

每日一题day25_7月1日测评结果

考生信息



张博翔

考号：1675 | 学校：陕西科技大学 | 邮箱：1761607418@qq.com | 职位：43班 |

参考区域: 陕西省西安市 (111.114.0.2) | 做题用时：01:37:20(2019-06-30 22:38:13 - 2019-07-01 00:15:42)

考生成绩



知识点技能图谱



历史笔试记录

序号	试卷名称	排名	总得分	得分详情	作弊嫌疑	笔试时间
1	每日一题day1_5月20日	4.0%	95.0/100	单选:45.0分 编程:50.0分	否	2019-05-19 16:08:49
2	每日一题day02_5月21日	51.0%	60.0/100	单选:30.0分 编程:30.0分	否	2019-05-20 17:40:56
3	每日一题day03_5月22日	4.0%	95.0/100	单选:45.0分 编程:50.0分	否	2019-05-21 12:59:15
4	每日一题day04_5月23日	8.0%	95.0/100	单选:45.0分 编程:50.0分	否	2019-05-22 10:40:18
5	每日一题day05_5月24日	29.0%	75.0/100	单选:25.0分 编程:50.0分	否	2019-05-22 20:39:16
6	每日一题day06_5月25日	10.0%	95.0/100	单选:45.0分 编程:50.0分	否	2019-05-24 10:41:21
7	每日一题day07_5月27日	6.0%	95.0/100	单选:45.0分 编程:50.0分	否	2019-05-26 15:09:41
8	每日一题day08_5月28日	18.0%	90.0/100	单选:40.0分 编程:50.0分	否	2019-05-27 14:06:31
9	每日一题day09_5月29日	13.0%	90.0/100	单选:40.0分 编程:50.0分	否	2019-05-28 15:11:53
10	每日一题day10_5月30日	6.0%	90.0/100	单选:40.0分 编程:50.0分	否	2019-05-29 11:32:20

序号	试卷名称	排名	总得分	得分详情	作弊嫌疑	笔试时间
11	每日一题day11_5月31日	14.0%	80.0/100	单选:30.0分 编程:50.0分	否	2019-05-30 11:22:09
12	每日一题day12_6月1日	31.0%	75.0/100	单选:25.0分 编程:50.0分	否	2019-05-31 10:33:31
13	每日一题day13_6月3日	9.0%	85.0/100	单选:35.0分 编程:50.0分	否	2019-06-02 15:20:12
14	每日一题day14_6月4日	41.0%	50.0/100	单选:25.0分 编程:25.0分	否	2019-06-03 11:53:13
15	每日一题day15_6月5日	6.0%	85.0/100	单选:35.0分 编程:50.0分	否	2019-06-04 12:02:48
16	每日一题day16_6月6日	6.0%	80.0/100	单选:30.0分 编程:50.0分	否	2019-06-05 16:32:52
17	每日一题day17_6月7日	4.0%	90.0/100	单选:40.0分 编程:50.0分	否	2019-06-06 10:57:01
18	每日一题day18_6月8日	16.0%	90.0/100	单选:40.0分 编程:50.0分	否	2019-06-07 15:47:43
19	每日一题day19_6月10日	3.0%	95.0/100	单选:45.0分 编程:50.0分	否	2019-06-09 11:55:46
20	每日一题day20_6月11日	4.0%	95.0/100	单选:45.0分 编程:50.0分	否	2019-06-10 14:43:47
21	每日一题day21_6月12日	5.0%	85.0/100	单选:35.0分 编程:50.0分	否	2019-06-11 11:37:10
22	每日一题day22_6月13日	6.0%	90.0/100	单选:40.0分 编程:50.0分	否	2019-06-12 11:18:48
23	每日一题day23_6月14日	1.0%	100.0/100	单选:50.0分 编程:50.0分	否	2019-06-13 11:26:22
24	每日一题day24_6月15日	7.0%	80.0/100	单选:30.0分 编程:50.0分	否	2019-06-14 12:03:39

编码能力



- 1 [平均分2.7分 | 62人正确/113人做题 | 用时 : <1分 | 得分 : 5.0 / 5.0
一进程刚获得三个主存块的使用权，若该进程访问页面的次序是 { 1321215123 }，采用LRU算法时，缺页数是_____次。
A 3
B 4
C 5
D 6

他的回答： C (正确)
正确答案： C

- 2 [平均分2.0分 | 49人正确/124人做题 | 用时 : 3分 | 得分 : 5.0 / 5.0
以下关于多线程的叙述错误的是：
A 线程同步的方法包括使用临界区，互斥量，信号量等
B 两个线程同时对简单类型全局变量进行写操作也需要互斥
C 实现可重入函数时，对自动变量也要用互斥量加以保护
D 可重入函数不可以调用不可重入函数

他的回答： C (正确)

正确答案： C

3 [平均分4.2分 | 105人正确/126人做题 | 用时：<1分] 得分：5.0 / 5.0

系统死锁的可能的原因是

- A 进程死循环
- B 资源循环等待
- C 程序内存访问越界
- D 进程释放资源

他的回答： B (正确)

正确答案： B

4 [平均分2.5分 | 60人正确/122人做题 | 用时：<1分] 得分：5.0 / 5.0

整数0x12345678，在采用bigendian中内存的排序序列是（ ）

- A 12 34 56 78
- B 78 56 34 12
- C 87 65 43 21
- D 21 43 65 87

他的回答： A (正确)

正确答案： A

5 [平均分3.0分 | 74人正确/125人做题 | 用时：12分] 得分：5.0 / 5.0

使用C语言将一个1G字节的字符数组从头到尾全部设置为字'A'，在一台典型的当代PC上，需要花费的CPU时间的数量级最接近（ ）

- A 0.001秒
- B 1秒
- C 100秒
- D 2小时

他的回答： B (正确)

正确答案： B

6 [平均分3.0分 | 75人正确/123人做题 | 用时：<1分] 得分：0.0 / 5.0

对于普通的计算机，对以下事件的平均耗时从小到大排序为_____：

- A.读取1KB内存数据 B.从硬盘连续读取1KB数据 C.读取一次L2缓存 D.一次磁盘寻道
- A C,A,D,B
- B C,D,A,B
- C D,C,A,B
- D D,A,C,B

他的回答： B (错误)

正确答案： A

7 [平均分1.9分 | 47人正确/121人做题 | 用时：<1分 | 得分：0.0 / 5.0

分页式虚拟存储管理系统中，页面的大小与可能产生的缺页中断次数（ ）

- A 成正比
- B 成反比
- C 无关
- D 成固定值

他的回答：A (错误)

正确答案：C

8 [平均分3.0分 | 74人正确/123人做题 | 用时：<1分 | 得分：5.0 / 5.0

关于子进程和父进程的说法，下面哪一个是正确的？（ ）

- A 一个父进程可以创建若干个子进程，一个子进程可以从属于若干个父进程
- B 父进程被撤销时，其所有子进程也被相应撤消
- C 子进程被撤销时，其从属的父进程也被撤销
- D 一个进程可以没有父进程或子进程

他的回答：D (正确)

正确答案：D

9 [平均分3.8分 | 91人正确/121人做题 | 用时：<1分 | 得分：5.0 / 5.0

关于线程和进程，下面说法正确的是（ ）

- A 终止一个进程比终止一个线程花费的时间少
- B 进程切换比同一进程内部的线程切换花费的时间少
- C 线程提高了不同执行程序间的通信效率
- D 进程和线程都是资源分配和调度的基本单位

他的回答：C (正确)

正确答案：C

10 [平均分2.8分 | 68人正确/123人做题 | 用时：<1分 | 得分：0.0 / 5.0

进程调度时，下列进程状态的变化过程哪一项是不可能发生的？（ ）

- A 阻塞挂起->阻塞
- B 就绪挂起->就绪
- C 就绪挂起->阻塞挂起
- D 阻塞挂起->就绪挂起

他的回答：B (错误)

正确答案：C

11 [平均分19.1分 | 36人正确/47人做题 | 提交: 2 次 | 得分：25.0 / 25.0

标题：星际密码 | 时间限制：1秒 | 内存限制：32768K | 语言限制：不限

【星际密码】星际战争开展了100年之后，NowCoder终于破译了外星人的密码！他们的密码是一串整数，通过一张表里的信息映射成最终4位密码。表的规则是：n对应的值是矩阵X的n次方的左上角，如果这个数不足4位则用0填充，如果大于4位的则只输出最后4位。

|1 1|^n => |Xn ..|

|1 0| |.. ..|

例如n=2时，

|1 1|^2 => |1 1| * |1 1| => |2 1|

|1 0| |1 0| |1 0| |1 1|

即2对应的数是“0002”。

输入描述：

输入有多组数据。

每组数据两行：第一行包含一个整数n (1≤n≤100)；第二行包含n个正整数Xi (1≤Xi≤10000)

输出描述：

对应每一组输入，输出一行相应的密码。

示例1：

输入

6
18 15 21 13 25 27
5
1 10 100 1000 10000

输出

418109877711037713937811
00010089410135017501

代码片段

功能实现			代码提交统计			代码执行统计		
总通过率	TA的	平均		TA的	平均	答案错误：1 答案正确：1		
	100%	76%	使用语言	C++				
	基本测试用例通过率	1/1 (100%)	76%	做题用时	00:47:28			00:58:56
				提交次数	2			4
代码效率					代码规范及可读性			
	TA的	参考				代码规范得分 5.0		
运行时间	277ms	1s						
占用内存	616K	32768K						

他的代码：

做题用时: 47 分钟 语言：C++ 运行时间：277ms 占用内存：616K 程序状态：答案正确

```
#include <iostream>
// #include <stdint>
using namespace std;
int main()
{
    int num1 = 0;
    while(cin >> num1)
    {
        for(int i = 0; i < num1; i++)
        {
            int num;
            cin >> num;
            if(num <= 3)
            {
                cout << "000" << num;
                continue;
            }
            int f1 = 2;
            int f2 = 3;
            int f3;
            for(int i = 4; i <= num; i++)
            {
                f3 = f1 + f2;
```

```
        f1 = f2;
        f2 = f3;
        f2 = f2 % 10000;
    }
    printf("%04d", f2);
}
cout << endl;
}
}
```

12 [平均分17.9分 | 75人正确/105人做题 | 提交: 11 次] 得分 : 25.0 / 25.0

标题：数根 | 时间限制：1秒 | 内存限制：32768K | 语言限制：不限

【数根】数根可以通过把一个数的各个位上的数字加起来得到。如果得到的数是一位数，那么这个数就是数根；如果结果是两位数或者包括更多位的数字，那么再把这些数字加起来。如此进行下去，直到得到是一位数为止。

比如，对于24 来说，把2 和4 相加得到6，由于6 是一位数，因此6 是24 的数根。

再比如39，把3 和9 加起来得到12，由于12 不是一位数，因此还得把1 和2 加起来，最后得到3，这是一个一位数，因此3 是39 的数根。

现在给你一个正整数，输出它的数根。

输入描述：

输入包含多组数据。

每组数据数据包含一个正整数n (1≤n≤10E1000)。

输出描述：

对应每一组数据，输出该正整数的数根。

示例1：

输入

24
39

输出

6
3

代码片段									
功能实现				代码提交统计				代码执行统计	
		TA的	平均			TA的	平均		
总通过率		100%	71%	使用语言		C++		答案错误 : 5	
基本测试用例通过率		1/1 (100%)	71%	做题用时		00:23:05	00:44:51	运行超时 : 2	
				提交次数		11	6	编译错误 : 1	
								答案正确 : 3	
代码效率							代码规范及可读性		
		TA的	参考				代码规范得分		
运行时间		5ms	1s				5.0		
占用内存		356K	32768K						

他的代码：

做题用时: 23 分钟 语言：C++ 运行时间：5ms 占用内存：356K 程序状态：答案正确

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

```
using namespace std;

int numberRoot(string num)
{
    int cur = num[num.size() - 1] - '0';
    //for (auto& c : num) cur += c - '0';
    //num = to_string(cur);
    while (num.size() > 1)
    {
        cur = 0;
        for (auto& c : num)
        {
            cur += c - '0';
        }
        num = to_string(cur);
    }
    return cur;
}

int main()
{
    string num;
    while (cin >> num)
    {
        cout << numberRoot(num) << endl;
    }
    return 0;
}
```