1. Consider the following code:

// Foo.h

class Foo

{

public:

Foo(int x);

virtual ~Foo();

virtual int GetX() const = 0;

protected:

int\* mX;

};

// Bar.h

#include "Foo.h"

class Bar : public Foo

{

public:

Bar(int x);

virtual ~Bar();

int GetX() const;

};

// Baz.h

#include "Foo.h"

class Baz : public Foo

{

public:

Baz(int x);

virtual ~Baz();

int GetX() const;

};

// Foo.cpp

#include "Foo.h"

Foo::Foo(int x)

{

mX = new int(x);

}

Foo::~Foo()

{

delete mX;

}

// Bar.cpp

#include "Bar.h"

Bar::Bar(int x)

: Foo(x)

{

}

Bar::~Bar()

{

}

int Bar::GetX() const

{

return \*mX;

}

// Baz.cpp

#include "Baz.h"

Baz::Baz(int x)

: Foo(x)

{

}

Baz::~Baz()

{

}

int Baz::GetX() const

{

return \*mX \* 2;

}

What is printed after the following code?

Foo\* f = new Bar(5);

Foo\* f2 = new Baz(5);

std::cout << f->GetX() + f2->GetX() << std::endl;

1. What does the following print?

std::cout << std::setw(15) << std::setfill('-') << std::internal << "Hello World";

1. Consider the following code.

// Foo.h

class Foo

{

public:

Foo(int x);

virtual ~Foo();

Foo operator+(const Foo& other);

private:

int mX;

};

// Foo.cpp

#include "Foo.h"

Foo::Foo(int x)

: mX(x)

{

}

Foo::~Foo()

{

}

Foo Foo::operator+(const Foo& other)

{

int x = (mX + other.mX) \* 2;

Foo temp(x);

return temp;

}

What is the value of mX of sum after the following code?

Foo t1(2);

Foo t2(3);

Foo sum = t1 + t2;

1. What is the printed after the following code?

float\* x = new float(12.456f);

int\* y = reinterpret\_cast<int\*>(x);

float\* z = reinterpret\_cast<float\*>(y);

std::cout << \*z << std::endl;

1. Consider the following code.

// Foo.h

class Foo

{

public:

Foo();

virtual ~Foo();

virtual void DoSomething();

protected:

static int mX;

};

// Bar.h

#include "Foo.h"

class Bar : public Foo

{

public:

Bar();

virtual ~Bar();

void DoSomething();

};

// Foo.cpp

#include "Foo.h"

Foo::Foo()

{

}

Foo::~Foo()

{

}

void Foo::DoSomething()

{

mX++;

}

// Bar.cpp

#include "Bar.h"

Bar::Bar()

{

}

Bar::~Bar()

{

}

void Bar::DoSomething()

{

mX += 2;

}

What is the value of mX of f1 and f2 after the following code?

Foo\* f1 = new Foo();

Foo\* f2 = new Bar();

f1->DoSomething();

f2->DoSomething();

1. Consider the following code.

void Swap(int& x, int& y)  
{  
 int tmp = x;

x = y;  
 y = tmp;  
}  
  
int main()  
{  
 int a = 14;  
 int b = 3;  
 swap(a, b);

std::cout << a << std::endl;

return 0;  
}

7. Consider the following code.  
  
string MirrorString(const string& str)  
{  
 string mirrorStr;  
  
 for(int i = str.length() - 1; i >= 0; --i)  
 {  
 mirrorStr += str[i];  
 }  
 return mirrorStr;  
}

int main()  
{  
 string line = “What”;  
 line += “up”;  
 line += MirrorString(line) + “bro”;  
 cout << line << endl;

}

What is the printed?

8. Consider the following code.

// Animal.h  
 class Animal  
 {  
 public:  
 Animal(const char\* name);  
 ~Animal();

virtual void PrintName() const = 0;

Protected:  
 const char\* mName;  
};

// Cat.h  
class Animal;

class Cat : public Animal  
{  
Public:  
 Cat(const char\* name);  
 ~Cat();

void PrintName() const;  
};

// Animal.cpp

#include "Animal.h"

Animal::Animal(const char\* name)  
 : mName(name)  
{  
}

Animal::~Animal()  
{  
 delete mName;  
}

// Cat.cpp

#include "Cat.h"  
#include <iostream>

Cat::Cat(const char\* name)  
 : Animal(name)  
{  
}

Cat::~Cat()  
{  
}

void Cat::PrintName() const  
{  
 std::cout << "I'm a cat named " << mName << std::endl;  
}

1. Does the following code have any error? If it has compiler error, fix it. if it has runtime error, explain why. If there's no error, say what the output will be.

Cat\* a = new Cat("Kitty");  
Cat b(\*a);  
delete a;  
b.PrintName();

1. implement a << operator to replace PrintName() function. You should be able to use the operator this way:

Cat cat("Kitty");  
 std::cout << Cat << std::endl;

9. What's the binary representation of variable b in the following code?

a)

float a = 3.f;  
int b = static\_cast<int>(a);

b)  
  
int a = 1;  
int b = reinterpret\_cast<float>(a);

10. Consider the following code.

// Cat.h

class Cat  
{  
public:  
 Cat();  
 int GetCount() const;  
private:  
 static int mCount;  
};

// Cat.cpp

#include "Cat.h"  
  
int Cat::mCount = 11;

Cat::Cat()  
{  
 mCount++;  
}  
  
int Cat::GetCount() const  
{  
 return mCount / 3;  
}

What does the following print?

Cat\* cat = new Cat();  
 Cat\* cat1 = new Cat();

std::cout << cat1->GetCount() << std::endl;

11. What is the value of bEqual?

a)

std::string str1("Java");  
std::string str2("Java");  
bool bEqual = (str1 == str2);

b)

std::string str1("Java");  
std::string str2("Java");  
bool bEqual = (str1.c\_str() == str2.c\_str());

12. Consider the following code.

class Vector  
{  
public:  
 int mX;  
 int mY;  
};

Vector operator\*(Vector& v, int x)  
{  
 Vector a;  
 a.mX = mX \* x;  
 a.mY = mY \* x;  
 return a;  
}

Vector operator/(Vector& v, float x)  
{  
 Vector a;  
 a.mX = mX / x;  
 a.mY = mY / x;  
 return a;  
}

Determine what is printed

Vector v1;  
 v1.mX = 1;  
 v1.mY = 2;  
  
 v2 = v1 \* 2;

std::cout << v2.mX << std::endl;

Vector v1;  
 v1.mX = 1;  
 v1.mY = 2;  
  
 v2 = v1 / 2.f;

std::cout << v2.mY << std::endl;

13. Assuming buf is a valid pointer, what is the problem in the code below? What would be an alternate way of implementing this that would avoid the problem?

size\_t sz = buf->size();

while ( --sz >= 0 )  
{  
 /\* do something \*/  
}

Doesn’t work because the for loop might make sz to -1 which cannot be since its size\_t. Cause infin loop.

14. What is the output of the following code:

#include <iostream>

class Base  
{  
 virtual void method() {std::cout << "from Base" << std::endl;}  
  
public:  
 virtual ~Base() {method();}  
 void baseMethod() {method();}  
};

class A : public Base  
{  
 void method() {std::cout << "from A" << std::endl;}

public:  
 ~A() {method();}  
};

int main()  
{  
 Base\* base = new A;  
 base->baseMethod();  
 delete base;  
  
 return 0;  
}

15. What is the problem with the following code, and how would you fix it?

class A  
{  
public:  
 A() {}  
 ~A(){}  
};

class B: public A  
{  
public:  
B():A(){}  
 ~B(){}  
};

int main()  
{  
 A\* a = new B();  
 delete a;  
}

16. What will be the output of the following program?

#include <iostream>  
  
class A  
{  
public:  
 int mData[2];

A(int x, int y) {mData[0] = x, mData[1] = y;}  
 virtual void f() {}  
};

int main()  
{  
 A a(10, 20);  
 int \*arr = (int\*)&a;

std::cout << arr[2] << std::endl;

return 0;  
}