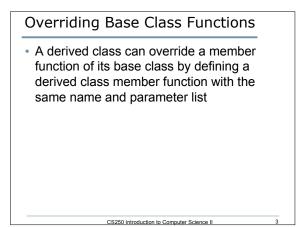


| | ss Base | |
|----|--|--|
| { | | |
| p | ublic: | |
| | <pre>Base() {cout <<"Entering the base.\n";}</pre> | |
| | Base(char *str) { cout << "This base is " | |
| | << str << ".\n"; } | |
| | <pre>~Base() {cout << "Leaving the base.\n";}</pre> | |
| }; | | |
| | ss Camp : public Base | |
| { | ublic: | |
| р | Camp() { cout << "Entering the camp.\n";} | |
| | Camp() { Cout << "Entering the Camp. \n"; } Camp(char *str1, char *str2) : Base(str1) | |
| | { cout << "The camp is " << str2 << ".\n";} | |
| | <pre>~Camp() {cout << "Leaving the camp.\n";}</pre> | |
| }; | "camp() (cout << heaving the camp. (n ,) | |
| | main() | |
| { | | |
| с. | <pre>amp cOutpost("secure", "secluded"); eturn 0;</pre> | |
| | ecuin o, | |



| Example | | | |
|---|--|--|--|
| class Person | | | |
| { | | | |
| private: | | | |
| <pre>string name;</pre> | | | |
| public: | | | |
| <pre>Person() { setName(""); }</pre> | | | |
| <pre>Person(string pName) { setName(pName); }</pre> | | | |
| <pre>void setName(string pName) { name = pName; }</pre> | | | |
| <pre>string getName() { return name; }</pre> | | | |
| }; | | | |
| | | | |
| | | | |

CS250 Introduction to Computer Science II

```
class Faculty : public Person
{
    private:
    Discipline department;
    public:
    Faculty(string fname, Discipline d)
       {setName(fname); setDepartment(d); }
    void setDepartment(Discipline d)
       { department = d; }
    Discipline getDepartment()
       { return department; }
};
```

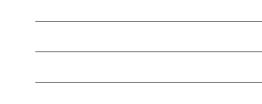
CS250 Introduction to Computer Science II

CS250 Introduction to Computer Science II

5

6





What is the Output

int main ()

TFaculty cTFaculty("Khoja", COMPUTER_SCIENCE, "DR."); cout << cTFaculty.getName() << endl;</pre>

Faculty *pAdviser = new Faculty("Williams, COMPUTER_SCIENCE); cout << pAdvisor->getName() << endl; return 0;

}

ł

• List all of the functions that are called. Include the class name.

CS250 Introduction to Computer Science II

Type Compatibility

- Objects of a derived class can be used wherever objects of a base class object are expected
- Rules for pointers and objects:
 - A derived class pointer can always be assigned to a base class pointer
 - A type cast is required to perform the opposite assignment

CS250 Introduction to Computer Science II

8

9

- This could cause an ERROR!!!

Example class Base { public: int i; Base(int k) {i = k;} }; class Derived : public Base { public: double d; Derived(int k, double g) : Base(k) { d = g;} };

CS250 Introduction to Computer Science II

Which are allowed?

- Base *pb = new Base(5);
- Derived *pd = new Derived(6, 10.5);
- Base *pb1 = pd;
- Base *pb2 = new Derived(7, 11.5);
- Derived *pd1 = static_cast<Derived *>(pb1);

CS250 Introduction to Computer Science II

10

11

- cout << pd1->d;
- pd = static_cast<Derived *>pb;
- cout << pd->d;

| Wha | it is the Output? |
|------------|--|
| class | ase |
| { nrot | cted: |
| | baseVar; |
| publ | c: |
| | e(int val = 2) { baseBar = val; } ; getVar() { return baseVar; } |
| | erived : public Base |
| { | - |
| priv | te: : deriVar: |
| nı ldug | |
| De | ived(int val = 100) { deriVar = val; } getVar() { return deriVar; } |
| }; | |
| int ma | n() |
| | *pObject; |
| | ed object; |
| | ct = & object; |
| retu | << pObject->getVar() << endl; |
| } Ietu | , |