Checkpoints 15.9 - 15.19

15.9 Explain the difference between overloading a function and redefining a function.

To overload is to add a specialization for a specific type signature. To redefine is to replace an existing implementation with a newer, better, less tested one.

15.10 Explain the difference between static binding and dynamic binding.

15.11 Are virtual functions statically bound or dynamically bound?

15.12 What will the following program display?

```
#include <iostream>
using namespace std;

class First {
protected:
  int a;
public:
  First(int x = 1) { a = x; }
  int getVal() { return a; }
};

class Second : public First {
private:
  int b;
public:
  Second(int y = 5) { b = y; }
  int getVal() { return b; }
};

int main() {
  First object1; Second object2;
  cout << object1.getVal() << endl;
  cout << object2.getVal() << endl;
  return 0;
}
```

1
5

15.13 What will the following program display?

```
#include <iostream>
using namespace std;
```

```
class First {
protected:
  int a;
public:
  First(int x = 1) { a = x; }
  void twist() { a *= 2; }
  int getVal() { twist(); return a; }
};

class Second : public First {
private:
  int b;
public:
  Second(int y = 5) { b = y; }
  void twist() { b *= 10; }
};

int main() {
  First object1; Second object2;
  cout << object1.getVal() << endl;
  cout << object2.getVal() << endl;
  return 0;
}
```

```
2
2
```

15.14 What will the following program display?

```
#include <iostream>
using namespace std;

class First {
protected:
  int a;
public:
  First(int x = 1) { a = x; }
  virtual void twist() { a *= 2; }
  int getVal() { twist(); return a; }
};

class Second : public First {
private:
  int b;
public:
  Second(int y = 5) { b = y; }
  virtual void twist() { b *= 10; }
};

int main() {
  First object1; Second object2;
  cout << object1.getVal() << endl;
  cout << object2.getVal() << endl;
  return 0;
}
```

2
1

15.15 What will the following program display?

```
#include <iostream>
using namespace std;

class Base {
protected:
  int baseVar;
public:
  Base(int val = 2) { baseVar = val; }
  int getVar() { return baseVar; }
};

class Derived : public Base {
private:
  int derivedVar;
public:
  Derived(int val = 100) { derivedVar = val; }
  int getVar() { return derivedVar; }
};

int main() {
  Base *optr; Derived object;
  optr = &object;
  cout << optr->getVar() << endl;
  return 0;
}
```

2

15.16 Does the following diagram [omitted] depict multiple inheritance or a chain of inheritance?

Chain.

15.17 Does the following diagram [omitted] depict a chain of inheritance or multiple inheritance?

Multiple. Kinky.

15.18 Examine the following classes. The table lists the variables that are members of the Third class (some are inherited). Complete the table by filling in the access specification cation each member will have in the Third class. Write “inaccessible” if a member is inaccessible to the Third class.

```
class First {
private:
  int a;
protected:
  double b;
public:
  long c;
};

class Second : protected First {
private:
  int d;
protected:
  double e;
public:
  long f;
};

class Third : public Second {
private:
  int g;
protected:
  double h;
public:
  long i;
};
```

Member Variable	Access Specification in Third Class
a	inaccessible
b	protected
c	protected
d	inaccessible
e	protected

Member Variable	Access Specification in Third Class
f	public
g	private
h	protected
i	public

15.19 Examine the following class declarations:

```
class Van {
protected:
  int passengers;
public:
  Van(int p) { passengers = p; }
};

class FourByFour {
protected:
  double cargoWeight;
public:
  FourByFour(float w) { cargoWeight = w; }
};
```

Write the declaration of a class named `SportUtility`. The class should be derived from both the `Van` and `FourByFour` classes above. (This should be a case of multiple inheritance, where both `Van` and `FourByFour` are base classes.)

```
class SportUtility : public Van, public FourByFour {
public:
  SportUtility(int p,float w) : Van(p), FourByFour(w) { }
};
```

Find the Errors 54 - 58

53. Line 1, s/\,/\:/g (comma should be a colon). lol colon. speaking of colons, missing a semicolon at the end of the class definition.
54. Line 1 is so messed up, it should read as "`class Truck : public Vehicle`" I'm not even sure where the protected word is going, so I'm just cutting it out.
55. The second comma on line 7 should be a colon. I fear for the keyboard of whoever typed this.
56. Line 6, `numSeats` has not been assigned a value yet. Consider using n instead.
57. Line 10, duplicate method definition.
58. Line 1, second colon should be a comma. Line 6, third comma should be a colon.