C++期末考试

第一大题：郑老师出

第二大题：程序阅读理解

注意：无需考虑头文件及命名空间，假设均已写妥。

Part 1. 魏林溪

1. 返回数组和指针的区别（结构化部分）

   运行如下的 C++代码，输出是什么？

```
char* GetString1()
{
    char  p[] =  "Hello World";
    return  p;
}

char* GetString2()
{
    char  *p =  "Hello World";
    return  p;
}

void main(){
    cout<<"GetString1 returns:"<< GetString1()<<endl;
    cout<<"GetString2 returns:"<< GetString2()<<endl;
}
```

   答案：输出两行，第一行 GetString1 returns: 后面跟的是一串随机的内容，而第二行 GetString2 returns: Hello World.

   （1） 两个函数的区别在于 GetString1 中是一个数组，而 GetString2 中是一个指针。

   （2） 当运行到 GetString1 时，p 是一个数组，会开辟一块内存，并拷贝"Hello World"初始化该数组。接着返回数组的首地址并退出该函数。由于 p 是 GetString1 内的一个局部变量，当运行到这个函数外面的时候，这个数组的内存会被释放掉。因此在_tmain 函数里再去访问这个数组的内容时，结果是随机的。

   （3） 当运行到 GetString2 时，p 是一个指针，它指向的是字符串常量区的一个常量字符串。该常量字符串是一个全局的，并不会因为退出函数 GetString2 而被释放掉。因此在_tmain 中仍然根据 GetString2 返回的地址得到字符串"Hello World"。

2. 缺省参数（面向对象部分）

   运行如下的 C++代码，输出是什么？

```
class  A
{
```

```
public:
    virtual  void  Fun(int  number = 10)
    {
        cout << "A::Fun with number " << number;
    }
};

class  B:  public  A
{
public:
    virtual  void  Fun(int  number = 20)
    {
        cout << "B::Fun with number " << number;
    }
};

void  main()
{
    B b;
    A &a = b;
    a.Fun();
}
```

答案：输出 B::Fun with number 10

（1） 由于 a 是一个指向 B 实例的引用，因此在运行的时候会调用 B::Fun。但缺省参数是在编译期决定的。在编译的时候，编译器只知道 a 是一个类型 a 的引用，具体指向什么类型在编译期是不能确定的，因此会按照 A::Fun 的声明把缺省参数 number 设为 10。

（2） 这一题的关键在于理解确定缺省参数的值是在编译的时候，但确定引用、指针的虚函数调用哪个类型的函数是在运行的时候。

Part2. 高士翔

3. 二维数组

```
void main()
{
    int m[3][3]={ {1}, {2}, {3} }, n[3][3]={ 1, 2, 3 };

    cout << m[1][0]+n[0][0] << endl;
    cout << m[0][1]+n[1][0] << endl;
}
```

答案：输出两行，第一行：3；第二行：0

4. 虚函数

```
class A{
public:
    virtual void fun1()
```

```
    {
        cout<<"调用 A::fun1()\n";
    }
    void fun2()
    {
        fun1();
    }
};

class B:public A{
public:
    void fun1()
    {
        cout<<"调用 B::fun1()\n";
    }
};

void print(A& a)
{
    a.fun1();
}

void main()
{
    A a,*pa;
    B b;

    b.fun1();

    pa=&a;
    pa->fun1();

    pa=&b;
    pa->fun1();

    print(a);
    print(b);
}
```

答案：输出 5 行

调用 B::fun1()
调用 A::fun1()
调用 B::fun1()
调用 A::fun1()
调用 B::fun1()

Part3. 黄韬

5. 数组，字符串的比较

```
void main()
{
    char str1[] = "abc";
    char str2[] = "abc";

    const char str3[] = "abc";
    const char str4[] = "abc";

    const char *str5 = "abc";
    const char *str6 = "abc";

    char *str7 = "abc";
    char *str8 = "abc";

    cout << ( str1 == str2 ) << endl;
    cout << ( str3 == str4 ) << endl;
    cout << ( str5 == str6 ) << endl;
    cout << ( str7 == str8 ) << endl;
}
```

答案：0011

str1,str2,str3,str4 是数组变量，它们有各自的内存空间；而 str5,str6,str7,str8 是指针，它们指向相同的常量区域。const 的目的是混淆视听。

6. 数组，指针

```
void main()
{
    int a[5]={1,2,3,4,5};
    int *ptr=(int *)(&a+1);
    cout << *(a+1) << ", " << *(ptr-1) << endl;
}
```

答案：2,5

(a+1)就是 a[1]，(ptr-1)就是 a[4]，执行结果是 2，5。&a+1 不是首地址+1，系统会认为加一个 a 数组的偏移，是偏移了一个数组的大小(本例是 5 个 int)。

Part4. 秦弋戈

7. 成员变量初始化

```
class base
{
private:
    int m_i;
    int m_j;
public:
```

```
    base( int i ) : m_j(i),m_i(m_j) {}
    base() : m_j(0),m_i(m_j){}
    int get_i() {return m_i;}
    int get_j() {return m_j;}
};

void main ()
{
    base obj(98);
    cout << obj.get_i() <<endl<< obj.get_j() <<endl;
}
```

答案：输出结果第一个为随机数，第二个是 98。

解析：本题想得到的结果是“98，98”。但是成员变量的声明是先 m_i ，然后是 m_j；初始化列表的初始化变量顺序是根据成员变量的声明顺序来执行的，因此，先初始化 m_i，但此时 m_j 还未初始化，m_i 会被赋予一个随机值。改变一下成员变量的声明顺序可以得到预想的结果。

8. 变量值

```
void main()
{
    int x = 9999;
    int countx = 0;
    while(x){
        x = x & (x-1);
        countx++;
     }
   cout << countx << endl;
}
```

答案：8

解析：countx 是把 x 转化为二进制之后含有 1 的个数.

第三大题：设计题

描述一个场景，要求给出设计类图。（可以考虑套用某个设计模式，他们学过的有工厂、迭代器，也可以选择他们未学过的，批卷时根据设计的优良程度给分）