



C++ 四级

2024 年 09 月

1 单选题（每题 2 分，共 30 分）

第 1 题 在 C++ 中，（ ）正确定义了一个返回整数值并接受两个整数参数的函数。

- ☐ A. `int add(int a, int b) { return a + b; }`
- ☐ B. `void add(int a, int b) { return a + b; }`
- ☐ C. `int add(a, b) { return a + b; }`
- ☐ D. `void add(int a, int b) { return a - b; }`

第 2 题 在 C++ 中，形参与实参的关系描述正确的是（ ）。

- ☐ A. 形参在函数调用时指定，实参在函数定义时传递
- ☐ B. 形参在函数定义时指定，实参在函数调用时传递
- ☐ C. 形参和实参可以互换
- ☐ D. 形参和实参必须是完全相同的类型，不能有任何差异。

第 3 题 运行以下代码，屏幕上将输出（ ）。

```
1  #include <iostream>
2  using namespace std;
3
4  int var = 100;
5
6  void function() {
7      int var = 200;
8
9      cout << var << " ";
10     cout << ::var << " ";
11 }
12
13 int main() {
14     cout << var << " ";
15
16     function();
17
18     var += 100;
19     cout << var << " ";
```

```
20  
21     return 0;  
22 }
```

- ☐ A. 100 200 100 200
- ☐ B. 100 200 100 300
- ☐ C. 100 200 200 200
- ☐ D. 100 200 200 300

第4题 运行下面代码，屏幕上输出是（ ）。

```
1  int arr[3] = {24, 9, 7};  
2  int* p = arr;  
3  p++;  
4  cout << *p << endl;
```

- ☐ A. 24
- ☐ B. 9
- ☐ C. 7
- ☐ D. 不确定

第5题 运行下面代码片段的的结果是（ ）。

```
1  int x = 20;  
2  int y = 24;  
3  int* p = &x;  
4  int* q = &y;  
5  
6  p = q;
```

- ☐ A. 将x赋值为24
- ☐ B. 将y赋值为20
- ☐ C. 将q指向x的地址
- ☐ D. 将p指向y的地址

第6题 在C++中，（ ）正确定义一个名为 student 的结构体，其中包含一个 name 字符数组和一个 age 整数？

- ☐ A. struct student { char name[20]; int age; };
- ☐ B. student struct { char name[20]; int age; };
- ☐ C. student struct { string name; int age; };
- ☐ D. struct student { char[20] name; int age; };

第7题 在C++中，（ ）正确声明了一个3行4列的二维数组。

- ☐ A. int arr[3, 4];

- ☐ B. `int arr[3][4];`
- ☐ C. `int arr[4][3];`
- ☐ D. `int arr(3, 4);`

第8题 一个二维数组定义为 `int arr[3][4];`（假设一个int变量占4个字节），则 `int arr[0]` 占用（ ）个字节的内存。

- ☐ A. 3
- ☐ B. 4
- ☐ C. 12
- ☐ D. 16

第9题 下面代码采用递推算法来实现整数 n 的阶乘（ $n! = n \times (n-1) \times \dots \times 2 \times 1$ ），则横线上应填写（ ）。

```
1  int factorial(int n) {
2      int result = 1;
3      for (int i = 2; i <= n; i++) {
4          _____ // 在此处填入代码
5      }
6      return result;
7  }
```

- ☐ A. `result *= i;`
- ☐ B. `result += i;`
- ☐ C. `result *= result;`
- ☐ D. `result += result;`

第10题 在排序算法中，稳定性指的是（ ）。

- ☐ A. 排序后数据不会丢失
- ☐ B. 排序后相同元素的相对顺序保持不变
- ☐ C. 排序后数据不会被修改
- ☐ D. 排序后数据的时间复杂度不变

第11题 下面代码实现了冒泡排序函数，则横线上应填写（ ）。

```
1  //交换数组arr的第i个元素和第j个元素
2  void swap(vector<int> &arr, int i, int j) {
3      int tmp = arr[i];
4      arr[i] = arr[j];
5      arr[j] = tmp;
6  }
7
8  int bubble_sort(vector<int> &arr) {
9      for (int i = arr.size() - 1; i > 0; i--) {
10         bool flag = false; // 标志位
11         _____ { // 在此处填入代码
```

```

12
13         if (arr[j] > arr[j + 1]) {
14             swap(arr, i, j);
15             flag = true;
16         }
17     }
18     if (!flag)
19         break; // 此轮“冒泡”未交换任何元素
20 }
21 }

```

- ☐ A. for (int j = 0; j < arr.size() - 1; j++)
- ☐ B. for (int j = arr.size() - 1; j > 0; j--)
- ☐ C. for (int j = 0; j < i; j++)
- ☐ D. for (int j = i-1; j <=0; j--)

第12题 上一题算法的时间复杂度为（ ）。

- ☐ A. $O(n^2)$
- ☐ B. $O(2^n)$
- ☐ C. $O(1)$
- ☐ D. $O(n)$

第13题 下面代码实现了插入排序函数（升序），则横线上应填写（ ）。

```

1 void insertion_sort(vector<int> &nums) {
2     for (int i = 1; i < nums.size(); i++) {
3         int base = nums[i], j = i - 1;
4
5         _____ { // 在此处填入代码
6             nums[j + 1] = nums[j];
7             j--;
8         }
9         nums[j + 1] = base;
10    }
11 }

```

- ☐ A. while (j >= 0 && nums[j] > base)
- ☐ B. while (j > 0 && nums[j] > base)
- ☐ C. while (j >= 0 && nums[j] < base)
- ☐ D. while (j > 0 && nums[j] < base)

第14题 小杨用文件重定向实现在 log.txt 文件中输出日志，则下面横线上应填写（ ）。

```

1 #include <iostream>
2 #include <fstream>
3 #include <string>
4 using namespace std;

```

```

5
6 int main() {
7     ofstream log_file("log.txt");
8
9     streambuf* original_cout = cout.rdbuf();
10    cout.rdbuf(log_file.rdbuf());
11
12    _____ // 在此处填入代码
13
14    cout.rdbuf(original_cout); // 恢复原始的标准输出缓冲区
15    return 0;
16 }

```

- ☐ A. `cout << "This output will go to the log file." << endl;`
- ☐ B. `log_file << "This output will go to the log file." << endl;`
- ☐ C. `cout >> "This output will go to the log file." >> endl;`
- ☐ D. `log_file >> "This output will go to the log file." >> endl;`

第15题 运行下面的代码，屏幕上将输出（ ）。

```

1 #include <iostream>
2 using namespace std;
3
4 int divide(int a, int b) {
5     if (b == 0) {
6         throw runtime_error("division by zero error ");
7     }
8     return a / b;
9 }
10
11 int main() {
12     int x = 10;
13     int y = 0; // 设为 0 会导致除零错误
14
15     try {
16         int result = divide(x, y);
17         cout << "result: " << result << endl;
18     } catch (const runtime_error& e) {
19         cout << "caught an exception: " << e.what() << endl;
20     }
21
22     return 0;
23 }

```

- ☐ A. division by zero error result: caught an exception:
- ☐ B. result: caught an exception: division by zero error
- ☐ C. caught an exception: division by zero error
- ☐ D. division by zero error caught an exception: division by zero error

2 判断题（每题 2 分，共 20 分）

第 1 题 代码 `int a = 10; int* p = &a;` 可以正确定义指针和初始化指针。

第 2 题 在 C++ 中，引用传递允许函数修改传递给它的参数的值。

第 3 题 指针的大小与其所指向的变量的数据类型的大小相同。

第 4 题 二维数组的行的大小的必须在定义时确定，列的大小可以动态变化。

第 5 题 递推算法通过逐步求解当前状态和前一个或几个状态之间的关系来解决问题。

第 6 题 选择排序是稳定的排序算法。

第 7 题 插入排序的时间复杂度总是比冒泡排序低。

第 8 题 在 C++ 中，如果没有捕获到异常（没有匹配的 catch 块），程序会继续执行而不会终止。

第 9 题 以下代码用递推法求斐波那契数列的第 n 项，时间复杂度为指数级。

```
1  int fibonacci(int n) {
2      if (n == 0) return 0;
3      if (n == 1) return 1;
4
5      int f0 = 0; // F(0)
6      int f1 = 1; // F(1)
7      int current;
8
9      for (int i = 2; i <= n; i++) {
10         current = f0 + f1; // F(n) = F(n-1) + F(n-2)
11         f0 = f1;
12         f1 = current;
13     }
14
15     return current;
16 }
```

第 10 题 执行下面 C++ 代码后，输出的是 20。

```
1  int point(int* p){
2      return *p * 2;
3  }
4
5  int main() {
6      int a = 10;
7      int* p = &a;
8      *p = point(p);
9      cout << *p << endl;
10 }
```