GESP C++ 五级试卷（样题）

（满分：100 分　考试时间：180 分钟）

学校：_____　　姓名：_____

题目	一	二	三	总分
得分				

一、单选题（每题 2 分，共 30 分）

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

1、以下不属于计算机输出设备的有（　）。

A. 麦克风

B. 音箱

C. 打印机

D. 显示器

2、小明想了一个 1 到 100 之间的整数。你可以做多次猜测，每次猜测之后，如果你没有猜中，小明会告诉你，你猜的数比他想的数大还是小。你希望你在运气最坏的情况下花费最少的次数猜中，请问你运气最坏的情况下会猜（　）次？（包括最后猜中的那次）。

A. 5

B. 6

C. 7

D. 100

3、关于分治算法，下列说法错误的是（　）。

A. 分治算法的核心思想是分而治之，即把问题转化为多个规模更小的子问题求解。

B. 分治算法可以不使用递归实现。

C. 分治算法的时间复杂度是 $O(\log N)$，其中 N 表示问题的规模。

D. 分治算法通常较容易在多核处理器上实现加速。

4、 有关下面C++代码说法错误的是（　）。

```cpp
#include <iostream>
using namespace std;

int factA(int n) {
    if (n <= 1)
        return 1;
    int ret = 1;
    for (int i = 2; i <= n; ++i)
        ret *= i;
    return ret;
}

int factB(int n) {
    return n == 1 ? 1 : n * factB(n - 1);
}

int main() {
    int n;
    cin >> n;
    cout << factA(n) << ' ' << factB(n) << endl;
    return 0;
}
```

A. factA()采用循环方式求n的阶乘，factB()采用递归方式求n的阶乘

B. 程序执行时如果输入5，能实现求其阶乘，程序输出结果为120 120

C. 任何递归程序都可以使用循环改写

D. 程序执行时如果输入100，不能正确求出100的阶乘

5、下面C++代码意在实现字符串反序的功能。关于这段代码，以下说法正确的是（　）

```cpp
#include <iostream>
#include <cstring>
using namespace std;

void printSReverse(char* sIn, int len) {
    if (len <= 1) {
        cout << sIn[0];
    }
    else {
        cout << sIn[0];
        printSReverse(sIn + 1, len - 1);
    }
}

int main() {
    char sIn[100] = "Hello";
    printSReverse(sIn, strlen(sIn));
    return 0;
}
```

A. 这段代码可以正确实现字符串反序的功能，其输出为`olleH`

B. 这段代码不能正确实现字符串反序的功能，其输出为`Hello`

C. 这段代码不能正确实现字符串反序的功能，其输出为`HHHHH`

D. 这段代码不能正确实现字符串反序的功能，其输出为`ooooo`

6、阅读下面 C++实现的二分查找代码，下列说法中错误的是（　）。

```cpp
int binarySearch(int * arr, int l, int r, int x) {
    if (r >= l) {
        int mid = l + (r - l) / 2;
        if (arr[mid] == x)
            return mid;
        else if (arr[mid] > x)
            return binarySearch(arr, l, mid - 1, x);
        else
            return binarySearch(arr, mid + 1, r, x);
    }
    else
       return -1;
}
```

A. 上面代码实现的二分查找，最少只需查找一次即可得到结果

B. 如果调用该函数在列表{2, 3, 4, 10, 12}中查找元素 0，则它实际被调用 3

次

C. 如果调用该函数在列表{2, 3, 4, 10, 12}中查找元素 3，则它实际被调用 3 次

D. 如果调用该函数在列表{2, 3, 4, 10, 12}中查找元素 10，则它实际被调用 3 次

7、使用如下代码片段定义四个字符串（假设头文件已正确定义），以下说法错误的是（ ）。

```
string str1 = "abc";
string str2 = str1;
char str3[] = "abc";
char * str4 = str3;
```

A. 对于四个字符串，都可以使用 std::cout 输出其中的内容（例如，cout << str3;）

B. str3 只占用 4 字节内存，但 str1 却要占用更多内存

C. 由于 str2 由 str1 直接赋值得到，因此二者指向同一块内存，即修改 str1 的内容后 str2 的内容也会随之改变

D. 由于 str4 由 str3 直接赋值得到，因此二者指向同一块内存，即修改 str3 的内容后 str4 的内容也会随之改变

8、有关下面 C++代码正确的是（ ）。

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

void f1() {
    cout << "f1()" << endl;
}

void f1(int x) {
    cout << "f1(" << x << ")" << endl;
}

int main() {
    f1();
    f1(0);
    f1('0');
    return 0;
}
```

A. 该程序不能正常运行，因为 f1 函数被重复定义。

B. 该程序可以正常运行，输出结果共 3 行，依次为 f1()、f1()、f1()

C. 该程序可以正常运行，输出结果共 3 行，依次为 f1()、f1(0)、f1(0)

D. 该程序可以正常运行，输出结果共 3 行，依次为 f1()、f1(0)、f1(48)

9、关于 C++程序的异常处理，以下选项中描述错误的是（ ）。

A. 编程语言中的异常和错误是不同的概念

B. 异常一旦发生，程序便一定不能继续执行

C. 通过 try、catch 等保留字提供异常处理功能

D. 程序使用 throw 在任何地方抛出各种异常

10、下面代码执行后的输出是（ ）。

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

int fibonacci(int N) {
    cout << N << ",";
    if (N == 1 || N == 2) {
        return 1;
    } else {
        return fibonacci(N - 1) + fibonacci(N - 2);
    }
}

int main() {
    cout << fibonacci(5) << endl;
    return 0;
}
```

A. 5,4,3,2,1,2,1,5

B. 5,4,3,2,1,2,3,2,1,5

C. 5,4,4,3,2,1,3,2,1,5

D. 5,4,3,2,1,3,2,1,5

11、下列代码中，函数 f 的作用是（ ）。

```
void f(int a, int b) {
    return b == 0 ? a : f(b, a % b);
}
```

A. 求 a 和 b 的最大公共质因子

B. 求 a 和 b 的最小公共质因子

C. 求 a 和 b 的最大公约数

D. 求 a 和 b 的最小公倍数

12、下面 C++代码用于排序，下列说法中错误的是（　）。

```cpp
void sortA(int * arr, int n) {
    for (int i = 0; i < n; ++i)
        for (int j = 0; j < n - i - 1; ++j)
            if (arr[j] > arr[j + 1]) {
                int tmp = arr[j];
                arr[j] = arr[j + 1];
                arr[j + 1] = tmp;
            }
}

void sortB(int * arr, int start, int end) {
    if (start >= end)
        return;

    int middle = (start + end) / 2;
    sortB(arr, start, middle);
    sortB(arr, middle + 1, end);

    int leftSize = middle - start + 1;
    int rightSize = end - middle;

    int * left = new int[leftSize];
    int * right = new int[rightSize];

    for (int i = 0; i < leftSize; i++)
        left[i] = arr[start + i];
    for (int j = 0; j < rightSize; j++)
        right[j] = arr[middle + 1 + j];

    int i = 0;
    int j = 0;
    int k = start;
    while (i < leftSize && j < rightSize) {
        if (left[i] <= right[j]) {
            arr[k] = left[i];
            i++;
        } else {
            arr[k] = right[j];
            j++;
        }
        k++;
    }
    while (i < leftSize) {
        arr[k] = left[i];
```

```
45          i++;
46          k++;
47      }
48      while (j < rightSize) {
49          arr[k] = right[j];
50          j++;
51          k++;
52      }
53
54      delete[] left;
55      delete[] right;
56  }
```

A. 两种排序算法的时间复杂度不同。

B. 两种排序算法的空间复杂度一致。

C. sortA 的时间复杂度在最好和最坏情况下都是 $O(N^2)$。

D. sortB 的平均时间复杂度、最好情况的时间复杂度都是 $O(\log N)$，最坏情况的时间复杂度是 $O(N^2)$。

13、上一题中的`sortB`函数，明显体现出的算法思想和编程方法包括（　）。

A. 递归

B. 分治

C. A、B 都正确

D. A、B 都不正确

14、下列哪个算法并没有体现分治思想？（　）。

A. 二分查找

B. 埃氏筛法。

C. 归并排序。

D. 快速排序。

15、下列关于链表说法，正确的是（　）。

A. 不能用数组来实现链表。

B. 在链表头部插入元素的时间复杂度是 $O(1)$。

C. 循环链表使得任意一个结点都可以很方便地访问其前驱与后继。

D. 从双向链表的任意一个节点出发，并不一定能够访问到所有其他节点。

二、判断题（每题 2 分，共 20 分）

题号	1	2	3	4	5	6	7	8	9	10
答案	√	×	×	×	√	×	×	√	√	×

1、计算机硬件主要包括运算器、控制器、存储器、输入设备和输出设备。

2、唯一分解定理指的是分解质因数只有唯一的一种算法。

3、埃氏筛法用于找出自然数 N 以内的所有质数，其时间复杂度为 $O(N\sqrt{N})$，因为判定一个数是否为质数的时间复杂度为 O(N)。

4、贪心法的每一步都采取局部最优策略，因此必然能实现全局最优。

5、在 C++语言中，函数的参数也可以是另一个函数。

6、在 C++语言中，内置的排序算法（algorithm 库中的 sort 函数）只能对 C++的基础类型（如 int、double 等）做排序，而不能对自定义类型做排序。

7、在任何场景下，链表都是比数组更优秀的数据结构。

8、在 C++语言中，可以使用 delete 来释放指针指向的内存空间。

9、选择排序和快速排序都是不稳定的。

10、二分查找法可以应用于有序序列（如升序排序的整数数组），也可以应用于无序序列（如乱序的整数数组）。

三、编程题（每题 25 分，共 50 分）

题号	1	2
答案		

1、小杨的锻炼

【问题描述】

小杨的班级里共有 N 名同学，每位同学都有各自的锻炼习惯。具体来说，第 i 位同学每隔 a_i 天就会进行一次锻炼（也就是说，每次锻炼会在上一次锻炼的 a_i 天后进行）。

某一天，班上的 N 名同学恰好都来进行了锻炼。他们对此兴奋不已，想要计算出下一次所有同学都来锻炼，至少要过多少天。但他们不会计算，你能帮帮他们吗？

时间限制：1.0s

内存限制：128.0MB

【输入描述】

第一行一个整数 N，表示同学的数量。

第二行 N 个用空格隔开的正整数，依次为 a_0, a_1, …, a_n-1。

【输出描述】

输出一个整数，表示下一次所有同学都来锻炼，至少要过多少天。

【特别提醒】

在常规程序中，输入、输出时提供提示是好习惯。但在本场考试中，由于系统限定，请不要在输入、输出中附带任何提示信息。

【样例输入 1】

3

1 2 3

【样例输出 1】

6

【样例解释 1】

第一位同学每天都锻炼；第二位同学每 2 天锻炼一次；第三位同学每 3 天锻炼一次。因此，6 天之后，三位同学都会进行锻炼。在此之前，第二位同学只会在第 2, 4 天进行锻炼，第三位同学只会在第 3 天进行锻炼，他们都无法相遇。

【样例输入 2】

4

2 4 8 16

【样例输出 2】

16

【样例解释 2】

第四位同学每 16 天锻炼一次，而第 16 天后也恰好是前三位同学锻炼的日子。

【样例输入 3】

4

2 4 6 8

【样例输出 3】

24

【数据规模】

对于 20% 的测试点，保证 N = 2。

对于 50% 的测试点，保证 N ≤ 4。

对于所有测试点，保证 2 ≤N ≤ 10， 1≤a_i≤ 50。

2、小杨的队列

【问题描述】

小杨的班级里共有 N 名同学，学号从 0 至 N - 1。

某节课上，老师要求同学们进行列队。具体来说，老师会依次点名 M 名同学，让他们加入队伍。每名新入队的同学需要先站到队伍末尾（刚开始队伍里一个人都没有，所以第一个入队的同学只需要站好即可），随后，整个队伍中的所有同学需要按身高从低到高重新排序（身高相同的同学之间的顺序任意）。

排队很容易，但重新排序难倒了同学们。稍加讨论后，他们发现可以通过交换位置的方法来实现排序。具体来说，他们可以让队伍中的两名同学交换位置，这样整个队伍的顺序就会发生变化，多经过这样的几次交换后，队伍的顺序就可以排好。

例如：队伍中有 4 名同学，学号依次为 10, 17, 3, 25，我们可以令 3 号同学和 10 号同学交换位置，则交换后的队伍顺序变为 3, 17, 10, 25，这就是一次交换位置。

聪明的小杨想要知道：在老师每次点名一位新同学加入队伍后，在原有队伍的基础上，同学们最少要进行几次**交换位置**，才能完成老师按身高排序的要求。

【输入描述】

第一行一个整数 N，表示同学的数量。

第二行 N 个用空格隔开的正整数，依次表示学号为 0, 1, …, N-1 的同学的身高（不超过 2,147,483,647）。

第三行一个整数 M，表示老师点名的数量。

接下来 M 行，依次描述 M 次点名：每行一个整数 x（0 ≤ x < N），表示要求学号为 x 的同学加入队伍。保证该名同学此前不在队伍中。

对于所有的测试点，保证 1≤ M ≤ N ≤ 2000。对于 50% 的测试点，保证所有同学的身高互不相同。

【输出描述】

输出 M 行，依次表示对于每次点名，同学们最少要进行几次**交换位置**，才能完成按身高排序的要求。

【特别提醒】

在常规程序中，输入、输出时提供提示是好习惯。但在本场考试中，由于系统限定，请不要在输入、输出中附带任何提示信息。

【样例输入 1】

```
5
170 165 168 160 175
4
0
3
2
1
```

【样例输出 1】

```
0
1
1
2
```

【样例解释 1】

初始时队伍为空，身高为 170 的 0 号同学加入队伍，不需要任何交换位置。

接着，身高为 160 的 3 号同学加入队伍的末尾，此时两位同学需要进行依次交换位置，才能保证身高更矮的 3 号同学排在身高更高的 0 号同学前面。

接着，身高为 168 的 2 号同学加入队伍的末尾，此时队伍中的同学学号（身高）依次为 3(160)，0(170)，2(168)，此时 2 号同学可以和 0 号同学进行一次交换位置，即可完成排序要求。

接着，身高为 165 的 1 号同学加入队伍的末尾，此时队伍中的同学学号（身高）依次为 3(160)，2(168)，0(170)，1(165)，此时可以令 1 号同学和 2 号同学进行一次交换位置，使队伍变为 3(160)，1(165)，0(170)，2(168)；随后再令 0 号同学和 2 号同学进行一次交换位置，使队伍变为 3(160)，1(165)，2(168)，0(170)，即可完成排序要求。

【样例输入 2】

4

20 20 20 10

4

0

1

2

3

【样例输出 2】

0

0

0

1

【样例解释 2】

前三位加入队伍的同学（0，1，2 号同学）身高都相同，不需要进行任何交换位置。最后加入队伍的 3 号同学身高最矮，需要和队头的 0 号同学交换位置，方可完成排序要求。