

2023年计算机程序设计（智）笔试题解析

创作：人工智能2402 韩子慕 人工智能2402 关舟涵

审核：人工智能2402 沈子毅

联系方式：zimuhan276@gmail.com

选择题

选择第1题

答案：A

考察变量名的规则：只能有字母、数字、下划线。数字不能开头。因此本题选A。

选择第2题

答案：B

考察字符常量的命名规则。字符常量用单引号，且只能是一个字符，因此ACD都是错的。B是转义字符，转义字符是一类特殊的字符常量，如 `\t`，`\n` 等。

选择第3题

答案：A

注意默认下取整；`a = (int) b * 4` 表示先进行强制转换再与4相乘；而 `a = (int) (b * 4)` 表示先与4相乘再进行强制转换。

选择第4题

答案：BD

学组讨论后认为，这道题出错了

在写代码时，变量之间的乘号是不可以省略的，B违反了这一点。

D没有初始化，应该先对 a 进行初始化，再加上 b 的值。【此处可能有疑问？如果 a 已经事先初始化了呢？那么也不对，因为对变量的声明不能重复，如果之前对变量初始化过，那么这里就不能写 `int`。】

A是最简单的创建变量并初始化；C是一个三目运算符，这表示创建整型变量 d，如果 c 等于8，那么0赋值给 d，否则1赋值给d。

选择第5题

答案：没有

学组讨论后认为，这道题也出错了

这道题考察运算符的运算顺序。先进行逻辑运算，再进行赋值运算。先运算这个三目运算符，b不等于1，因此执行 `--c`，c的值更新为2，再把2赋值给c。因此a, b, c分别等于8, 2, 2，本题没有正确答案。

选择第6题

答案：没有

本题考察变量的作用域。一对大括号表示一个作用域，同一作用域内的变量名不可以重复，但是作用域内外的变量名可以重复。程序从上到下顺序执行，第一个输出的 a 是初始化为99的 a，所以第一个输出99；第二个输出的 a 在作用域内没有创建，所以默认使用主函数内创建的 a，即初始化为8的 a，所以输出11；第三个在作用域内创建了一个 a，并把一个逗号表达式的值赋给了它。逗号表达式的结果是最后的式子的结果，故 a 的初始值就是4；第四个输

出的 `a` 就是在主函数块作用域内的 `a`，但是，第二次操作的时候已经讲主函数块作用域内的 `a` 修改为11，所以经过 `a--` 后，输出的 `a` 应该为10，所以这题也没有正确答案。

选择第7题

答案：B

本题考察while循环和do-while循环，注意区别这两种循环。while循环是先判定条件，再决定是否执行，而do-while循环是先进行循环，再来判定是否进行下次循环。理解了这一点，本题就能迎刃而解，算数的细节这里不必多言。

选择第8题

答案：C

A: for循环的三个条件都可以缺省，这可能给程序带来麻烦，但在语法上是没有问题的。

B: 有关do while语句和while语句的区别参看第7题。

C和D: break终止整个循环，而continue结束本次循环。

选择第9题

答案：D

本题考察指针与数组。谨记：数组名就是指向数组首地址的指针，所以在此题中 `a` 和 `p` 的效力是一样的，因此 `a[i]` 就是 `p[i]`，B和C也完全等价（可以理解为指针向后移动*i*个单位，然后取出其内容），这样就可以选出D（D在语法上就不对）。

选择第10题

答案：D

本题考察二维数组的初始化，对于二维数组的行数和列数，可以缺省其行数，但是列数不可缺省，D就错了。数组内的数据可以用大括号分隔，也可以不用大括号分割（系统默认先填满第一个行，再往下填）（因此A, B均对）。一个大括号内的数据在初始化时没有写满的，系统默认初始化为0（C）。

选择第11题

答案：D

在C++中字符可以和整型一起运算，运算时字符取对应的ASCII码的值。因此 `i` 以它的ASCII码105参与运算。另外`c`是整型数据，所以`a/b`的结果要向下取整。

选择第12题

答案：B

A: 函数调用错误，应该改为 `Swap(&a,&b);`。

B: 正确的，这是引用的传递。

C: 值传递，不能改变实参内存中的数据，不能实现调换两个数的作用。

D: 这是错误的引用传递。应把 `Swap(&a, &b)` 改为 `Swap(a,b)`，同时，在Swap函数内，应把 `*x`、`*y` 改为 `x`、`y`。

选择第13题

答案：C

这道题读懂for循环即可，是一个累加器。注意当 `sum > 26` 时循环终止，而遇到5的倍数时只跳过本次循环，而不终止。具体计算在此略去。

选择第14题

答案: B

传递数组时，一般传递数组名（指向数组首地址的指针）和数组的长度（以便子函数确定边界），所以数组名、指向数组首地址的指针、数组的大小都可以作为实参传递。一般不会把数组内所有元素都作为参数传递，这会使内存开销很大，不切实际。

选择第15题

答案: 没有

本题考察switch语句，因为 $x=1$ ， $y=0$ ，本题首先执行的代码是 $a++$ ；，但是在 $\text{switch}(x)$ 的 case 1 下没有 break ，因此 case 2 还要执行。最终结果应为 $a = 3$ ， $b = 0$ ，没有正确答案（不过C或许是答案，可能打错了）。

补充:

21年考了一道很类似的题，也是要注意 case 1 后面没有 break ，因此 case 2 还要执行。

14..下面程序的输出结果是 ()

```
#include "stdio.h"
main() {
    int x=1, y=0, a=0, b=0;
    switch(x) {
        case 1:
            switch(y)
            {
                case 0: a++; break;
                case 1: b++; break;
            }
        case 2: a++; b++; break;}
    printf("a=%d, b=%d\n",a,b);}

```

- A. $a=2, b=1$ B. $a=1, b=1$ C. $a=1, b=0$ D. $a=2, b=2$

选择第16题

答案: C

本题考察指针。指针也是一个变量，它存储的是另一个变量的地址，A错。指针可以指向函数名，B错。指向数组名的指针可以对所有数据操作（通过指针的移动和取内容实现），C对。指向指针的指针（也称二级指针），常用于表示二维数组，而不是一维数组，D错。

选择第17题

答案: C

定义对象时自动调用构造函数对对象进行初始化，不能显式地调用构造函数。在一个类中存在默认的析构函数，因此在类中没有显式地定义析构函数也能完成内存的释放。

选择第18题

答案: D

构造函数不可以被继承, 正因如此, 如果基类有非默认构造函数, 派生类在其构造函数中需要显式调用基类构造函数。

选择第19题

由于24级起, 程设缩减学时, 没有讲虚函数, 所以这个题略了。

选择第20题

答案: CD

这道题可能也出错了

A: 类定义只是定义了一个数据结构, 而并没有为它分配内存。

B: private限定符既可以限定成员属性, 也可以限定成员函数。

C: 默认构造函数是 `A(){}`, 确实没有任何形式参数。

D: 析构函数确实只能有一个, 不能有参数, 也没法重载。

程序分析与编程

第1题

答案:

统计如下:

xiaomin's number

2023001

xiaomin's age

19

xiaomin's gender

m

xiaomin's score

0--9:0

10--19:0

20--29:0

30--29:1

40--49:1

50-59:0

60--69:2

70--79:1

80--89:3

90--99:2

第2题

答案:

(1) `void Swap(int* a, int* b);`

(2) `&a, &b`

(3) `temp = *a;`

- (4) *a = *b;
- (5) *b = temp;

第3题

答案:

- (1) struct Student
 - (2) struct Student * ptr = stu;
 - (3) stu 或 ptr
 - (4) stu 或 ptr
 - (5) tmp
- (3) (4) 传递数组名和传递指针是一样的。

第4题

答案:

初始化全局对象

进入main函数

初始化自动局部对象

调用子函数

进入function()函数

初始化自动局部对象

初始化静态局部对象

进入function()函数

初始化自动局部对象

main函数执行结束

本题注意构造函数的重载，根据传入参数的数量、类型的不同匹配相应的构造函数执行；还要注意静态变量只加载一次，即它的生命周期从创建开始一直到主函数结束，所以第二次调用function时，a3对象不再重复初始化。

第5题

答案:

```
class Line: public Dot
Dot1: x=3 y=8
Dot2: x=2 y=6
Length=2.2360
Center: x=2.5 y=7
```

本题考察面向对象的编程（类的继承，构造函数），实现了给定两点，求出他们之间线段的长度以及中点坐标的功能。本题存在一个问题，作为笔试考题，线段长度为 $\sqrt{5}$ ，这输出结果恐怕还要手动开一会根号，有点麻烦。

第6题

一种写法:

```
MytestClass(float a, float b){  
    x = a;  
    y = b;  
}
```

```
MytestClass(){  
    x = 0;  
    y = 0;  
}
```

```
MytestClass(MytestClass & a){  
    x = a.x;  
    y = a.y;  
}
```

另一种写法:

```
MytestClass(float a = 0.0, float b = 0.0){  
    x = a;  
    y = b;  
}
```

```
MytestClass(MytestClass & a){  
    x = a.x;  
    y = a.y;  
}
```

当然,也可以使用初始化列表的方法完成这道题,例如:

```
MytestClass(float a = 0.0, float b = 0.0): x(a), y(b){  
}
```

```
MytestClass(MytestClass & a): x(a.x), y(a.y){  
}
```

第7题

答案:

```
template <typename T> // 等价于 template <class T>
T findmax(T arr[], int n) {
    int idx = 0;
    for (int i=1; i<n; i++) {
        if (arr[i] > arr[idx]) {
            idx = i;
        }
    }
    return arr[idx];
}
```

第8题

答案:

```
friend float Sum2(Mytest& t) {
    return t.x + t.y;
};
```

第9题

答案:

```
Flag1=1
Flag1=2
Flag1=3
i=50
count=3
Flag2=2
Flag2=1
Flag2=0
```

本题疑似存在问题，count变量并没有初始化，默认count初始化为0可以得到以上结果。

另外注意本题创建了一个对象数组 b[2]，其中包含了两个对象，所以要调用两次构造函数和析构函数。

第10题

答案:

本题较为开放，答案仅供参考，具体函数实现可以个性化处理。

```

#include <iostream>
using namespace std;
class Student{
    char name[50];
    int id;
    float score[10];

public:
    void Set(){
        char a[50]; int b; float arr[10];
        cout << "Please input student's name:" << endl;
        cin >> name;
        cout << "Please input student's id:" << endl;
        cin >> id ;
        cout << "Please input student's score for each subject:"<< endl;
        for(int i = 0; i < 10; i++){
            cin >> score[i];
        }
    }

    void Print(){
        cout << name << '\t';
        cout << id << '\t' << endl;
        for(int i = 0; i < 10; i++){
            cout << score[i] << '\t';
        }
        cout << endl;
    }

    void Average(){
        float ave = 0;
        for(int i = 0; i < 10; i++){
            ave += 0.1*(score[i]);
        }
        cout << ave << endl;
    }
};

int main(){
    Student zhangsan, lisi;
    zhangsan.Set();
    zhangsan.Print();
    zhangsan.Average();
    lisi.Set();
    lisi.Print();
    lisi.Average();
    return 0;
}

```