第二十五届全国青少年信息学奥林匹克联赛初赛

提高组 C++语言模拟试题

竞赛时间:2019 年 10 月 14 日 18:00～20:00

选手注意:

- 试题纸共有 10 页，答题纸共有 2 页，满分 100 分。请在答题纸上作答，写在试题纸上的一律无效。
- 不得使用任何电子设备(如计算器、手机、电子词典等)或查阅任何书籍资料。

一、单项选择题(共 15 题，每题 2 分，共计 30 分；每题有且仅有一个正确选项)

1.以下关于 CSP-J/S 的描述错误的是（）

A.任何人都可以自愿报名参加 CSP-J/S

B.CSP-J/S 是 CCF 独立主办的认证，和任何其他机构主办的等级考试无关

C.CSP-J/S 和 NOIP 有密切关系

D.CSP-J/S 认证成绩优异者，可参加 NOI 省级选拔，省级选拔成绩优异者可参加 NOI

2.-128 的补码表示为()

A.00000000

B.00000001

C.10000000

D.11111111

3. 以下不属于 TCP 拥塞控制算法的是()

A. 慢启动

B.拥塞避免

C.快启动

D.快速重传

4. 以下不是基于 UDP 协议的是()

A.DNS

B.RIP

C.TELNET

D.TFTP

5. 定义如下函数 add_edge 和全局变量:

```
int to[MAX],nxt[MAX],h[MAX],top;
void add_edge(int u,int v){
        to[++top]=v,nxt[top]=h[u],h[u]=top;
```

}

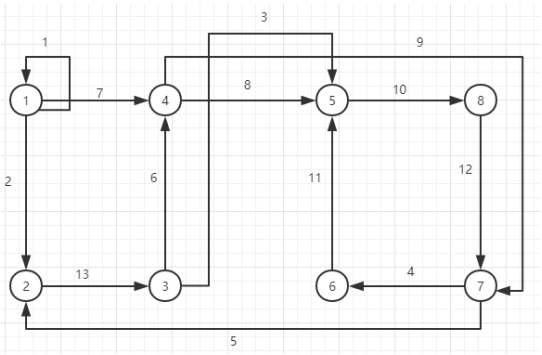
如下图节点编号从 1 开始，按边的编号顺序，以前向星的方式存储，请问 nxt[h[3]]的值为（）

A.6

B.3

C.8

D.7



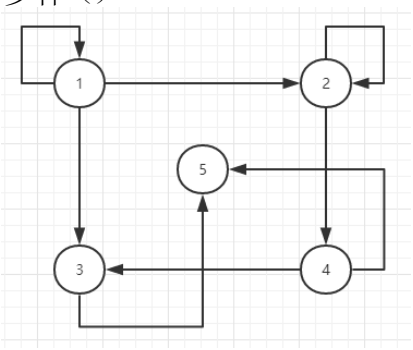
6. 如下图所示，从节点 1 走 6 步走到节点 5 的方案数有多少种（）

A.5

B.8

C.7

D.6



7.同时查找 2n 个数中的最大值和最小值，最少比较次数为( )。

A.3(n-2)/2

B.3n-2

C.4n-2

D.2n-2

8.设 A 和 B 是两个长为 n 的有序数组，现在需要将 A 和 B 合并成一个排好序的 数组，请问任何以元素比较作为基本运算的归并算法最坏情况下至少要做 ( )次比较。

A.n^2

B.$n\log_2 n$

C.2n

D.2n-1

9.G 是一个非连通简单无向图，共有 36 条边，则该图至少有( )个顶点

A.10

B.9

C.8

D.7

10.由四个不同的点构成的简单无向连通图的个数是( )

A.32

B.35

C.38

D.31

11.前缀表达式- + * 4 + 2 3 1 5 的值为（）

A.16

B.17

C.19

D.15

12.2+3*(4-(5+6))/7 的逆波兰表达式为（）

A.2 3 4 5 6 - + * 7 / +

B.2 3 4 5 6 - + * / 7 +

C.2 3 4 5 6 + - * 7 / +

D.2 3 4 5 6 + + * / 7 -

13.若某算法的计算时间表示为递推关系：

$$T(n) = 2.5T(2n/5) + n\log_2^2 n$$

则该算法的复杂度为()

A. $O(n)$

B. $O(n\log_2 n)$

C. $O(n\log_2^2 n)$

D. $O(n\log_2^3 n)$

14.若某算法的计算时间表示为递推关系：

$$T(n) = 3T(n/4) + n\log_2 n$$

则该算法的复杂度为()

A. $O(n)$

B. $O(n\log_2 n)$

C. $O(n\log_2^2 n)$

D. $O(n\log_2^3 n)$

15. 现有变量 a,b,c,d，取值范围均为[0,15]，假设每个值出现的概率相同，则表达式 $a \oplus b \oplus c \oplus d$ 的值能被 3 整除的概率（ ）(⊕ 为计算机中的异或运算符，结果用分数形式表达)

A. 3/8

B.1/2

C.1/4

D.1/8

二、阅读程序写结果(共 4 题，每题 10 分，共计 40 分)

1.

```
#include<iostream>
using namespace std;
int a,b,c;

int* cal(int *p,int &q,int r){
    q+=r;
    *p+=q;
    return p;
}

int main(){
    cin>>a>>b>>c;
    c=*cal(&a,b,c);
    cout<<a<<" "<<b<<" "<<c;
}
```

1.1 cal 函数中参数 p 使用指针传递，q 和 r 则是值传递

A.正确

B.错误

1.2 cal 函数返回一个指向 int 类型存储空间的地址

A.正确

B.错误

1.3 当输入 1 2 3 时，程序输出结果为（）

A.6 2 3

B.6 5 3

C.6 5 6

D.1 2 6

1.4 当输入 23 45 11 时，程序的输出结果为（）

A.79 56 11

B.79 56 79

C.44 56 79

D.79 56 44

2.

```
#include<iostream>
#include<cmath>
#define MAX 1000
#define p sqrt(3)
using namespace std;
```

```
int n,dp[1000][3];
int h0=1,h1=3;
double ans1=(2+p)/(2*p),ans2=(-2+p)/(2*p);
int main(){
    cin>>n;
    dp[1][0]=dp[1][1]=dp[1][2]=1;
    for(int i=2,tmp;i<=n;i++){
        dp[i][0]=dp[i-1][1]+dp[i-1][2];
        dp[i][1]=dp[i-1][0]+dp[i-1][1]+dp[i-1][2];
        dp[i][2]=dp[i-1][0]+dp[i-1][1]+dp[i-1][2];
        tmp=h1;
        h1=2*(h1+h0);
        h0=tmp;
    }
    for(int i=1;i<=n;i++){
        ans1=ans1*(1+p);
        ans2=ans2*(1-p);
    }
    cout<<h1<<endl;
    cout<<dp[n][0]+dp[n][1]+dp[n][2]<<endl;
    cout<<ans1+ans2<<endl;
}
```

2.1 上述程序的输出中 h1 和 dp[n][0]+dp[n][1]+dp[n][2]的值相等

A.正确

B.错误

2.2 上述程序的输出中 dp[n][0]+dp[n][1]+dp[n][2]和 ans1+ans2 的值相等

A.正确

B.错误

2.3 当 n 等于 5 时，第一行输出（即 h1）结果为（ ）

A.164

B.60

C.448

D.128

2.4 当 n 等于 10 时，第三行输出（即 ans1+ans2）结果为（）

A.9136

B.68192

C.24960

D.3344

3.

```
#include<iostream>
#include<cstring>
#define LL long long
using namespace std;
LL l,r;
LL f[12][10][10][2][2][2],a[20];
LL Dfs(LL now,LL p,LL pp,LL _4,LL _8,LL top,LL hw){
    if(_4&&_8) return 0;
    if(!now) return hw;
    if(!top && f[now][p][pp][_4][_8][hw]!=-1)
    return f[now][p][pp][_4][_8][hw];
    LL Up=top?a[now]:9;
    LL ret(0);
    for(LL i=0;i<=Up;++i)
      ret+=Dfs(now-1,i,p,_4|(i==4),_8|(i==8),
      top&&(i==Up) ,hw|(i==pp&&i==p));
    if(!top) f[now][p][pp][_4][_8][hw]=ret;
    return ret;
}
inline LL Solve(LL x){
    LL tot(0);
    while(x){
        a[++tot]=x%10;
        x/=10;
    }
    if(tot!=11) return 0;
    LL ret(0);
    for(LL i=1;i<=a[tot];++i)
        ret+=Dfs(tot-1,i,0,(i==4),(i==8),i==a[tot],0);
    return ret;
}
int main(){
    cin>>l>>r;
    memset(f,-1,sizeof(f));
    cout<<Solve(r)-Solve(l-1);
    return 0;
}
```

3.1 同时包含 4 和 8 的数字都不会被统计

A.正确

B.错误

3.2 相邻数位中，超过 3 个数位相同的数字都不会被统计

A.正确

B.错误

3.3 下列哪个是合法（会被统计）的数字（）

A.2323234823

B.1015400080

C.23333333333

D.10010012022

3.4 当输入 12121284000 12121285550 时，程序输出结果为（）

A.5

B.457

C.455

D.6

4.

```
#include <iostream>
#include <string>

using namespace std;

size_t equalizeLength(string &s1, string &s2)
{
    size_t len1 = s1.size(), len2 = s2.size();
    if (len1 < len2)
    {
        for (int i = 0; i < len2 - len1; ++i)
            s1 = '0' + s1;
        return len2;
    }
    else if (len1 > len2)
    {
        for (int i = 0; i < len1 - len2; ++i)
            s2 = '0' + s2;
    }
    return len1;
}

string strAddition(string s1, string s2)
{
    string ret;
    int carry = 0;
    size_t len = equalizeLength(s1, s2);

    for (int i = len - 1; i >= 0; --i)
    {
        int firstBit = s1.at(i) - '0';
```

```
        int secondBit = s2.at(i) - '0';

        int sum = (firstBit ^ secondBit ^ carry) + '0';
        ret = static_cast<char>(sum) + ret;

        carry = (firstBit & secondBit) | (firstBit & carry) | (secondBit
& carry);
    }
    if (carry)
        ret = '1' + ret;
    return ret;
}

long int Karatsuba(string s1, string s2)
{
    size_t len = equalizeLength(s1, s2);

    // base case
    if (len == 0) return 0;
    if (len == 1) return (s1[0] - '0') * (s2[0] - '0');

    size_t floor = len / 2;
    size_t ceil = len - floor;
    string a = s1.substr(0, floor);
    string b = s1.substr(floor, ceil);
    string c = s2.substr(0, floor);
    string d = s2.substr(floor, ceil);

    long int p1 = Karatsuba(a, c);
    long int p2 = Karatsuba(b, d);
    long int p3 = Karatsuba(strAddition(a, b), strAddition(c, d));
    return (1<<(2 * ceil)) * p1 + (1<<(ceil)) * (p3 - p1 - p2) + p2;
}

int main() {
    string s1,s2;
    cin>>s1>>s2;
    cout <<Karatsuba(s1, s2) << endl;
    return 0;
}
```

4.1 上述程序实现大整数加法

A.正确

B.错误

4.2 上述程序的算法复杂度大于 $O(n\log n)$（其中 n 为 max(s1.length(),s2.length())）

A.正确
B.错误

4.3 当输入 111 011 时程序输出为（）
A.10
B.4
C.21
D.2

4.4 当输入 10101 101010 时程序输出为（）
A.441
B.882
C.1764
D.220

五、完善程序(每题 15 分，共计 30 分)

1.（链表反转）单向链表反转是一道经典算法问题，比如有一个链表是这样的，1->2->3->4->5，反转后成为 5->4->3->2->1。现给定如下链表节点的定义：

```
struct LinkNode{
    int value;
    LinkNode* next;};
```

非递归实现：

```
LinkNode* Reverse(LinkNode* header){
    if (header == NULL || header->next == NULL){
        return header;
    }

    LinkNode* pre = header, *cur = header->next;
    pre->next = NULL;
    while(cur != NULL)
    {
        LinkNode* next = ____1____;
        ___2____ = pre;
        pre = cur;
        cur = next;
    }
    return pre;}
```

递归实现：

```
LinkNode * Reverse(LinkNode * head){
    if (head == NULL || head->next == NULL){
        return head;
    }
    LinkNode * newhead = ___3____;
    ___4_____ = head;
    head->next = ___5____;
    return newhead;
}
```

1.1 上述程序___1___中应该填写（）

A.pre-> next
B.cur-> next
C.header-> next
D.NULL

1.2 上述程序___2___中应该填写（）

A.pre-> next
B.cur-> next
C.header-> next
D.NULL

1.3 上述程序___3___中应该填写（）

A.ReverseList(head)
B.ReverseList(pre)
C.ReverseList(cur)
D.ReverseList(head->next)

1.4 上述程序___4___中应该填写（）

A.pre-> next->next
B.cur-> next->next
C.header-> next->next
D.NULL

1.5 上述程序___5___中应该填写（）

A.pre-> next
B.cur-> next
C.header-> next
D.NULL

2.(最小环问题)给定一张无向图，求图中一个至少包含 3 个点的环，环上的节点不重复，并且环上的边的长度之和最小。该问题称为无向图的最小环问题。在本题中，你需要输出最小环的方案，若最小环不唯一，输出任意一个均可。若无解，输出 No solution.
图的节点数不超过 100100。
输入：

第一行两个正整数 n,m 表示点数和边数。
接下来 m 行，每行三个正整数 x,y,z，表示节点 x,y 之间有一条长度为 z 的边。
输出：
一个最小环的方案：按环上顺序输出最小环上的点。若最小环不唯一，输出任意一个均可。若无解，输出 No solution.

```
#include <bits/stdc++.h>
#define MAXN 105
#define INF 0x3f3f3f3f
using namespace std;
inline int read(){
    int x=0,f=1;
    char ch=getchar();
    while (ch<'0'||ch>'9'){
        if (ch=='-') f=-1;
        ch=getchar();
    }
    while (ch>='0'&&ch<='9'){
        x=(x<<3)+(x<<1)+(ch^'0');
        ch=getchar();
    }
    return x*f;}
static int stk[MAXN],top;
static int pos[MAXN][MAXN];//表示 i~j 的中点节点
#define Push(x) stk[++top]=(x);
void GetAns(int i,int j){
    if (pos[i][j]==0) return ;
    GetAns(i,____1____);
    Push(pos[i][j]);
    GetAns(pos[i][j],____2____);}
static int G[MAXN][MAXN],D[MAXN][MAXN];
int main(){
    int n=read(),m=read();
    memset(G,0x3f,sizeof(G));
    memset(D,0x3f,sizeof(D));
    for (register int i=1;i<=m;++i){
        int u=read(),v=read();
        D[v][u]=D[u][v]=G[u][v]=G[v][u]=min(G[u][v],read());
    }
    int ans=INF;
    for (register int k=1;k<=n;++k){
        for (register int i=1;i<k;++i){
            for (register int j=i+1;j<k;++j){
                if (D[i][j]==INF||G[j][k]==INF||G[k][i]==INF) continue;
                if (D[i][j]+G[j][k]+G[k][i]<ans){
```

```
                ans=_____3_____;
                top=0;Push(i);GetAns(i,j);Push(j);Push(k);
            }
        }
    }
    for (register int i=1;i<=n;++i){
        for (register int j=1;j<=n;++j){
            if (_____4_____){
                D[i][j]=D[i][k]+D[k][j];
                pos[i][j]=k;
            }
        }
    }
}
if (ans==INF) return puts("No solution."),0;
for (register int i=1;i<=top;++i) printf("%d ",_____5____);}
```

1.1 上述程序___1___中应该填写（）

A.j

B.pos[i][j]

C.i

D.pos[j][i]

1.2 上述程序___2___中应该填写（）

A.j

B.pos[i][j]

C.i

D.pos[j][i]

1.3 上述程序___3___中应该填写（）

A.D[i][j]+G[k][j]+G[i][k]

B.D[i][j]+G[j][k]+G[k][i]

C.D[i][k]+G[k][j]+G[i][j]

D.D[i][j]+G[j][i]+G[i][k]

1.4 上述程序___4___中应该填写（）

A.D[k][j]>D[i][k]+D[k][j]

B.D[i][j]>D[i][k]+D[k][j]

C.D[i][j]<D[i][k]+D[k][j]

D.D[i][k]>D[i][k]+D[k][j]

1.5 上述程序___5___中应该填写（）

A.pos[i][i]

B.stk[i]

C.pos[1][i]

D.pos[i][1]