

第二十五届全国青少年信息学奥林匹克联赛初赛

CSP-S C++语言模拟试题

竞赛时间:2019 年 10 月 13 日 8:30~10:30

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. CSP-J/S 允许以团体为单位进行报名
- B. CSP-J/S 不和任何行政机构挂钩
- C. CSP-J/S 和 NOIP 没有关系
- D. 不能同时取得 CSP-J 和 CSP-S 认证, 因为第二轮时间冲突

2. $(11111110)_2 + (11111011)_2$ 计算结果为 ()

- A. -7
- B. -1
- C. 7
- D. 1

3. $(BF)_{16} + (567)_8$ 计算结果为 ()

- A. $(1060)_8$
- B. $(1000110110)_2$
- C. $(565)_{10}$
- D. $(235)_{16}$

4. 内存地址的最重要特点是 ()

- A. 随机性
- B. 顺序性
- C. 连续性
- D. 唯一性

5. 我国第一台电子计算机于那一年试制成功():

- A. 1953 B. 1955 C. 1958 D. 1949

6. 硬盘工作时应特别注意避免 ()

- A. 噪声 B. 潮湿 C. 震动 D. 日光

7. 以下逻辑表达式的值恒为真的是()。

- A. $Q \vee (\neg P \wedge Q) \vee (P \wedge \neg Q)$
B. $P \vee Q \vee (P \wedge \neg Q) \vee (\neg P \wedge Q)$
C. $P \vee \neg Q \vee (P \wedge \neg Q) \vee (\neg P \wedge \neg Q)$
D. $P \vee (\neg P \wedge Q) \vee (\neg P \wedge \neg Q)$

8. 前缀表达式 $+3*2+5\ 12$ 的值为()。

- A. 23 B. 65 C. 25 D. 37

9. 已知一颗二叉树的前序遍历为 ABDHIEJKCF LMGN O, 中序遍历为 HDIBJEKALFMCNGO。以下描述错误的是()

- A. 节点 M 在节点 A 的右子树中
B. 节点 E 的父亲是节点 B
C. 节点 O 的兄弟节点为 N
D. 节点 E 的兄弟节点为 F

10. 已知一颗二叉树的中序遍历为 EBG FADC, 后续遍历为 EGFBDCA, 前序遍历为()

- A. ABEFGCD
B. ACDBEFG
C. ABFGECD
D. ABEGFDC

11. 现有多重集合 $T = \{ 3*a, 4*b, 5*c \}$, 求取 10 个组合在一起有多少不同的方法 ()

A.16 B.10 C.6 D.7

12. 12 个高矮不同的人，排成两排，每排必须是从矮到高排列，而且第二排比对应的第一排的人高，问排列方式有多少种（）

A.429
B.97
C.132
D.198

13. 若某算法的计算时间表示为递推关系：

$$T(n) = 4T(n/2) + n$$

则该算法的复杂度为（）

A. $O(n^2)$ B. $O(n)$ C. $O(\log_2 n)$ D. $O(n \log_2 n)$

14. 若某算法的计算时间表示为递推关系：

$$T(n) = 2T(\sqrt{n}) + \log_2 n$$

则该算法的复杂度为（）

A. $O(n)$ B. $O(n \log n)$ C. $O(n \log \log n)$ D. $O(\log n \log \log n)$

15. 一棵完全二叉树共有 699 个节点，则在二叉树中叶子节点数（）

A. 300 B. 350 C. 250 D. 275

二、阅读程序写结果 (共 4 大题，无特殊说明选择题 3 分，判断对错 1.5 分，共计 40 分)

1.

```

#include <iostream>
#define MAX 10000
#define ll long long
using namespace std;
ll a[MAX],n,m,flag;
int main() {
    cin>>n>>m;
    if(n<m) swap(n,m);
    for(int i=0;i<n;i++) cin>>a[i];
    for(int i=0;i<n;i++){
        for(int j=i,ans=0;j<n;j++,ans=0){
            for(int k=i;k<=j;k++){
                ans+=a[k];
            }
            if(ans%m==0) flag=1;
        }
    }
    cout<<(flag?"YES":"NO")<<endl;
}

```

1.1 上述程序即可能输出 NO，也可能输出 YES

- A. 正确
- B. 错误

1.2 flag 的初始值为 0

- A. 正确
- B. 错误

1.3 当输入为

12 104857601

1 2 3 4 5 6 8 9 10 11 12

时，程序输出结果为（）

A.YES

B.NO

C.没有输出结果

D.runtime error

1.4 当输入为

20 19

32145 13665 10121 8952 7907 29868 6692 27381 21442 27343 18367 18263 20291 1431 4549 21358 16583 29428 31284 9680

时，程序的输出结果为（）（4分）

A.YES

B.NO

C.没有输出结果

D.runtime error

2.

```
#include<iostream>
```

```
#include<cstdio>
```

```
#include<cstdlib>
```

```
#include<algorithm>
```

```
#include<cmath>
```

```
#include<cstring>
```

```
#define ll long long
```

```
using namespace std;
```

```

inline ll read()
{
    ll 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*10+ch-'0';ch=getchar();}
    return x*f;
}

struct data{ll a[10];};
ll a,b,t[25];
data f[25][10];
data operator+(data a,data b)
{
    data t;
    for(int k=0;k<=9;k++)
        t.a[k]=a.a[k]+b.a[k];
    return t;
}

data cal(ll x)
{
    data ans;for(int i=0;i<=9;i++)ans.a[i]=0;
    if(!x)
    {
        ans.a[0]=1;
        return ans;
    }
    int len=15;
    while(t[len]>x)len--;
    for(int i=1;i<len;i++)

```

```

        for(int j=1;j<=9;j++)
            ans=ans+f[i][j];
    ans.a[0]++;
    int cur=x/t[len];
    for(int i=1;i<cur;i++)
        ans=ans+f[len][i];
    x%=t[len];
    ans.a[cur]+=x+1;
    for(int i=len-1;i;i--)
    {
        cur=x/t[i];
        for(int j=0;j<cur;j++)
            ans=ans+f[i][j];
        x%=t[i];
        ans.a[cur]+=x+1;
    }
    return ans;
}
int main()
{
    t[1]=1;for(int i=2;i<=15;i++)t[i]=t[i-1]*10;
    for(int i=0;i<=9;i++)f[1][i].a[i]=1;
    for(int i=2;i<=12;i++)
        for(int x=0;x<=9;x++)
            for(int y=0;y<=9;y++)
            {
                f[i][y]=f[i][y]+f[i-1][x];
                f[i][y].a[y]+=t[i-1];
            }
}

```

```

    }
    a=read();b=read();
    data t1=cal(b),t2=cal(a-1);
    for(int i=0;i<=9;i++)
    {
        printf("%lld",t1.a[i]-t2.a[i]);
        if(i!=9)printf(" ");
    }
    return 0;
}

```

2.1 上述程序中，预处理后 $t[i]$ 存储 10^i

A. 正确

B. 错误

2.2 上述程序重载了结构体 data 的+运算符

A. 正确

B. 错误

2.3 当输入为 10 45 时，程序输出结果为 ()

A. 4 4 14 4 10 4 3 3 3 3

B. 4 14 14 14 10 4 3 4 3 4

C. 4 14 14 14 10 4 3 3 3 3

D. 4 14 10 14 10 4 3 3 3 3

2.4 当输入为 44 121 时，程序输出结果为 () (4 分)

A. 18 40 9 7 14 18 18 18 18

B. 18 40 9 7 14 18 14 18 14 18

C.18 40 9 7 14 7 14 7 14 7
D.18 14 9 7 14 18 18 18 18 18

```
3.
#include<iostream>
#include<cstdio>
#include<cstring>
#define inf 1000000000
using namespace std;
char ch1[10005],ch2[10005];
int la,lb,cnt;
struct data{int a[1205],l;}a,b;
bool com()
{
    if(a.l<b.l)return 0;
    if(a.l>b.l)return 1;
    for(int i=a.l;i>0;--i)
        if(a.a[i]>b.a[i])return 1;
        else if(a.a[i]<b.a[i])return 0;
    return 1;
}
void print(data a)
{
    while(a.a[a.l]==0)a.l--;
    for(int i=a.l;i>0;--i)
        if(i==a.l)printf("%d",a.a[i]);
```

```

        else printf("%09d",a.a[i]);
    }
inline data sub(data a,data b)
{
    int k;
    data c;
    for(int i=1;i<=1200;++i)
    {
        if(i<=b.l)c.a[i]=a.a[i]-b.a[i];
        else if(i<=a.l)c.a[i]=a.a[i];
        else c.a[i]=0;
        if(c.a[i]<0)
        {
            c.a[i]+=inf;
            a.a[i+1]--;
        }
    }
    c.l=a.l;
    while(c.a[c.l]==0&& c.l) c.l--;
    return c;
}
void diva()
{
    for(int i=1;i<=a.l;i++)
    {
        if(a.a[i]&1)a.a[i-1]+=inf/2;
        a.a[i]>>=1;
    }
}

```

```

        if(!a.a[a.l])a.l--;
    }
void divb()
{
    for(int i=1;i<=b.l;i++)
    {
        if(b.a[i]&1)b.a[i-1]+=inf/2;
        b.a[i]>>=1;
    }
    if(!b.a[b.l])b.l--;
}
void mul()
{
    for(int i=a.l;i>0;i--)
    {
        a.a[i]<<=1;
        a.a[i+1]+=a.a[i]/inf;
        a.a[i]%=inf;
    }
    while(a.a[a.l]>0)a.l++;
    for(int i=b.l;i>0;i--)
    {
        b.a[i]<<=1;
        b.a[i+1]+=b.a[i]/inf;
        b.a[i]%=inf;
    }
    while(b.a[b.l]>0)b.l++;
}

```

```

int main()
{
    scanf("%s%s",ch1+1,ch2+1);
    la=strlen(ch1+1);lb=strlen(ch2+1);
    if(la%9)a.l=la/9+1;
    else a.l=la/9;
    if(lb%9)b.l=lb/9+1;
    else b.l=lb/9;
    for(int i=1;i<=a.l;++i)
    {
        int k1=max(1,la-i*9+1),k2=la-(i-1)*9;
        for(int j=k1;j<=k2;++j)
            a.a[i]=a.a[i]*10+ch1[j]-'0';
    }
    for(int i=1;i<=b.l;++i)
    {
        int k1=max(1,lb-i*9+1),k2=lb-(i-1)*9;
        for(int j=k1;j<=k2;++j)
            b.a[i]=b.a[i]*10+ch2[j]-'0';
    }
    while(1)
    {
        if((a.a[1]%2==0)&&(b.a[1]%2==0)){diva();divb();cnt++;}
        else if((a.a[1]%2==0))diva();
        else if((b.a[1]%2==0))divb();
        if(com()){a=sub(a,b);if(!a.l){while(cnt--)mul();print(b);break;}}
        else {b=sub(b,a);if(!b.l){while(cnt--)mul();print(a);break;}}
    }
}

```

```
    //system("pause");  
    return 0;  
}
```

3.1 上述程序主要功能为计算两个整数的高精度乘法

A. 正确

B. 错误

3.2 如果 $a > b$ ，则 com 函数返回 1

A. 正确

B. 错误

3.3 当输入为 12 54 时，程序输出结果为（）

A. 5

B. 6

C. 7

D. 8

3.4 当输入为 44 121 时，程序输出结果为（）

A. 9

B. 10

C. 11

D. 12

4.

```
#include<iostream>
```

```
#include<cstring>
```

```
#include<cstdio>
```

```

#include<cstdlib>
#include<algorithm>
#include<queue>
#include<cmath>
#include<map>
#include<queue>
#define ll unsigned long long
#define inf 2000000000
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*10+ch-'0';ch=getchar();}
    return x*f;
}
int ans,n,m,K;
char ch[205];
ll H1[30005][205],G1[30005][205];
ll tmp[30005];
void cal(int x)
{
    for(int i=1;i<=m;i++)
    {
        H1[x][i]=(H1[x][i-1]*149+ch[i]);
    }
    for(int i=m;i;i--)
    {

```

```

        G1[x][i]=(G1[x][i+1]*137+ch[i]);
    }
}
int main()
{
    n=read();m=read();K=read();
    for(int i=1;i<=n;i++)
    {
        scanf("%s",ch+1);
        cal(i);
    }
    for(int j=1;j<=m;j++)
    {
        for(int i=1;i<=n;i++)
        {
            tmp[i]=(H1[i][j-1]*233+G1[i][j+1]*213);
        }
        sort(tmp+1,tmp+n+1);
        int now=1;
        for(int i=2;i<=n;i++)
        {
            if(tmp[i]==tmp[i-1])ans+=now,now++;
            else now=1;
        }
    }
    printf("%d\n",ans);
    return 0;
}

```

4.1 执行 `sort(tmp+1,tmp+n+1)`，会对 `tmp[1]...tmp[n]` 从小到大排序（2 分）

A. 正确

B. 错误

4.2 当且仅当这两个字符串等长时，不会被统计（2 分）

A. 正确

B. 错误

4.3 当输入

4 3 64

Fax

fax

max

mac

时程序输出为（）

A. 4

B. 5

C. 6

D. 7

4.4 当输入

10 3 64

YPY

YGY

mhy

zyd

yxc

ljz

zhh

wsb

gzy

yx_

时程序输出为 () (4 分)

A.8

B.9

C.10

D.2

五、完善程序(每题 15 分，共计 30 分)

1. (字符串 LCS) 字符序列的子序列是指从给定字符序列中随意地 (不一定连续) 去掉若干个字符 (可能一个也不去掉) 后所形成的字符序列。令给定的字符序列 $X = \langle x_0, x_1, \dots, x_{m-1} \rangle$, 序列 $Y = \langle y_0, y_1, \dots, y_{k-1} \rangle$ 是 X 的子序列, 存在 X 的一个严格递增下标序列 $\langle i_0, i_1, \dots, i_{k-1} \rangle$, 使得对所有的 $j = 0, 1, \dots, k-1$, 有 $x_{i_j} = y_j$ 。例如, $X = \text{“ABCBDAB”}$, $Y = \text{“BCDB”}$ 是 X 的一个子序列。对给定的两个字符序列, 求出他们最长的公共子序列长度, 以及最长公共子序列个数。

输入

第 1 行为第 1 个字符序列, 都是大写字母组成, 以 “.” 结束。长度小于 5000。

第 2 行为第 2 个字符序列, 都是大写字母组成, 以 “.” 结束, 长度小于 5000。

输出

第 1 行输出上述两个最长公共子序列的长度。

第 2 行输出所有可能出现的最长公共子序列个数, 答案可能很大, 只要将答案对 100,000,000 求余即可。

```
#include <iostream>
```

```
#include <cstdio>
```

```
#include <cstring>
```

```
using namespace std;
```

```
const int mod=100000000;
```

```
const int maxn=5005;
```

```

char a[maxn], b[maxn];
int f[2][maxn], g[2][maxn];
int main()
{
    scanf("%s", a+1); scanf("%s", b+1);
    int n=strlen(a+1)-1, m=strlen(b+1)-1;
    g[1][0]=1;
    for(int j=0; j<=m; j++)
        _____1_____;
    for(int i=1; i<=n; i++)
    {
        int now=i&1, pre=now^1;
        for(int j=1; j<=m; j++)
        {
            f[now][j]=_____2_____;
            if(a[i]==b[j])
            {
                f[now][j]=_____3_____;
                if(f[now][j]==f[pre][j-1]+1)
                    g[now][j]=g[pre][j-1];
            }
            else
            {
                g[now][j]=0;
                if(f[now][j]==f[pre][j-1])
                    g[now][j]-=g[pre][j-1];
            }
            if(f[now][j]==f[pre][j])

```

```

        g[now][j]=_____4_____;

        if(f[now][j]==f[now][j-1])
            g[now][j]=_____5_____;
    }
}
printf("%d\n%d",f[n&1][m],g[n&1][m]);
return 0;
}

```

1.1 上述程序____1____中应该填写（）

- A.g[0][j]=0
- B.g[j][0]=0
- C.g[0][j]=1
- D.g[j][0]=1

1.2 上述程序____2____中应该填写（）

- A.min(f[pre][j-1], f[now][j-1])
- B.max(f[pre][j], f[now][j])
- C.max(f[pre][j], f[now][j-1])
- D.max(f[now][j], f[pre][j-1])

1.3 上述程序____3____中应该填写（）

- A.min(f[pre][j-1], f[now][j-1]+1)
- B.max(f[pre][j], f[now][j]+1)
- C.max(f[pre][j], f[now][j-1]+1)
- D.max(f[now][j], f[pre][j-1]+1)

1.4 上述程序___4___中应该填写 ()

- A. $(g[now][j] + g[pre][j-1]) \% mod$
- B. $(g[now][j-1] + g[pre][j]) \% mod$
- C. $(g[now][j] + g[pre][j]) \% mod$
- D. $(g[now][j-1] + g[pre][j-1]) \% mod$

1.5 上述程序___5___中应该填写 ()

- A. $(g[now][j] + g[pre][j-1]) \% mod$
- B. $(g[now][j-1] + g[pre][j]) \% mod$
- C. $(g[now][j] + g[pre][j]) \% mod$
- D. $(g[now][j-1] + g[pre][j-1]) \% mod$

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]

第二十五届全国青少年信息学奥林匹克联赛初赛

CSP-S模拟卷参考答案

一、单项选择题（共15 题，每题2 分，共计30 分）

1	2	3	4	5	6	7	8
D	A	B	D	C	C	D	D
9	10	11	12	13	14	15	
D	A	C	C	A	D	B	

二、阅读程序写结果（共4题，每题10分，共计40分）

第一题	1.1	1.2	1.3	1.4
	B	A	C	A
第二题	2.1	2.2	2.3	2.4
	B	A	C	A

第三题	3.1	3.2	3.3	3.4
	B	A	B	C
第四题	4.1	4.2	4.3	4.4
	A	B	A	D

三、完善程序（每空3 分，共计30 分）

第一题	1.1	1.2	1.3	1.4	1.5
	C	C	D	C	A
第二题	2.1	2.2	2.3	2.4	2.5
	B	A	B	B	B