CSP-J 第一轮模拟赛 1

一、判断题（每题 1 分，共 10 分，选项 T=正确，F=错误，每空标号处均填选择的答案对应的大写字母）

1. CSP-J/S 非专业级别比专业级别更难（ ）
2. CSP-J/S 非专业认证在校生与在职人员皆可参加（ ）
3. CCF 建议将 CSP-J/S 成绩作为职业晋升和升学的唯一依据（ ）
4. CSP-J 和 CSP-S 皆举办两轮（ ）
5. 可以直接参加 CSP-J/S 第二轮（ ）
6. CSP-J/S 受理因选手个人失误提交的申诉（ ）
7. CSP-J/S 属于 NOI 系列赛事（ ）
8. 可以在 CSP-J/S 赛前发布虚假的“CSP-J/S 试题解析”干扰认证秩序（ ）
9. CSP-J/S 允许选手自带鼠标/键盘（ ）
10. CSP-J/S 认证者可在认证完成离开考场后返回考场（ ）

二、单选题（每题 2 分，共 30 分）

1. 以下编程语言中，同时符合面向对象与编译执行的是（ ）

A. C　　B. C++　　C. Pascal　　D. Python

2. 二进制补码求和：11010010+00000101=( )

A. 11000011　　B. 10110110　　C. 11010111　　D. 10001001

3. 与十进制数 17.5625 相对应的 8 进制数是（ ）

A. 21.5625　　B. 21.44　　C. 21.73　　D. 21.731

4. 下列网络设备中，不属于局域网设备的是（ ）

A. 路由器　　B. 集线器　　C. 网卡　　D. 中继器

5. 以下不属于结构化程序的结构是（ ）

A. 顺序结构　　B. 输入输出结构 C. 分支结构　　D. 循环结构

6. 在待排序文件已基本有序的前提下，下述排序方法中效率最高的是（ ）

A. 插入排序　　B. 选择排序　　C. 快速排序　　D. 归并排序

7. 设循环队列中数组的下标范围是 0～m-1，其头尾指针分别为 f 和 r，则其元素个数为( )

A. r-f　　B. r-f+1　　C. (r-f+1)%m　　D. (r-f+m)%m

8. 现有 A、B、C、D、E、F、G 七个元素依次进栈操作，但可随时出栈，出了栈后不能再进栈，下面哪个是不可能的( )

A. ABCDEFG　　B. ADBCGFE　　C. ABCDEGF　　D. GFEDCBA

9. 关于拓扑排序，下面说法正确的是（ ）

A. 所有连同的有向图都可以实现拓扑排序

B. 对一个图而言，拓扑排序的结果是唯一的

C. 拓扑排序中入度为 0 的结点总会排在入度大于 0 的结点的前面

D. 拓扑排序结果序列中的第一个结点一定是入度为 0 的点

10. 完全二叉树的顺序存储方案，是指将完全二叉树的结点从上至下，从左到右依次存放到一个顺序结构的数组中。假定根结点存放在数组的 1 号位置，则第 k 号结点的父结点如果存在的话，应当存放在数组的( )号位置。

C. 2k　　B. 2k+1　　C. k/2 下取整　　D. (k+1)/2 下取整

11. 以下逻辑表达式的值恒为真的是（ ）。

A. P∨(¬P∧Q)∨(¬P∧¬Q)　　B. Q∨(¬P∧Q)∨(P∧¬Q)

C. P∨Q∨(P∧¬Q)∨(¬P∧Q)　　D. P∨¬Q∨(P∧¬Q)∨(¬P∧¬Q)

12. 表达式（1+34）*5-56/7 的后缀表达式为（ ）

A. 1+34*5-56/7　　B. -*+1 34 5/56 7

C. 1 34+5*56 7/-　　D. 1 34 5*+56 7

E. 1 34+5 56 7-*/

13. 某大学计算机专业的必修课及其先修课程如下表所示：

课程代号	C_0	C_1	C_2	C_3	C_4	C_5	C_6	C_7
课程名称	高等数学	程序设计语言	离散数学	数据结构	编译技术	操作系统	普通物理	计算机原理
先修课程			C_0, C_1	C_1, C_2	C_3	C_3, C_7	C_0	C_6

下列课程安排方案哪个不是合理的（　　）。

A. C0, C1, C2, C3, C6, C7, C5, C4　　B. C0, C1, C2, C3, C4, C6, C7, C5

C. C0, C1, C6, C7, C2, C3, C4, C5　　D. C0, C1, C6, C7, C5, C2, C3, C4

14. 在 n 个结点的顺序表中，算法的时间复杂度是 O(1)的操作是（　　）。

A. 访问第 i 个结点（1≤i≤n）和求第 i 个结点的直接前驱（2≤i≤n）

B. 在第 i 个结点后插入一个新结点（1≤i≤n）

C. 删除第 i 个结点（1≤i≤n）

D. 将 n 个结点从小到大排序

15. 给定一个正整数 N=8934632178，现决定依次删除其中 6 个数位上的数字（每次删除一个数位上的数字），每次删除后按原来的次序组成一个新数 M 的值均是当前状态下的最小数，则第四次应该删除的数字是（　　）。

A、6　　B、8　　C、7　　D、4

三、问题解答（每题 5 分，共 10 分，每空标号处均填选择的答案对应的大写字母）

1. 将 2 个红球，1 个蓝球，1 个白球放到 10 个编号不同的盒子中去，每个盒子最多放一个球。有多少种放法(　　)

A. 5040　　B. 2520　　C. 420　　D. 1260

2. 两个人抛硬币，规定第一个抛出正面的人可以吃到苹果，请问先抛的人能吃到苹果的概率多大（　　）

A. 1/2　　B. 2/3　　C. 3/4　　D. 5/8

四、阅读程序（每题 6 分，共 24 分，每空标号处均填选择的答案对应的大写字母）

1.

```
#include <cstdio>
int main(){
    int i,s,max;
    int a[11];
    for(i=1;i<=10;i++)
        scanf("%d",&a[i]);
    max=a[1];
    s=a[1];
    for(i=2;i<=10;i++){
        if(s<0)
            s=0;
        s+=a[i];
        if(s>max)
            max=s;
```

```
    }
    printf("max=%d\n",max);
    return 0;
}
```
输入：8 9 -1 24 6 5 11 15 -28 9
输出：max=( )
A. 24　　B. 77　　C. 70　　D. 61

2.
```
#include<iostream>
#include<string>
using namespace std;

int main()
{
    string map= "22233344455566677778889999";
    string tel;
    int i;
    cin>>tel;
    for(i=0;i<tel.length();i++)
       if((tel[i]>='0') && (tel[i]<='9') )
           cout<<tel[i];
       else if( (tel[i]>='A') && (tel[i]<='Z'))
           cout<<map[tel[i]-'A'];
    cout<<endl;
    return 0;
}
```
输入:CCF-CSP/J-2019
输出:_____
A. 223-287/5-2019　　B. 2262181592019　　C. 226-285/9-2019　　D. 22328752019

3.
```
#include<iostream>
using namespace std;

int solve(int n,int m)
{
    int i,sum;
    if(m==1) return 1;
    sum=0;
    for(i=1;i<n;i++)
        sum+= solve(i,m-1);
    return sum;
}
```

```
int main()
{
    int n,m;
    cin>>n>>m;
    cout<<solve(n,m)<<endl;
    return 0;
}
```

输入：7 4

输出：_____

A. 20　　B. 28　　C.21　　D.24

4.

```
#include <stdio.h>
int main(){
    int i,j,k;
    int a[101];
    for(i=0;i<=100;i++)
        a[i]=i;
    for(k=5;k>=2;k--){
        for(i=1;i<=100;i++)
            if(i%k==0)
                a[i]=0;
        for(i=1;i<=99;i++)
            for(j=1;j<=100-i;j++)
                if(a[j]>a[j+1]){
                    a[j]=a[j]+a[j+1];
                    a[j+1]=a[j]-a[j+1];
                    a[j]=a[j]-a[j+1];
                }
    }
    j=1;
    while(a[j]==0&&j<100)
        j++;
    for(i=j;i<=100;i++)
        a[0]=a[0]+a[i];
    printf("%d\n",a[0]);
    return 0;
}
```

A. 935　　B.235　　C.970　　D.870

五、完善程序（前 5 空 2 分，后 4 空 4 分，共 26 分，每空标号处均填选择的答案对应的大写字母）

1. 高精度除法：

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

```
#include<algorithm>
using namespace std;
const int MAXN=5005;
int A[MAXN], B[MAXN], Ans[MAXN] , C[MAXN], Len_A,Len_B,Len_Ans; //Ans是商 C是余数
void Read(int *A) {
    string cur;
    cin>>cur;
    A[0]=cur.length();
    for(int i=1; i<=A[0]; i++) A[i]=cur[i-1]-48;
    reverse(A+1,A+A[0]+1);
}
bool Big() { //比大小
    if(C[0]>B[0]) return true;
    if(C[0]<B[0]) return false;
    for(int i=C[0]; i>=1; i--)    //C数组和B数组长度相同
        if(C[i]<B[i]) return false;
        else if(C[i]>B[i]) return true;
    return   (1)  ;
}

void Minus() { //减法
    int c=0;
    for(int i=1; i<=C[0]; i++) {
        C[i]-=  (2)  ;
        if(C[i]<0) {
            C[i]+=10;
            c=1;
        } else c=0;
    }
    while(C[0]>1&&C[C[0]]==0) C[0]--;  //去掉首位的0
}
void output(int *A) {
    for(int j=A[0]; j>=1; j--) cout<<A[j];
    cout<<endl;
}
int main() {
    Read(A);//被除数
    Read(B);//除数
    C[0]=0;
    for(int i=A[0]; i>=1; i--) {
        for(int j=C[0]; j>=1; j--)   (3)  ;
        C[1]=A[i];
          (4)  ;
        while(C[0]>1&&C[C[0]]==0) C[0]--;
```

```
        while(Big()) {
            Minus();
              (5)  ;
        }
    }
    Ans[0]=A[0];
    while(Ans[0]>1&&Ans[Ans[0]]==0) Ans[0]--;
    output(Ans);
    output(C);
    return 0;
}
```

1-5 空依次应该填：( )( )( )( )( )

A. True　　B. False　　C. B[i]　　D. B[i] +c　　E. c[j]=c[j+1].　　F. C[j+1]=C[j]　　G. C[j-1]=c[j]　　H. c[j]=C[j-1]　　I. C[0]++　　J. C[0]--　　K. Ans[i]++　　L. Ans[0]--

2.

地鼠游戏

地鼠游戏是一项需要反应速度和敏捷判断力的游戏。游戏开始时，会在地板上一下子冒出很多地鼠来，然后等你用榔头去敲击这些地鼠，每个地鼠被敲击后，将会增加相应的游戏分值。问题是这些地鼠不会傻傻地等你去敲击，它总会在冒出一会时间后又钻到地板下面去（而且再也不上来），每个地鼠冒出后停留的时间可能是不同的，而且每个地鼠被敲击后增加的游戏分值也可能是不同，为了胜出，游戏参与者就必须根据每个地鼠的特性，有选择地尽快敲击一些地鼠，使得总的得分最大。

这个极具挑战性的游戏王钢特别喜欢，最近他经常在星期天上午玩这个游戏，慢慢地他不但敲击速度越来越快（敲击每个地鼠所需要的耗时是 1 秒），而且他还发现了游戏的一些特征，那就是每次游戏重新开始后，某个地鼠冒出来后停留的时间都是固定的，而且他记录了每个地鼠被敲击后将会增加的分值。于是，他在每次游戏开始后总能有次序地选择敲击不同的地鼠，保证每次得到最大的总分值。求最大游戏总分值。

```
#include <bits/stdc++.h>
using namespace std;
int n,ans;
struct node {
    int time,v;
} a[2000];
int cmp(node x,node y) {
    return   (6)  ;
}
int main() {
    scanf("%d",&n);
    for (int i=1; i<=n; i++) scanf("%d",&a[i].time);
    for (int i=1; i<=n; i++) scanf("%d",&a[i].v);
    sort(a+1,a+n+1,cmp);
    for (int i=1; i<=n; i++) {
        if (  (7)  ) {
```

```
            int k=i;
            while (  (8)  &&k>1) {
                if (  (9)  ) swap(a[k],a[k-1]);
                a[k-1].time--;
                k--;
            }
        }
    }
    for (int i=1; i<=n; i++) if(a[i].time>0) ans+=a[i].v;
    printf("%d\n",ans);
}
```

6. A. x.time<y.time　　B.x.time>y.time
C.x.v<y.v　　D.x.v>y.v

7. A. a[i].time==a[i+1].time　　B. a[i].v==a[i+1].v
C a[i-1].time==a[i].time　　D.a[i-1].v==a[i].v

8. A. a[k].time!=a[k-1].time　　B. a[k].time>a[k-1].time
   C. a[k].time<a[k-1].time　　D. a[k].time==a[k-1].time

9. A. a[k].v<=a[k-1].v　　B. a[k].v>=a[k-1].v
   C. a[k].time<a[k-1].time　　D. a[k].time>a[k-1].time