一、选择题（每题2分，共计30分）

1、若一台计算机的字长为32位，则表明该机器（C）。

A、能处理的数值最大为4位十进制数

B、能处理的数值最多为4个字节

C、在CPU中能够作为一个整数加以处理的二进制数据为4个字节

D、在CPU中运算的结果最大为2^32

2、已知8个数据元素为（26，75，15，23，14，62，72，19），按照依次插入结点的方法生成一棵二叉排序树，则该树的深度为（B），注树根深度为1。

A、3 B、4 C、5 D、6

解析：第一个树（树根）的深度为1。二叉排序树的做法是递归判断，小的插到左子树，大的插到右子树

3、一棵完全二叉树的结点总数为18，其叶节点数为（C）。

A、7 B、8 C、9 D、10

4、用1个或多个互不相同的正整数之和表示1-511之间的所有整数，至少要（B）个不同的正整数。

A、6 B、9 C、15 D、24

解析：二进制拆分，2^8<511<2^9

5、一个盒子里有20个大小形状相同的小球，其中5个红的，5个黄的，10个绿的，从盒子中任取一球，若它不是红球，则它是绿球的概率是（C）。

A.5/6 B.3/4 C.2/3 D.1/3

解析：条件概率。P(A|B)=P(AB)/P(B)

6、下列各数中最大的是（D）

A、11010110.0101（二进制） B、D6.53（十六进制）

C、214.32（十进制） D、326.25（八进制）

解析：小数的进制转换，n转10进制：按权展开相加，10转n：乘n取整

7、由a,b,c三种不同的数字组成一个7位数，要求不出现两个a相邻，也不出现两个b相邻，这样的7位数的个数为（ ）。C

A.349 B.512 C.577 D.618

解析：方法1：可以画小数据的搜索树，根据树的形态发现规律为An=2\*An-1+An-2

方法2：设dp[i][j](j[1,3])表示第i位选择j(a,b,c中的一个)的方案数，则有dp[i][1]=dp[i-1][2]+dp[i-1][3],dp[i][2]=dp[i-1][1]+dp[i-1][3],dp[i][3]=dp[i-1][1]+dp[i-1][2]+dp[i-1][3]

8、下列做法可行的是（D）。

A、NOI考试中带水和食物进入考场。

B、在CSP复赛正式开考前，建议敲几段代码熟悉一下键盘和鼠标。

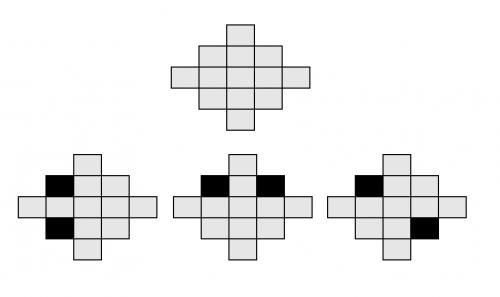
C、在NOI2021比赛中使用C语言

D、在CSP复赛中使用#include<bits/stdc++.h>头文件

9、下列不属于内存储器（A）。

1. 硬盘 B、RAM C、ROM D、Cache

10、给定如图所示的正方形（n=3）。我们可以在这个正方形的某个格子上放置大炮。但是，如果大炮在同一行或者同一列，他们就会互相攻击到。大炮数量为2，请你求出可能的放置方案总数，使得大炮互相不会攻击到。现在告诉你n=3时，总数为46，则n=4时，总数为\_\_\_\_\_\_\_。C

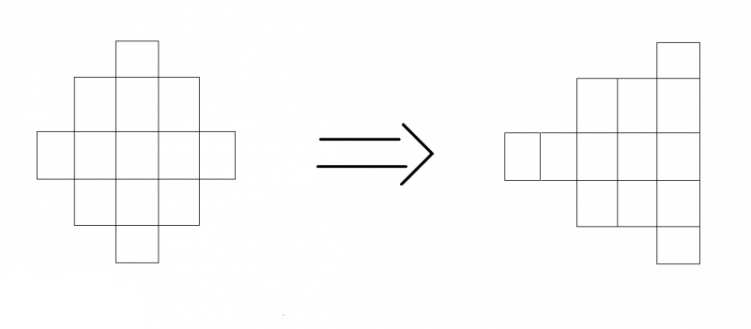


如图所示：上面为n=3的正方形，下面三幅图，前两幅可以互相攻击到，第三幅无法互相攻击到。

A、127 B、157 C、206 D、248

解析：首先注意n=4的正方形不要画错（长为4宽为4）。

考虑到限制条件为不同行不同列，那么我们可以做一个等价变换：



把原图变为一个宽度递增的图形，这样做的好处是避免了后效性问题。

设f[i][k]表示前i列放了k个的方案，

有f[i][k]=f[i−1][k]+f[i−1][k−1]∗(lon[i]−(k−1)) lon(i)表示当前列的宽度

11、以下关于图的说法不正确的是（B）。

A、所有顶点的度数之和等于边数的2倍

B、所有顶点的度数之和不一定等于边数的2倍

C、任意一个图一定有偶数个奇点 （总度数一定为偶数，因为是边数的两倍）

D、在有向图中顶点的入度之和等于出度之和

12、下列不属于冯诺依曼计算机模型的核心思想有（D）。

A、采用二进制表示数据和指令 B、采用“存储程序”工作方式

C、计算机硬件有五大部件（运算器、控制器、存储器、输入和输出设备）

D、结构化程序设计方法

13、下列逻辑运算不正确的是（D）

A、A∧(A∨B)=A B、A∨(A∧B)=A

C、A∧(C∨B)=A∧B∨A∧C D、A∨(B∧C)=A∨B∧A∨C

14、下列关于排序中既是就地排序又是稳定的排序是（）：D

就地（原地）排序是指：基本上不需要额外辅助的的空间，允许少量额外的辅助变量进行的排序。

1. 基数排序（O（基数\*长度+n）） B、希尔排序（O（1）空间但不稳定） C、归并排序（O（n）空间） D、冒泡排序

15、下列关于图的说法正确的是（A）：

A、欧拉图的每一个顶点都能作为某条欧拉闭迹的起点

B、一个图有欧拉闭迹当且仅当该图有零个奇点。 （注意没有说是连通图）

C、若一个无向图有奇数条边，则它必然不是二分图。

D、若G是汉密尔顿图，则对G上的汉密尔顿圈C，任意删去n个点，最多可将C划分为n段，反之亦然。

三、阅读程序写结果（共3题，每题10分，共计30分）

1、

1 #include<iostream>

2 #include<iomanip>

3 using namespace std;

4 int d[11]={0};

5 int main(){

6 int n;

7 cin>>n;

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

9 int x;

10 cin>>x;

11 d[x]++;

12 }

13 d[0]=0;

14 for(int i=1;i<=10;i++) d[i]+=d[i-1];

15 for(int i=1;i<=10;i++)

16 if(d[i]!=d[i-1])

17 cout<<i<<' '<<d[i-1]+1<<' ';

18 return 0;

19 }

16、x输入不能为负数。（ ） 是

17、第2行去掉不影响输出结果。（ ） 是

18、输出结果一定非减。 （ ）否

19、if判断中’!=’改为’>’结果不变。 （ ）是

20、输入为：

6

4 5 6 1 2 3

答案为：（ ） B

A．2 2 1 1 3 3 4 4 5 5 6 6

B．1 1 2 2 3 3 4 4 5 5 6 6

C．1 1 3 2 2 3 4 4 5 5 6 6

D．3 1 2 1 3 3 4 4 5 5 6 6

21、输入为：

20

3 4 6 1 7 6 9 4 10 7

6 6 3 3 8 7 9 10 6 7

答案为：（ ）。 D

A．1 1 2 3 4 5 6 7 7 8 9 10 12 16 17 19

B．1 1 2 3 4 5 6 7 7 8 9 10 12 17 16 19

C．1 1 3 2 4 5 6 7 7 12 8 9 16 17 10 19

D．1 1 3 2 4 5 6 7 7 12 8 16 9 17 10 19

2、

#include <iostream>

using namespace std;

int number,ndata,sum;

int data[100];

void solve(int s,int sign,int n){

int i;

for(i=s;i<ndata;i++){

sum+=sign\*(number/n/data[i]);

solve(i+1,-sign,n\*data[i]);

}

}

int main(){

int i;

cin>>number>>ndata;

sum=0;

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

cin>>data[i];

solve(0,1,1);

cout<<sum<<endl;

return 0;

}

22、输入为10 2 3 3时，答案为4。（ ） 否

23、sign不会出现1和-1之外的数字。（ ） 是

24、时间复杂度为O(ndata^2)。 （ ）否

25、任意输入，sum都不为负。 （ ）否

26、输入为：

ndata为6时，for循环总执行次数为（ ） C

A．36

B．57

C．63

D．81

27、输入为：1000 3 5 13 11

答案为：（ ）。 C

A． 256

B． 276

C． 328

D． 388

输入：

输出: 328

解析：递归中等难度题，建议没做出来的同学自己手模重新做一下

3、

1 #include<iostream>

2 using namespace std;

3 int n=12;

4 char ch[]= {'q','A','S','O','R','T','E','X','A','M','P','L','E'};

5 void shift(int k,int n) {

6 char v;

7 int j;

8 v=ch[k];

9 j=k+k;

10 while(j<=n) {

11 if(j<n&&ch[j]<ch[j+1])

12 j++;

13 if(v<ch[j]) {

14 ch[j/2]=ch[j];

15 j\*=2;

16 } else

17 return;

18 ch[j/2]=v;

19 }

20 }

21 void hpsrt() {

22 int k;

23 char tmp;

24 for(k=n/2; k>0; k--)

25 shift(k,n);

26 cout<<"No.1:";

27 for(k=1; k<=n; k++)

28 cout<<ch[k];

29 cout<<' ';

30 for(k=n; k>0; k--) {

31 tmp=ch[1];

32 ch[1]=ch[k];

33 ch[k]=tmp;

34 shift(1,k-1);

35 }

36 }

37 int main() {

38 int k;

39 hpsrt();

40 cout<<"No.2:";

41 for(k=1; k<=n; k++)

42 cout<<ch[k];

43 cout<<endl;

44 return 0;

45 }

28、程序运行过程中不会出现数组下标越界。（ ） 是

if(j<n&&ch[j]<ch[j+1])，k=6时，虽然ch[]数组会出现13，但是if第一个条件就不满足，做不到第二个条件，所以不会出现下标越界

29、j=k+k改为j=k不影响程序结果。（ ） 否

30、若24行’>’改为’>=’，输出中将包含小写字母。 （ ）否

31、若30行’>’改为’>=’，输出中将包含小写字母。 （ ）是

32、输出结果中，No.1:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_为：（ ） C

A．XTSOREAAMPLE

B．MEAAELORSTPX

C．XTORSEAAMPLE

D．AAEELMOPRSTX

33、输出结果中，No.2:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_为：（ ） D

A．XTSOREAAMPLE

B．MEAAELORSTPX

C．XTORSEAAMPLE

D．AAEELMOPRSTX

解析：难题。NO.1手模，NO.2为堆排序

完善程序（共2题，10空，共40分）

1、给出一张n个点m条边的无向图，每个点的初始状态都为0。

你可以操作任意一个点，操作结束后所有相邻的端点的状态都会改变，由0变成1或由1变成 0。你需要求出最少的操作次数，使得在所有操作完成之后所有n个点的状态都是1。

(1≤n≤35,1≤m≤595,1≤a,b≤n)

洛谷2962

#include<bits/stdc++.h>

using namespace std;

map<long long,int>mp;

int n,m;

int mid,flag,minn=0x7f7f7f7f;

long long ed,q[40],p[40];

void dfs(int x,long long now,int step) {

if(x==mid+1) {

if(now==ed)

minn=min(step,minn);

if(!flag) {

int tmp=mp[now];

if( 1 )

mp[now]=step;

} else {

int tmp= 2 ;

if(!tmp) return ;

minn=min( 3 ,minn);

}

return ;

}

dfs(x+1,now,step);

dfs(x+1, 4 ,step+1);

}

int main() {

p[1]=1;

for(int i=2; i<=40; i++)

p[i]=p[i-1]<<1;

scanf("%d%d",&n,&m);

ed=p[n+1]-1;

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

int x,y;

scanf("%d%d",&x,&y);

q[x]+=p[y];

q[y]+=p[x];

}

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

5 ;

mid=n/2;

dfs(1,0,0);

flag=1;

mid=n;

dfs(n/2+1,0,0);

cout<<minn;

return 0;

}

34、1处填（ ）。C

A．!tmp B．tmp>step

C. !tmp||tmp>step D．!tmp&&tmp>step

35、2处填（ ）。A

A．mp[ed-now] B．mp[ed+now]

C. mp[ed|now] D．mp[ed&now]

36、3处填（ ）。D

A．tmp B． step-tmp

C. tmp+step D．tmp+step

37、4处填（ ）。B

A．q[x] B．now^q[x]

C. now+q[x] D．now\*q[x]

38、5处填（ ）。A、D都可以

A. q[i]+=p[i] B．q[i]-=p[i]

C. q[i]=p[i] D．q[i]^=p[i]

2、高精度除法压位优化：

#include<bits/stdc++.h>

using namespace std;

const int MAXN=1e5;

const int siz=8;

const long long MOD=1e8;//MOD=10^si

char ch1[MAXN],ch2[MAXN];

long long n;

long long a[MAXN>>2],b[MAXN>>2],s[MAXN>>2];

long long cp[MAXN>>2],lt[MAXN>>2],wsd[MAXN>>2];

void write(long long num[]) {

printf("%lld",num[num[0]]);

for(int i=num[0]-1; i; --i) printf("%08lld",num[i]);

puts("");

}

void clear(long long num[]) {

for(int i=num[0]; i; --i) num[i]=0;

num[0]=1;

}

void ry(long long num[]) {

for(int i=num[0]; i; --i) {

if( 1 ) num[i-1]+=MOD;//将1借到下一位

num[i]>>=1;

}

if(!num[num[0]]&&num[0]>1) --num[0];

}

void ly(long long num[]) {

++num[0];

for(int i=1; i<=num[0]; ++i) {

num[i]<<=1;

if(num[i-1]>=MOD) num[i-1]-=MOD,++num[i];

}

if(!num[num[0]]&&num[0]>1) --num[0];

return;

}

void cpy(long long num1[],long long num2[]) {

for(int i=num1[0]; i>num2[0]; --i) num1[i]=0;

for(int i=0; i<=num2[0]; ++i) num1[i]=num2[i];

}

int cmp(long long num1[],long long num2[]) {

if(num1[0]>num2[0]) return 1;

if(num1[0]<num2[0]) return -1;

for(int i=num1[0]; i; --i) {

if(num1[i]>num2[i]) return 1;

if(num1[i]<num2[i]) return -1;

}

return 0;

}

void init() {

scanf("%s%s",ch1,ch2);

int l1=strlen(ch1),l2=strlen(ch2);

for(int i=l1-1; i>=0; i-=siz) {

long long pw=1;

++a[0];

for(int j=i; j>i-siz&&j>=0; --j) {

a[a[0]]+=(ch1[j]^48)\*pw;

pw=(pw<<3)+(pw<<1);

}

}

for(int i=l2-1; i>=0; i-=siz) {

long long pw=1;

++b[0];

for(int j=i; j>i-siz&&j>=0; --j) {

b[b[0]]+=(ch2[j]^48)\*pw;

pw=(pw<<3)+(pw<<1);

}

}

return;//反序读入存储

}

void pls(long long a[],long long b[]) {

clear(s);

s[0]=max(a[0],b[0])+1;

for(int i=1; i<=s[0]; ++i) {

s[i]+=a[i]+b[i];

if(s[i]>=MOD) s[i]-=MOD,++s[i+1];//进位

}

if(!s[s[0]]&&s[0]>1) --s[0];

return;

}

void mnu(long long a[],long long b[]) {

clear(s);

s[0]=max(a[0],b[0]);

for(int i=1; i<=s[0]; ++i) {

s[i]+=a[i]-b[i];

if(s[i]<0) s[i]+=MOD,--s[i+1];//借位

}

while(!s[s[0]]&&s[0]>1) --s[0];

return;

}

void div(long long a[],long long b[]) {

clear(cp),cp[1]=1;

clear(lt);

while(cmp(a,b)!=-1) 2 ;

while(cp[0]>1||cp[1]) {

if(cmp(a,b)!=-1) {

mnu(a,b), 3 ;

4 ;

}

5 ;

}

cpy(s,lt),cpy(lt,a);

return;

}

int main() {

init();

clear(s);

div(a,b);

write(s);

write(lt);

return 0;

}

39、1处填（ ）。B

A．i>1 B．(num[i]&1)&&i>1

C. num[i]&1 D．(num[i]|1)&&i>1

40、2处填（ ）。B

A．ly(a),ly(lt) B．ly(b),ly(cp)

C. ly(lt) D．ly(b),ly(s)

41、3处填（ ）。D

A．cpy(b,s) B．cpy(s,a)

C. cpy(a,b) D．cpy(a,s)

42、4处填（ ）。C

A．mnu(lt,cp),cpy(lt,s) B．cpy(lt,s)

C. pls(lt,cp),cpy(lt,s) D．pls(s,cp),cpy(a,s)

43、5处填（ ）。A

A. ry(b),ry(cp) B．ry(a),ly(cp)

C. ly(a),ly(cp) D．ry(b),ly(cp)