2019 CSP-S1 模拟赛3

一: 单项选择题(共10题,每题2分,共计20分;每题有且仅有一个正确选项)

1: 下列哪一种语言是纯函数式编程语言

A: java

B: C++

C: ruby

D: haskell

2: 下列关于CSP-JS认证的说法错误的是:

A:CSP-JS第一轮试卷为纸质版（少数机试方式认证的省份除外），第二轮试卷为电子版。

B:在CSP-JS两轮认证中，认证开始15分钟后，认证者不能再进入认证点。如有认证者提前离开认证点，除身体特别原因外，须在认证进行2小时后方可准予离开。

C:在第一轮认证期间，任何人不得将试卷携带出考场。

D:认证期间，如有认证者相互讨论、使用网络、利用各种方式拷贝或传递信息等违反考场纪律的，涉事认证者均可被立刻取消参赛资格，并从当年算起被禁赛三年。

3: 下面哪个选项是-1234在计算机中的二进制码

A: 11111111111111111111101100101110

B: 10000000000000000000010011010010

C: 10000000000000000000100011010010

D: 11111111111111111111001100101110

4: 下列关于无向图的dfs生成树的说法正确的是

A: 无向图的dfs生成树是最小生成树的一种

B: 无向图的dfs生成树是用kruskal算法求出来的

C: 无向图的dfs生成树分为树边和返祖边两种边

D:无向图的dfs生成树分为树边,返祖边,横叉边三种边

5: 分辨率为900*600, 16位色的位图, 存储图像信息所需的空间为

A: 1054.6875 KB

B: 2019.375KB

C: 32.96KB

D: 4218.75

6: 若某算法的计算时间表示为递推关系式:

T(N) = 2T(N/2) + n^2

则该算法的时间复杂度为:

A:O(n^2logn) B: O(n^2) C: n^3 D:O(nlogn)

7: 有一个整数n=10, 每次会随机选择n的一个因子k, 将n变成k, 请问n变成1的期望步数是多少

A: 8/3 B:8/5

C: 3/2 D: 9/4

11、将数组{1,2,3,4,5,6,7,8}中的元素用插入排序的方法按从大到小的顺序排列，需要比较的

次数是（ ）

A、7 B、27 C、28 D、64

9:已知2019年10月1日是星期二,请问1949年9月30日是星期几

A:星期一 B:星期二 C:星期四 D:星期五

10: 已知二叉树的中序遍历为FGABDCE , 后序遍历为GFADECB

请问先序遍历是?

A: BAFGCDE

B: BCFGADE

C: BCDEAFG

D: BACFGDE

二, 不定项选择题 (共5题, 每题2分, 共计10分;每题有一个或多个正确选项,多选或少选均不得分)

11: 以下属于UDP与TCP的区别的是

A: TCP面向连接（如打电话要先拨号建立连接）。UDP是无连接的，即发送数据之前不需要建立连接

B: TCP提供可靠的服务，通过TCP连接传送的数据，无差错，不丢失，不重复，且按序到达。UDP尽最大努力交付，即不保证可靠交付。

C:TCP传输效率相对较低。UDP传输效率高，适用于对高速传输和实时性有较高的通信或广播通信。

D:TCP与 UDP都支持一对一，一对多，多对一和多对多的交互通信。

12: 下列关于机器字长的说法正确的是

A: 机器字长是指计算机进行一次整数运算所能处理的二进制数据的位数

B: 64位机器中一个int* 指针变量的大小是4个字节

C: 机器字长反映了计算机的运算精度，即字长越长，数的表示范围也越大，精度也越高。

D:机器字长与主存储器的字长都会影响CPU的工作效率

13:以下关于二分图的说法正确的是

A:二分图的最小点覆盖等于二分图的最大匹配

B:二分图的最大独立集等于二分图的最大匹配

C:二分图的最小边覆盖等于二分图的最大匹配

D:二分图的最小点覆盖等于二分图的最小边覆盖

14设栈 S 的初始状态为空，元素 a,b,c,d,e,f,g 依次入栈，以下出栈序列不可能出现的是

A、a,b,c,e,d,f,g B、b,c,a,f,e,g,d C、a,e,d,c,b,f,g D、g,e,f,d,c,b,a

15: 关于线段树与树状数组的描述正确的是

A:根节点区间为[1,n] 的 线段树存在一种标号方法可以恰好只用2倍的空间来标号所有节点

B:树状数组的一个常用作用是维护前缀或者后缀信息

C:线段树区间查询的本质是将一个区间信息划分成最多logn个线段树节点的信息的并

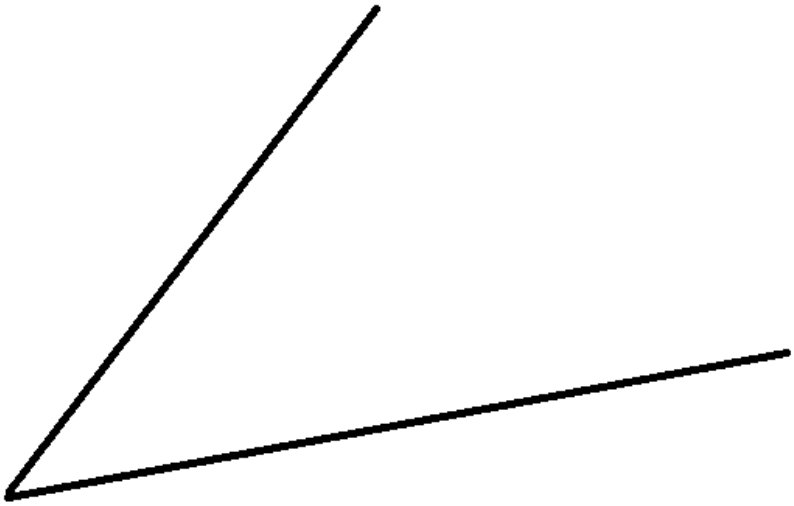
D:树状数组也能支持单点修改,区间维护最大值

.

三. 问题求解(共2题, 每题5分, 共计10分)

1: 在一个夜黑风高的夜晚, 8个人要过桥,每个人单独过桥的时间分别为 10 ,30 ,35, 20, 15, 12, 17, 26 已知桥非常的窄,只允许两个人同时过桥,过桥时间为这两个人中较慢的那个人的过桥时间. 而且他们只有一把手电筒, 没有手电筒是无法过桥的, 请问全部人过桥需要多少时间.

2: 13条如下形状的折线最多将平面分割成多少个区域, 折线可以看成一个点出发的两条射线



四: 阅读程序写结果(共4题, 每题8分, 共计32分)

23:

```
#include<bits/stdc++.h>
using namespace std;

int main() {
    int sum = 0;
    for (int i = 1; i <= 50; i++) {
        sum = sum + i * i * i * i;
    }
    cout << sum <<endl;
}
```

答案

24:

```
#include<bits/stdc++.h>
using namespace std;

int cnt;
int f(int n)
{
    if(n == 1 || n == 2) {
        return ;
    }
    cnt++;
    return f(n - 1) + f(n - 2) + 1;
}
int main() {
   cnt = 0;
   f(20);
   cout <<cnt <<endl;
}
```

输出:

25:

```
#include<cstdio>
#include<cstring>
int m;
char b[1000];
int p[1000];
void getp(){
    p[1]=0;
    int i,j=0;
    for(i=2;i<=m;i++){
        while(j>0&&b[j+1]!=b[i]) j=p[j];
        if(b[j+1]==b[i]) j+=1;
        p[i]=j;
    }
```

```
}
int main() {
    scanf("%s", b + 1);
    m = strlen(b + 1);
    getp();
    for (int i = 1; i <= m; i++) printf("%d ", p[i]);
    return 0;
}
```

输入: aababaaba

26:

```
#include<stdio.h>
#include<string.h>
#define maxn 300000
int a[maxn],c[maxn],p[maxn];
int find(int x){return x==p[x] ? x :
p[x]=find(p[x]);}
int lowbit(int x){
    return x&-x;
}
void update(int x,int d){
    for(;x<=maxn;x+=lowbit(x))
        c[x]+=d;
}
int find_kth(int k)
{
    int ans = 0, cnt = 0, i;
    for (i = 20; i >= 0; i--)
    {
        ans += (1 << i);
        if (ans >= maxn|| cnt + c[ans] >= k)

            ans -= (1 << i);
        else
            cnt += c[ans];
    }
    return ans + 1;
}
```

```
int main()
{
    int i,n,m,q,x,y,k,l,r;
    scanf("%d%d",&n,&m);
    for(i=1;i<=n;i++) p[i]=i;
    for(i=1;i<=n;i++) a[i]=1;
    update(1,n);
    int num=n;
    for(i=1;i<=m;i++)
    {
        scanf("%d",&q);
        if(q==0)
        {
            scanf("%d%d",&x,&y);
            x=find(x);
            y=find(y);
            if(x==y) continue;
            update(a[x],-1);
            update(a[y],-1);
            update(a[x]+a[y],1);
            p[y]=x;
            a[x]+=a[y];
            num--;
        }
        else
        {
            scanf("%d",&k);
            k=num-k+1;
            printf("%d ",find(k));
        }
    }
    return 0;
}
```

输入

```
10 10
0 1 2
1 4
0 3 4
1 2
0 5 6
1 1
0 7 8
1 1
0 9 10
1 1
```

输出

```
```

五. 完善程序(共2题, 每题14分, 共计28分)

27: 寻找一个排列的下一个排列 , 得分为(2,3,3,3,3)

```
class Solution {
public:
    void reverse1(vector<int>& nums,int l,int r){
        while(l<r){
            int temp=nums[l];
            nums[l]=nums[r];
            nums[r]=temp;
            l++,r--;
        }
    }
    void nextPermutation(vector<int>& nums) {
        int j;
      for(int i=nums.size()-1;i>0;i--){
```

```
            if(__(1)__){
                for( j=nums.size()-1;__(2)__;j--){
                    if(___(3)__){
                        break;
                    }
                }
                 swap(___(4)___,____(5)__);
                  reverse1(nums,i,nums.size()-1);
                  return;
            }
        }
        reverse(nums.begin(),nums.end());
    }
};
```

28:求最近点对 (2,3,3,3,3)

```
#include <iostream>
#include <cstring>
#include <algorithm>
#include <cstdio>
#include <cmath>
using namespace std;
struct point
{
    double x,y;
}p[100005];
int a[100005];
int cmpx(const point &a,const point &b)
{
    return a.x < b.x;
}
int cmpy(const int &a,const int &b)
{
    return p[a].y < p[b].y;
}
double dis(int a,int b)
```

```
{
    return sqrt((p[a].x - p[b].x) * (p[a].x - p[b].x)
+ (p[a].y - p[b].y) * (p[a].y - p[b].y));
}
double cloest(int left, int right)
{
    if(left == right)
        return 1000000000;
    if(left + 1 == right)
        return __(1)__;
    int mid = (left + right) >> 1;
    double d1 = cloest(left,mid);
    double d2 = cloest(mid+1,____(2)_);
    double d = min(d1,d2);
    int i,j,k = 0;
    for( i = left ; i <= right; i++ )
    {
        if(___(3)____)
            a[k++] = i;
    }
    sort(a,a+k,___(4)_);
    for( i = 0; i < k - 1; i++ )
    {
        for( j = i+1; j < i + 7 && j < k; j++ )
        {
            if(p[a[j]].y - p[a[i]].y >= d)
                break;
            d = min(d,__(5)__);
        }
    }
    return d;

}
int main()
{
    int i,n;
    while(scanf("%d",&n) != 0)
    {
        if(!n)
            break;
        for( i = 0; i < n; i++ )
```

```
        {
            scanf("%lf %lf",&p[i].x,&p[i].y);
        }
        sort(p,p+n,cmpx);
        printf("%.2f\n",cloest(0,n-1));
    }
    return 0;
```