(11)\_2<<2=(1100)\_2(11)2<<2=(1100)2;

而“^”表示二进制异或运算符，它将两个参与运算的数中的每个对应的二进制位—进行比较，若两个二进制位相同，则运算结果的对应二进制位为0,反之:为1。

#include <cstdio>

using namespace std;

Int n;

const int max\_size = 1 << 10;

int res[max\_size][max\_size];

void recursive(int x, int y, int n, int t) {

if (n == 0) {

res[x][y] = ①;

return;

}

int step = 1 << (n - 1);

recursive(②, n - 1, t);

recursive(x, y + step, n - 1, t);

recursive(x + step, y, n - 1, t);

recursive(x③, n - 1, !t);

}

int main() {

scanf("%d", &n);

recursive(0, 0, ④);

int size =⑤;

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

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

printf("%d", res[i][j]);

puts("");

}

return 0;

}

①处应填()

1. n%2 B.0 C.t D.1

②处应填()

A.x-step,y-step B.X,y-step C.x-step,y D.x,y

③处应填()

A.x-step,y-step B.x+step,y+step C.x-step,y D.X,y-step

④处应填()

A.n-1,n%2 B.n,0 C.n,n%2 D.n-1,0

⑤处应填()

A.1<<(n+1) B.1<<n C.n+1 D.1<<(n-1)

**【答案】**

1. ①处应填C.t

2. ②处应填D.x,y

3. ③处应填B.x+step,y+step

4. ④处应填B.n,0

5. ⑤处应填B.1<<n

**```**

二、

(计数排序)计数排序是一个广泛使用的排序方法。下面的程序使用双关键字计数排序，将n对10000以内的整数，从小到大排序。

例如有三对整数(3,4)(3,4)、(2,4)(2,4)、(3,3)(3,3),那么排序之后应该是

(2,4)(2,4)、(3,3)(3,3)、(3,4)(3,4)。

输入第一行为nn,接下来nn行，第ii行有两个数a[i]a[i]和b[i]b[i],分别表

示第ii对整数的第一关键字和第二关键字。

从小到大排序后输出。

数据范围 1<n<10^71<n<107, 1<a[i],b[i]<10^41<a[i],b[i]<104

提示：应先对第二关键字排序，再对第一关键字排序。数组ord[]存储第二关键字排序的结果，数组res[]存储双关键字排序的结果。

试补全程序

#include <cstdio>

#include <cstring>

using namespace std;

const int maxn = 10000000;

const int maxs = 10000;

int n;

unsigned a[maxn], b[maxn], res[maxn], ord[maxn];

unsigned cnt[maxs + 1];

int main() {

scanf(“%d”, &n);

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

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

memset(cnt, 0, sizeof(cnt));

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

①; // 利用cnt数组统计数量

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

cnt[i + 1] += cnt[i]

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

②; // 记录初步排序结果

memset(cnt, 0, sizeof(cnt));

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

③; // 利用cnt数组统计数量

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

cnt[i + 1] += cnt[i];

for (int i = n - 1; i >= 0; --i)

④; // 记录最终排序结果

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

printf("%d %d", ⑤);

return 0;

}

①处应填()

A.++cnt [i] B.++cnt[b[i]] C.++cnt[a[i]\*maxs+b[i]] D.++cnt[a[i]]

②处应填()

A.ord[--cnt[a[i]]]=i B.ord[--cnt[b[i]]]=a[i]

C.ord[--cnt[a[i]]]=b[i] D.ord[--cnt[b[i]]]=i

③处应填()

A.++cnt[b[i]] B.++cnt[a[i]\*maxs+b[i]] C.++cnt[a[i]] D.++cnt [i]

④处应填()

A.res[--cnt[a[ord[i]]]]=ord[i] B.res[--cnt[b[ord[i]]]]=ord[i]

C.res[--cnt[b[i]]]=ord[i] D.res[--cnt[a[i]]]=ord[i]

⑤处应填()

1. a[i],b[i] B.a[res[i]],b[res[i]]

C.a[ord[res[i]]]jb[ord[res[i]]] D.a[res[ord[i]]]j b[res[ord[i]]]

答案：

①处应填(B) ++cnt[b[i]]

②处应填(D) ord[--cnt[b[i]]]=i

③处应填(A) ++cnt[a[i]]

④处应填(D) res[--cnt[a[ord[i]]]]=ord[i]

⑤处应填(B) a[res[i]], b[res[i]]

三、完善程序

1. (质因数分解)给出正整数n,请输出将n质因数分解的结果，结果从小到大输出。

例如：输入n=120,程序应该输出22235,表示120=2×2×2×3×5。输入保证2≤n≤109。提示：先从小到大枚举变量i,然后用i不停试除n来寻找所有的质因子。

试补全程序。

#include <cstdio>

using namespace std;

int main() {

int n, i;

scanf("%d", &n);

for (i =①; ② <= n; i++) {

③{

printf("%d", i);

n = n / i;

}

}

if (④)

printf("%d", ⑤);

return 0;

}

1) ①处应填()

A. n - 1

B. θ

C. 1

D. 2

2) ②处应填()

A. n / i

B. n / (i \* i)

C. i \* i \* i

D. i \* i

3) ③处应填( )

A. if (i \* i <= n)

B. if (n % i == 0)

C. while (n % i == 0)

D. while (i \* i <= n)

4) ④处应填()

A. n > 1

B. n <= 1

C. i + i <= n

D. i < n / i

5) ⑤处应填( )

A. 2

B. i

C. n / i

D. n

1) ①处应填 D. 2

2) ②处应填 D. i \* i

3) ③处应填 C. while (n % i == 0)

4) ④处应填 A. n > 1

5) ⑤处应填 D. n

答案

① 2

②i \* i

③while (n % i == 0)

④n > 1

⑤ n

#include <cstdio>using namespace std;

int main() {

int n, i;

scanf("%d", &n);

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

while (n % i == 0) {

printf("%d", i);

n = n / i;

}

}

if (n > 1)

printf("%d", n);

return 0;}

。

四、

(最小区间覆盖)给出n个区间，第i个区间的左右端点是[ai,b₁]。现在要在这些区间中选出若干个，使得区间[0,m]被所选区间的并覆盖(即每一个θ≤i≤m都在某个所选的区间中)。保证答案存在，求所选区间个数的最小值。

输入第一行包含两个整数n和m(1≤n≤5000,1≤m≤10⁹)。接下来n行，每行两个整数ai,b₁(O≤a₁,b₁≤m)。

提示：使用贪心法解决这个问题。先用θ(n²)的时间复杂度排序，然后贪心选择这些区间，

试补全程序。

#include <iostream>

using namespace std;

const int MAXN = 5000;

int n, m;

struct segment {

int a, b;

} A[MAXN];

void sort() //排序

{

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

for (int j = 1; j < n; j++)

if (①)

{

segment t = A[j];

②

}

}

int main()

{

cin >> n >> m;

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

cin >> A[i].a >> A[i].b;

sort();

int p = 1;

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

if (③)

A[p++] = A[i];

n = p;

int ans = 0, r = 0;

int q = 0;

while (r < m)

{

while (④)

{

q++;

⑤;

}

ans++;

}

cout << ans << endl;

return 0;

}

1)①处应填()

A. A[j].b<A[j-1].b

B. A[j].b>A[j-1].b

C.A[j].a<A[j-1].a

D. A[j].a>A[j -1].a

2)②处应填( )

A.A[j-1]=A[j];A[j]=t;

B.A[j+1]=A[j];A[j]=t;

C.A[j]=A[j-1];A[j-1]=t;

D.A[j]=A[j+1];A[j+1]=t;

3)③处应填( )

A. A[i].b<A[p -1].b

C. A[i].b>A[p -1].b

B. A[i].b>A[i -1].b

D. A[i].b<A[i -1].b

4)④处应填( )

A. q+1<n &&A[q+1].b<=r

B. q+1<n &&A[q+1].a<=r

C. q<n&&A[q].a <=r

D.q<n &&A[q].b<=r

5)⑤处应填( )

A. r =max(r,A[q +1].a) B. r =max(r,A[q].b)

C. r =max(r,A[q+1].b) D.q++

1) ①处应填 C.A[j].a<A[j-1].a

2) ②处应填 C.A[j]=A[j-1];A[j-1]=t;

3) ③处应填 C.C. A[i].b>A[p -1].b

4) ④处应填B. q+1<n &&A[q+1].a<=r。

5)⑤处应填 B. r =max(r,A[q].b)

本程序旨在读取一系列整数，然后统计并输出这些整数中正数、负数和零的个数，请完善下列程序

#include <iostream>using namespace std;

int main() {

int num;

int positiveCount = 0;

int negativeCount = 0;

int zeroCount = 0;

cout << "请输入一系列整数（输入 -1 结束）：" << endl;

while (true) {

cin >> num;

if (num == -1) {

break;

} else if (num > 0) {

positiveCount++;

} else if (num < 0) {

negativeCount++;

} else {

zeroCount++;

}

}

cout << "正数的个数：" << positiveCount << endl;

cout << "负数的个数：" << negativeCount << endl;

cout << "零的个数：" << zeroCount << endl;

return 0;}