2015初赛讲解

信智学院　陈琰宏

1 单项选择题

1. 1MB 等于（　D　）。

A. 1000 字节　　B. 1024 字节

C. 1000 *1000 字节　　D. 1024 *1024 字节

2. 重新排列 1234 使得每一个数字都不在原来的位置上，一共有（　D　）种排法。

A. 24　　B. 12　　C. 16　　D. 9

2 1 4 3　　3 1 4 2　　4 3 2 1

2 3 4 1　　3 4 1 2　　4 1 2 3

2 4 1 3　　3 4 2 1　　4 3 1 2

这个问题推广一下，就是错排问题，是组合数学中的问题之一

3. 操作系统的作用是（　C）。

A. 把源程序译成目标程序　　B. 便于进行数据管理

C. 控制和管理系统资源　　D. 实现硬件之间的连接

1 单项选择题

4. 前序遍历序列与中序遍历序列相同的二叉树为（D）。
A. 根结点无左子树的二叉树
B. 根结点无右子树的二叉树
C. 只有根结点的二叉树或非叶子结点只有左子树的二叉树
D. 只有根结点的二叉树或非叶子结点只有右子树的二叉树

先序：根左右
中序：左根右　　先序=中序？　先序无左变：根右
　　中序无左变：根右

5. 设某算法的计算时间表示为递推关系式 T(n)=T(n-1)+n(n 为正整数）及 T(0)= 1，则 该算法的时 间复杂度为（ D ）。
A. O(logn)　　B. O(n log n)　C. O(n)D. O(n^2)

T(n)=T(n-1)+n=T(n-2)+n-1+n=T(n-3)+n-2+n-1+n=...=
T(0)+1+2+3+...+n=O(n^2)

1 单项选择题

6. 一棵结点数为 2015 的二叉树最多有(　　)个叶子结点。

A. 1005　　B. 1006　　C. 1007　　D. 1008

n0=n2+1

n=n1+n2+n0, 如果n1=0, 则n0=(n+1)/1=1008

7. 与二进制小数 0.1 相等的十六进制数是（A）

A. 0.8　　B. 0.4　　C. 0.2　　D. 0.1

8. 所谓的“中断”是指（ B　　）。

A. 操作系统随意停止一个程序的运行

B. 当出现需要时，CPU 暂时停止当前程序的执行转而执行处理新情况的过程

C. 因停机而停止一个程序的运行

D. 电脑死机

1 单项选择题

9. 计算机病毒是（　B　）。

A. 通过计算机传播的危害人体健康的一种病毒

B. 人为制造的能够侵入计算机系统并给计算机带来故障的程序或指令集合

C. 一种由于计算机元器件老化而产生的对生态环境有害的物质

D. 利用计算机的海量高速运算能力而研制出来的用于疾病预防的新型病毒

10. FTP 可以用于（　　）　。

A. 远程传输文件　　B. 发送电子邮件

C. 浏览网页　　D. 网上聊天

11. 如果根的高度为 1，具有 61 个结点的完全二叉树的高度为（B）。

A. 5　　B. 6　　C. 7　　D. 8

$2^6=64>61$

1 单项选择题

12. 6 个顶点的连通图的最小生成树，其边数为（ B）。

A. 6　　B. 5　　C. 7　　D. 4

最小生成树的边=n-1=5

13. 链表不具备的特点是（A）。

A. 可随机访问任何一个元素　　B. 插入、删除操作不需要移动元素

C. 无需事先估计存储空间大小　　D. 所需存储空间与存储元素个数成正比

14. 线性表若采用链表存储结构，要求内存中可用存储单元地址（D）

A.必须连续　　B.部分地址必须连续　C. 一定不连续　D.连续不连续均可

15. 今有一空栈 S，对下列待进栈的数据元素序列 a,b,c,d,e,f 依次进行进栈，进栈，出栈，进栈，进 栈，出栈的操作，则此操作完成后，栈 S 的栈顶元素为（B）。

A. f　　B. c　　C. a　　D. b

2 阅读程序写结果1

```
1  #include <iostream>
2  #include <string>
3  using namespace std;
4  int main() {
5      string str;
6      int i;
7      int count;
8      count = 0;
9      getline(cin,str);
10     for (i = 0; i < str.length(); i++) {
11         if(str[i] >= 'a' && str[i] <= 'z')
12             count++; //统计小写字母的个数
13     }
14     cout << "It has " << count << " lowercases" << endl;
15     return 0;
16 }
```

2 阅读程序写结果1

●判断题

(1)第9行输入的字符串可以是任意字符,包括字母、数字、各类符号甚至中文汉字及符号。(T)

(2)执行第10行循环后,count的值可能为0。(T)

输入都是大写字母就可能为0

(3)若去掉11行,输出结果不变。(F)

没有了条件判断，当然不一样

(4)若输入的字符串中各字符互不相同,则 count不为0。(F)

如ABCD,没有小写就是0

●选择题

(5)输入CSP2001 ， count的结果是(D)。

A.3 B.7 C.8 D.0

(6)若字符串的长度为n,算法的时间复杂度是(A)。

A.O(n) C.O(n2)

B.O(nlogn) D.O(nlog2n)

2 阅读程序写结果2

```
#include<iostream>
#include<string>
using namespace std;
int main( ){
    int len, maxlen;
    string s, ss;
    maxlen=0;
    do {
        cin>>ss;
        len=ss.length();
        if (ss[0]=='#') break;
        if (len>maxlen){
            s=ss;//求长度最大的字符串
            maxlen=len;
        }
    }while(true);
    cout<<s<<endl;
    return 0;
}
```

判断题

(1)输出可以包含#。(T)

如果#不做为字符串的第一个字符就不会退出，可以输出。

(2)如果去掉第7行的初始化，可能得不到正确答案。(T)

局部变量没有初始化，值未定。

(3)输出一定有字符。(F)

如果输入都是#，则输出为空

(4)如里把12行的>改为>=,结果不会改变。(F)

后面长度相等的会替换前面的。

选择题

(5)程序的时间复杂度级别为(A)。

A.线性 B.对数 C.常数 D.平方

(6)输入I am a citizen of China # ,输出(C)。

A.am B.a C.citizen D. China

2 阅读程序3[6556]

```
1  #include <iostream> //找规律f(i)=f(i-1)*2+1
2  using namespace std;
3  int fun(int n, int fromPos, int toPos) { //f(n)
4      int t, tot;
5      if (n == 0) return 0;
6      for (t = 1; t <= 3; t++)
7          if (t != fromPos && t != toPos) break;
8      tot = 0;
9      tot += fun(n - 1, fromPos, t); //f(n-1)
10     tot++;                         //+1
11     tot += fun(n - 1, t, toPos); //f(n-1)
12     return tot;
13 }
14 int main( ) {
15     int n;
16     cin >> n;
17     cout << fun(n, 1, 3) << endl;
18     return 0;
19 }
```

找出递推式：
f(i)=f(i-1)*2+1
通式：2^n-1
后两个参数对
程序没有影响，

3 阅读程序写结果3

判断题

(1)当n为小于1000的正整数时,将第9行和第11行一起去掉,程序输出结果为1。（ T ）去掉以后，没有实现累加，就是1

(2)当n为小于1000的正整数时,将第9行或第11行中其中一行去掉,程序输出n。（ T ）f(n)=f(n-1)+1

(3)函数中的 fromPos与 toPos与答案无关。（ T ）

(4)该程序的时间复杂度为O(2^n)。（ T ）

选择题

(5)fun(5,1,3)的值为(A)。

A.31 B.23 2^n-1

C.66 D.9

(6)fun(n,1,3)的通项公式为(C)。

A.fun(n-1,1,3)*2＋1 B.2*n

C.2^n-1 D.(((1+√5)/2)n-((1-√5)/2)n)/√5

完善程序1[6338]：打印月历

输入月份 m (1 ≤m ≤ 12)，按一定格式打印 2015 年第 m 月的月历。（第 三、四空 2.5 分， 其余 3 分）
例如，2015 年 1 月的月历打印效果如下（第一列为周日）：

S	M	T	W	T	F	S
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31

```cpp
#include <iostream>
using namespace std;
const int dayNum[]= {-1, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};
int m,offset, i;
int main() {
    cin >> m;
    cout <<" S  M  T  W  T  F  S " <<endl; // '\t'为 TAB  制表符
    offset=3;//   (2015年的一月一号是从星期4开始的)
    for (i = 1; i < m; i++)//用来判断第m个月时，我们需要打几个制表符
        offset = (offset+dayNum[i])%7;
        // (过初始的值加上前m个月的天数对7取余就能得到第m个月需要打几个制表符)

    for (i = 1; i <=  dayNum[m]; i++)
    {//  从第一天一直到m月的最后一天
        cout << i<<"  ";// ;  (输出这是第几号)
        if (i == dayNum[m] || (offset+i)%7== 0)
            cout << endl;// 控制格式，最后一天或者到了星期六要换行）

    }
    return 0;
}
```

(1)①处应填(　　)。

A.2　　B.3　　C. 4　　D.5

(2)2处应填(　　)。

A. dayNum[0]　　B.dayNum[i-1]　　C. dayNum[i]　　D.dayNum[3]

(3)3处应填(　　)。

A. dayNum[m]　　B.dayNum[m* m]

C.m　　D.m*m

(4)4处应填(　　)。

A.&&　　B.II

C. !　　D.==

(5)⑤处应填(　　)。

A. dayNum[offset]　　B.!

C.(offset+dayNum[0])　　D.(offset+i)

3 完善程序2[6613]

(中位数）给定 n (n 为奇数且小于 1000)个整数，整数的范围在 0～m (0 < m < 231) 之间，请使用二分法求这 n 个整数的中位数。所谓中位数，是指将这 n 个数排序之后， 排在正中间的数。（第五空 2分，其余 3 分）

3 完善程序

```
#include <iostream>
using namespace std;
const int MAXN = 1000;
int n, i, lbound, rbound, mid, m, count;
int x[MAXN];
int main(){
    cin >> n >> m;
    for (i = 0; i < n; ++i)
    cin >> x[i];
    lbound=0;
    rbound=m;//
    while ( lbound<rbound   ) {  //1
        mid=(lbound+rbound)/2;
        int count=0 ; //2
        for (i = 0; i < n; i++)
            if ( mid<x[i]  )   //3
                count++ ; //4
        if (count > n / 2)
            lbound = mid + 1;
        rbound=mid  ;//5
    }
    cout << rbound << endl;
    return 0;
}
```

```
 2 const int   SIZE = 100;
 3 int stack1[SIZE], stack2[SIZE];
 4 int top1, top2;
 5 int n, m, i, j;
 6 void clearStack()
 7 {
 8     int i;
 9     for ( i = top1; i < SIZE; i++ )
10         stack1[i] = 0;
11     for ( i = top2; i < SIZE; i++ )
12         stack2[i] = 0;
13 }
14 int main( )
15 {
16     scanf("%d%d", &n, &m);
17     for ( i = 0; i < n; i++ )
18         scanf("%d", &stack1[i]);
19     top1 = ___1____;//1 初始数据存在stack1，n-1的下一个元素为n
20     top2 = ___2____;//2 st2初始为空即-1 ， 下一个位置为0
```

```
21      for ( j = 0; j < m; j++ )
22      {
23          scanf("%d",&i);
24          while ( i < top1 - 1 )
25          {//下一次访问的元素在i之前时，
26          //通过将st1的栈顶元素取出放到st2,实现“ 指针的向前移动”
27              top1--;//从st1拷贝元素到st2 形成0~i的元素在st1
28              ___3____;//3
29              top2++;//
30          }
31          while (i > top1 - 1)
32          {//下一次访问的元素在i之后时，
33          //通过将st2的栈顶元素取出放到st1,实现“ 指针的向后移动”
34              top2--;
35              ___4____;
36              top1++;
37          }
38          clearStack();
39          printf("%d\n", stack1[___5____;]);//st1的栈顶元素为答案
40      }
41      return(0);
42  }
```

```
 2  const int   SIZE = 100;
 3  int stack1[SIZE], stack2[SIZE];
 4  int top1, top2;
 5  int n, m, i, j;
 6  void clearStack()
 7  {
 8      int i;
 9      for ( i = top1; i < SIZE; i++ )
10          stack1[i] = 0;
11      for ( i = top2; i < SIZE; i++ )
12          stack2[i] = 0;
13  }
14  int main( )
15  {
16      scanf("%d%d", &n, &m);
17      for ( i = 0; i < n; i++ )
18          scanf("%d", &stack1[i]);
19      top1 = n;//1 初始数据存在stack1，n-1的下一个元素为n
20      top2 = 0;//2 st2初始为空即-1 ， 下一个位置为0
```

```
21      for ( j = 0; j < m; j++ )
22      {
23          scanf("%d",&i);
24          while ( i < top1 - 1 )
25          {//下一次访问的元素在i之前时，
26          //通过将st1的栈顶元素取出放到st2,实现“ 指针的向前移动”
27              top1--;//从st1拷贝元素到st2 形成0~i的元素在st1
28              stack2[top2]=stack1[top1];//3
29              top2++;//
30          }
31          while (i > top1 - 1)
32          {//下一次访问的元素在i之后时，
33          //通过将st2的栈顶元素取出放到st1,实现“ 指针的向后移动”
34              top2--;
35              stack1[top1]=stack2[top2];
36              top1++;
37          }
38          clearStack();
39          printf("%d\n", stack1[top1-1]);//st1的栈顶元素为答案
40      }
41      return(0);
42  }
```

●选择题

(1)①处应填(B)。

A.0 B.n C.1 D.-1

(2)2处应填(D).

A.1 B.2 C.n D.0

(3)③处应填(A)。

A. stack2[top2]=stack1[top1] B.top1=top2

C. stack2[top1]=stack1[top1] D.stack2[top1]=stack1[top2]

(4)4处应填(C)。

A.stack1[top1]=stack2[top1] B.stack1[top2]=stack2[top1]

C.stack1[top1]=stack2[top2] D.stack1[top2]=stack2[top2]

(5)⑤处应填(C)。

A. top1 B.top2 C.top1-1 D.top2-1

今天的课程结束啦……

下课了…
同学们再见！