C++笔记2

标签（空格分隔）： C++ 面试

Q1

```
char a[] = "xyz", b[] = {'x', 'y', 'z'};
if (strlen(a) > strlen(b))
    printf("a > b\n");
else
    printf("a <= b\n");
char a[] = "xyz", b[] = {'x', 'y', 'z'};
```

a容易理解，strlen(a)=3;
b是数组，元素在内存中是连续存储的，而strlen函数求字符串长度是要以'\0'结尾，但是b没有'\0',
strlen的内部函数指针会一直向后搜索，直至找到'\0',内存中的其他区域也是有数据的，只是没有意义，
所以，strlen的指针最后指到哪里无法确定，但是结果肯定大于等于3.

Q2

C++将父类的析构函数定义为虚函数，下列正确的是哪个?
释放父类指针时能正确释放子类对象
C++的多态肯定是使用父类的指针指向子类的对象，所以肯定是释放子类的对象，如果不使用虚函数的话， 父类的指针就只能够释放父类的对象。

Q3

虚函数的作用是实现了继承性。**错（多态性）**。

Q4

```
int a[2][3] = {1,3,5,7,9,11};
```

这里储存的是两个元素为3的数组，即a[0] = {1,3,5};a[1] = {7,9,10};

Q5

instance是java的二元运算符，用来判断他左边的对象是否为右面类（接口，抽象类，父类）的实例

Q6

```
#include <iostream>
using namespace std;
class A
{
public:
    void print()
    {
        cout << "A:print()";
    }
};
class B: private A
{
public:
    void print()
    {
        cout << "B:print()";
    }
};
class C: public B
{
public:
    void print()
    {
A:: print();
    }
};
int main()
{
    C b;
    b.print();
}
```

私有继承，C已经无法调用A的print()函数了。

Q7

```
void test3(){
    char str[10];
    str++;
    *str='0';
}
```

常量数组str，不能执行str++;
这实际的语句就是str=str+1;而str是数组名，数组名是常量，所以不能给常量赋值。（可以执行str+1，但是不能str=.）

Q8

进程间通信方法有：文件映射、共享内存、匿名管道、命名管道、邮件槽、剪切板、动态数据交换、对象连接与嵌入、动态连接库、远程过程调用等

Q9

类方法是指类中被static修饰的方法，无this指针

Q10

一个非空广义表的表尾只能使子表。

Q11

```
#include <iostream>
using namespace std;
struct Foo {
    Foo() {}
    Foo(int) {}
    void fun() {}
};
```

```
int main(void) {
    Foo a(10); //语句1
    a.fun(); //语句2
    Foo b(); //语句3
    b.fun(); //语句4
    return 0;
    16.
}
```

语句4错了，因为语句3是一个函数声明...正确的无参应为Foo b;

Q12

```
#include <iostream>

void GetMemeory(char *p)
{
    p = (char *)malloc(100);
}
void Test()
{
    char *str = NULL;
    GetMemeory(str);
    strcpy(str, "Thunder");
    strcat(str + 2, "Downloader");
    printf(str);
}
```

程序崩溃。原因，参数是值传递，因此p和str指向同一块地方NULL。但是函数又给参数p重新开辟空间，所以改变的是p指向的空间，str没变（还是指向NULL）。所以因为str没有空间存放字符串，导致崩溃。
正确的应该使用双指针。

```
void GetMemory(char **p){
    *p = (char *)malloc(100);
}

void Test(){
    char *str = NULL;
    GetMemory(&str);
    strcpy(str,"Thunder");
    strcat(str+2,"Downloader");
    printf(str);
}
```

输出为Thunderownloader.

Q13

一个类有基类、内部有一个其他类的成员对象，构造函数的执行顺序是:先执行基类的(如果基类当中有虚基类，要先执行虚基类的，都是按照继承时的顺序依次执行)，再执行成员对象的（声明的顺序），最后执行自己的。（切记都不是初始化列表的顺序）

Q14

1.结构体的大小等于结构体内最大成员大小的整数倍,结构体大小必须是所有成员大小的整数倍 .
2、结构体内的成员的首地址相对于结构体首地址的偏移量是其类型大小的整数倍，比如说double型成员相对于结构体的首地址的地址偏移量应该是8的倍数。
3、 为了满足规则1和2编译器会在结构体成员之后进行字节填充!

Q15

C++不是类型安全的语言,内联说明（inline specification）对于编译器来说只是一个建议，编译器可以选择忽略这个建议。

Q16

A： a+=(a++) 先计算a++ ， 因为a为后++,a左边是左值不会报错 ;
B： a+=(++a) 先计算++a,因为a为前++, a左边是左值不会报错 ;
C:：(a++) += a 这个是错误的。因为左值只能是变量,不能是表达式，(a++)是后++, 所以a不会先计算a++,是表达式，。所以会报错。
D：(++a) +=(a++) 先计算++a,然后再去计算a +=(a++) ,所以左边也是左值 ;

Q17

析构函数没有参数列表，无法重载，但是可以重写

Q18

vector是顺序存储的，只有在尾部插入才不会导致迭代器失效，在头部插入或者中间插入都会导致插入的部位

以及其后的所有迭代器都失效；

Q19

非空子串的个数共有n(n+1)/2.

Q20

关于c++的inline关键字,以下说法正确的是()
正确答案: D 你的答案: E (错误)

A 使用inline关键字的函数会被编译器在调用处展开

B 头文件中可以包含inline函数的声明

C 可以在同一个项目的不同源文件内定义函数名相同但实现不同的inline函数

D 定义在Class声明内的成员函数默认是inline函数

E 优先使用Class声明内定义的inline函数

F 优先使用Class实现的内inline函数的实现

A 如果只声明含有inline关键字，就没有内联的效果。 内联函数的定义必须放在头文件中, 编译器才能在调用点内联展开定义. 有些函数即使声明为内联的也不一定会被编译器内联, 这点很重要; 比如虚函数和递归函数就不会被正常内联. 通常, 递归函数不应该声明成内联函数.
B 内联函数应该在头文件中定义，这一点不同于其他函数。编译器在调用点内联展开函数的代码时，必须能够找到 inline 函数的定义才能将调用函数替换为函数代码，而对于在头文件中仅有函数声明是不够的。
C 当然内联函数定义也可以放在源文件中，但此时只有定义的那个源文件可以用它，而且必须为每个源文件拷贝一份定义(即每个源文件里的定义必须是完全相同的)，当然即使是放在头文件中，也是对每个定义做一份拷贝，只不过是编译器替你完成这种拷贝罢了。但相比于放在源文件中，放在头文件中既能够确保调用函数是定义是相同的，又能够保证在调用点能够找到函数定义从而完成内联(替换)。
对于由两个文件compute.C和draw.C构成的程序来说，程序员不能定义这样的min()函数，它在compute.C中指一件事情，而在draw.C中指另外一件事情。如果两个定义不相同，程序将会有未定义的行为：

> 为保证不会发生这样的事情，建议把`inline`函数的定义放到头文件中。在每个调用该`inline`函数的文件中包含该头文件。这种方法保证对每个`inline`函数只有一个定义，且程序员无需复制代码，并且不可能在程序的生命期中引起无意的不匹配的事情。

D 正确。 定义在类声明之中的成员函数将自动地成为内联函数，例如：
1

```
class A { public: void Foo(int x, int y) { ... } // 自动地成为内联函数 }
```

EF 在每个调用该inline函数的文件中包含该头文件。这种方法保证对每个inline函数只有一个定义，且程序员无需复制代码，并且不可能在程序的生命期中引起无意的不匹配的事情。最好只有一个定义！

Q21

1.循环链表是另一种形式的链式存贮结构。它的特点是表中最后一个结点的 指针 域指向 头结点 ，整个链表形成一个环。
（1）单循环链表——在单链表中，将终端结点的指针域NULL改为指向表头结点或开始结点即可。
（2）多重链的循环链表——将表中结点链在多个环上。
2 队列（Queue）是只允许在一端进行插入，而在另一端进行删除的运算受限的线性表；
3. 栈（stack）在计算机科学中是限定仅在栈顶进行插入或删除操作的线性表。
4. “关联数组”是一种具有特殊索引方式的数组。不仅可以通过整数来索引它，还可以使用字符串或者其他类型的值（除了NULL）来索引它。 详情查看： http://baike.baidu.com/link?url=yYrNB5t4PrCvs-XfxfEM0ZZfALpsEi3FYopk1v0BuopUSWOr7mS0Lou8C-SzhDnSuv7BH5vKIoIblvi8GgUmGq
关联数组和数组类似，由以名称作为键的字段和方法组成。 它包含标量数据，可用索引值来单独选择这些数据，和数组不同的是， 关联数组的索引值不是非负的整数而是任意的标量。这些标量称为Keys，可以在以后用于检索数组中的数值。
关联数组的元素没有特定的顺序，你可以把它们想象为一组卡片。每张卡片上半部分是索引而下半部分是数值。
5.链表（Linked list）是一种常见的基础数据结构，是一种线性表，是一种物理存储单元上非连续、非顺序的存储结构。双向链表也叫 双链表 ，是链表的一种，它的每个数据结点中都有两个 指针 ，分别指向直接后继和直接前驱。所以，从双向链表中的任意一个结点开始，都可以很方便地访问它的前驱结点和后继结点。一般我们都构造双向 循环链表 。

Q22

相同类型的类对象是通过拷贝构造函数来完成整个复制过程的

Q23

```
class MyClass
{
public:
    MyClass(int i = 0)
    {
        cout << i;
    }
    MyClass(const MyClass &x)
    {
        cout << 2;
    }
    MyClass &operator=(const MyClass &x)
    {
        cout << 3;
        return *this;
    }
```

```
    ~MyClass()
    {
        cout << 4;
    }
};
int main()
{
    MyClass obj1(1), obj2(2);
    MyClass obj3 = obj1;
    return 0;
}
```

这里obj调用拷贝构造函数，所以输出122444.
若改为

```
MyClass obj3;
obj3 = obj1;
```

则输出1203444.先调用默认构造函数，默认赋值0，在调用复制构造函数，输出3.

Q24

1） 进程间通信方法有：文件映射、共享内存、匿名管道、命名管道、邮件槽、剪切板、动态数据交换、对象连接与嵌入、动态连接库、远程过程调用等

（2） 事件、临界区、互斥量、信号量可以实现线程同步

Q25

在给定文件中查找与设定条件相符字符串的命令为?
grep

Q26

n 个字符构成的字符串，假设每个字符都不一样，问有多少个子串?
长度为1：n
长度为2：n - 1
长度为3：n - 2
长度为4：n - 3
长度为n：1
加上空串

共(n)(n - 1) / 2 + 1;

Q27

某字符串满足:concat(head(s),head(tail(tail(s))))=”ac”,(head,tail的定义同广义表),则S=()
head和tail运算类似lisp
head(a,(b, c, d)) = a
taid(a,(b, c, d)) = ((b,c,d))
concat为拼接
即head(s) == ‘a’
只要第一个为a第三个为c就可以：取两次尾部一次头部

Q28

派生类实例化时，先调用基类的构造函数，然后是派生类的类成员变量构造函数 构造的顺序是按照成员变量的**定义**先后顺序。

Q29

常见的不能声明为虚函数的有:普通函数(非成员函数)、静态成员函数、内联成员函数、构造函数、友元函数。

Q30

Bayes判决函数：
1：P(ω1)P(ω1/x)> P(ω2)P(ω2/x) ==> x∈ω1
2：P(ω2)P(ω2/x)> P(ω1)P(ω1/x) ② ==> x∈ω2