CPA Chapter 6 Practice Quiz

C++ Institute Volunteer Program 2015-2016

AUTHOR:	ADRIAN NECULA	ADRIAN.NECULA@LIVE.COM
AUTHOR'S BIO:	I am working as a C/C++ programmer at Siemens	

Chapter: 6	Inheritance		
Section: 1,2,3			
C++ Associate (CPA)	Chapter: 6	Section: 1,2,3	Question type: Multiple-choice
Subject: Inheritance and Static Cast			Question Number: 1

Question: Which of the following statements are true? (assumption: all #include and the rest of the code are correct)

```
 class Music{
 public:
    virtual void MusicType(){ cout<<"Silent"; }
 };
 class Rock: public Music{
 public:
    int fans;
    Rock():fans(9999){};
     void MusicType(){ cout<<"Rock"; }
 };
 int main(int argc, char** argv) {
    Music *m1 = new Music();
    Rock *r1;
    r1 = static_cast<Rock*>(m1);
    if (r1) {
       r1->MusicType();
    }
    return 0;
}
```

Answers:

A) The code will generate a compiler error
B) The code will generate a runtime error
C) The value of “r1->fans “ will be “9999”
D) Variable a violates the encapsulation principle
E) The output of the program will be “Silent”
F) The output of the program will be “Rock”
G) The value of the “r1->fans” cannot be determined at compile time

Chapter: 6	Inheritance		
Section: 1,2,3			
C++ Associate (CPA)	Chapter: 6	Section: 1,2,3	Question type: Single-choice
Subject: Static Cast			Question Number: 2

Question: What is the output of the following code fragment? (assumption: all #include and the rest of the code are correct)

```
    class Music{
    public:
       virtual void MusicType(){ cout<<"Silent"; }
    };
    class Rock: public Music{
    public:
       int fans;
       Rock():fans(9999){};
        void MusicType(){ cout<<"Rock"; }
     };
    class Classical: public Music{
    public:
       void MusicType(){ cout<<"Classical"; }
     };
    int main(int argc, char** argv) {
      Classical *c1 = new Classical();
      Rock *r1 = static_cast<Rock*>(c1);
       if (r1) {
          r1->MusicType();
      }
       return 0;
}
```

Answers:

A) The code will generate a compiler error
B) The code will generate a runtime error
C) Classical
D) Rock
E) Silent
F) None of the above

Chapter: 6	Inheritance		
Section: 1,2,3,4			
C++ Associate (CPA)	Chapter: 6	Section: 1,2,3,4	Question type: Single-choice
Subject: Inheritance and Dynamic Cast			Question Number: 3

Question: What is the output of the following code fragment? (assumption: all #include and the rest of the code are correct)

```
   class Music{
   public:
      virtual void MusicType(){ cout<<"Silent"; }
   };
   class Rock: public Music{
   public:
       void MusicType(){ cout<<"Rock"; }
       void Instrument(){ cout<<"Electric Guitar\n"; }
   };
   class Classical: public Music{
   public:
      void MusicType(){ cout<<"Classical"; }

   };
   int main(int argc, char** argv) {
      Music *m1 = new Music(), *m2 = new Classical();
      Classical *c1 = new Classical(), *c2;
      Rock *r1 = dynamic_cast<Rock*>(c1);
      Rock *r2 = dynamic_cast<Rock*>(m1);
      c1 = dynamic_cast<Classical*> (m2);
      c2 = dynamic_cast<Classical*> (m1);
      if (r1 || r2 ) {
         cout<<"||"; r1->MusicType();
      }
      if (c1 && c2){
         cout<<"&&";  c2->MusicType();
      }
      if (c1 || c2){
         cout<<"||"; c1->MusicType();
      }
      return 0;
}
```

Answers:

A) ||Rock||Classical
B) ||Rock&&Classical||Classical
C) ||Classical
D) &&Classical||Classical

Chapter: 6	Inheritance		
Section: 5	Various Supplements		
C++ Associate (CPA)	Chapter: 6	Section: 5	Question type: Fill the blanks
Subject: Copy constructor and Compositions			Question Number: 4

Question: The Engine class copy constructor is called ... times. The CarBody class copy constructor is called ... times. The Car class copy constructor is called ... times.

```
class Engine{
public:
   Engine(){};
   Engine(const Engine &a){ }
};
class CarBody{
public:
   CarBody(){}
   CarBody(const CarBody &b){ }
};
class Car{
public:
   Engine e;
   CarBody cb;
   Car(){}
   Car(const Car &c):cb(c.cb),e(c.e){ }
};
void func(Engine e , CarBody cb , Car c){}
void func(Engine e, CarBody *cb =  NULL , Car *c = NULL){}
int main(){
   Engine e1, e2 = e1, *e3 = &e2;
   CarBody cb1, cb2 = cb1, &cb3 = cb1;
   Car c1, c2 = c1;
   func(e1,cb1,c1);
   func (e1, &cb1);
   return 0;
}
```

Answers: [see question instructions above]

Chapter: 6	Inheritance		
Section: 1,2,3,4			
C++ Associate (CPA)	Chapter: 6	Section: 1,2,3,4	Question type: Single-Choice
Subject: Inheritance and Casts			Question Number: 5

Question: What is the output of the following code fragment? (assumption: all #include and the rest of the code are correct).

```
class Base{
public:
   Base(){ cout<<"A";}
  virtual void F(){ cout<<"A";}
};
class Derived: public Base{
public:
   Derived(){cout<<"B";}
   virtual void G(){cout<<"B";}
};
int main(){
   Base *b1 = new Derived();
   Base *b2 = new Base();
   Derived *d2 = static_cast<Derived*>(b2);
   Derived *d3 = dynamic_cast<Derived*>(b2);
   Derived *d4 = dynamic_cast<Derived*> (b1);
   if (d3 == NULL){
      cout<<"A";
      d4->F();
   }
   else{
      cout<<"B";
      d2->G();
   }
   return 0;
}
```

Answers:

A) ABAAA

B) Runtime error

C) ABABB

D) BAAA

Chapter: 6	Inheritance		
Section: 6,7			
C++ Associate (CPA)	Chapter: 6	Section: 6,7	Question type: Fill the blanks
Subject: Constants and friend functions			Question Number: 6

Question: The line/lines with the number/numbers ... should be commented so the program will compile. After the lines were commented the output off the program will be ...

```
class Test{
 public:
    Test():i(7){}
    int GetI(){ return i;}
    void SetI(int i) {this->i = i;}
    friend int main();
 private:
    int i;
 };
int main(){
   Test *val = new Test();
   const Test *c1 = val;
   Test* const c2 = val;
   /*1*/ val->SetI(3);
   /*2*/ cout<<c2->GetI();
   /*3*/ c2->i+3;
   /*4*/ cout<<c1->GetI();
   /*5*/ c2->i++;
   /*6*/ c1 = NULL;
   /*7*/ cout<<c2->GetI();
   /*8*/ c2 = NULL;
   return 0;
 }
```

Answers: [see question instructions above]

Correct answers:
Q1 - D,E,G

Explanation: D,E,G are correct because: The “static_cast” operator will successfully downcast the “m1” pointer variable but because the derived class defines new functionality that is not present in the base class the call to “r1->fans” will point to an uninitialized block of memory. The member variable “fans” is public (it violates the encapsulation principle were the variable should be “private” and we should have defined getters and setters for the variable).

Correct answers:
Q2 - A

Explanation: A: “static_cast” operator allows cast only between related types.

Correct answers:
Q3 - C

Explanation: C: “c1” - Only variable that has a value different from NULL

Correct answers:
Q4 - Engine class copy constructor is called 5 times. The CarBody class copy constructor is called 4 times. The Car class copy constructor is called 2 times.

Explanation: The copy constructor is not called when we create a reference or a pointer to an object. Also the copy constructor is called when parameters are passed to functions by value (but in no invoked when parameters are passed by reference or pointer).

Correct answers:
Q5 - A

Explanation:
The down casting of ‘b2’ to ‘d3’ is not safe (so the “dynamic_cast” operator returns NULL). Then the program doesn’t go on the else branch (so a runtime error is avoided).

Correct answers:
Q6 - The line/lines with the number/numbers 4, 8 should be commented so the program will compile. After the lines were commented the output off the program will be 34

Explanation:
”c1” is a pointer to a constant so the call to “GetI()” will generate a compiler error unless you add the “const” keyword to it. “c2” is a constant pointer so you cannot change it value after it had been initialized.

AUTHOR:	PRACHI PODDAR	PRACHI.PODDAR108@GMAIL.COM
AUTHOR'S BIO:	Prachi works at EdgeVerve, India as a product engineer (research and development). Her areas of interest are big data analysis, databases and conceptual programming in C, C++ & JAVA. Her hobbies are playing tennis and reading books.	

Chapter: 6	Inheritance		
Section: 4	Objects as parameters and dynamic casting		
C++ Associate (CPA)	Chapter: 6	Section: 4	Question type: single-choice
Subject: Dynamic casting			Question Number: 1

Question: Which of the following concepts means determining at runtime what method to invoke?

A) Data Hiding
B) Dynamic Typing
C) Dynamic binding
D) Dynamic loading

Chapter: 6	Inheritance		
Section: 2	Classes, inheritance and type compatibility		
C++ Associate (CPA)	Chapter: 6	Section: 2	Question type: single-choice
Subject: Inheritance in C++			Question Number: 2

Question: What happens when we try to compile the class definition in the following code snippet?

```
Class Birds{};
Class Peacock : protected Birds{};
```

A) It will not compile because class body of Birds is not defined.
B) It will not compile because class body of Peacock is not defined.
C) It will not compile because a class cannot be protectedly inherited from other class.
D) It will compile successfully.

Chapter: 6	Inheritance		
Section: 5	Various supplements		
C++ Associate (CPA)	Chapter: 6	Section: 5	Question type: single-choice
Subject: Relationship among classes			Question Number: 3

Question: What does the class definition in following code represents?

```
Class Bike
{
    Engine objEng;
};
Class Engine
{
float CC;
};
```

A) Kind of relationship
B) Has a relationship
C) Inheritance
D) None of these

Chapter: 6	Inheritance		
Section: 2	Classes, inheritance and type compatibility		
C++ Associate (CPA)	Chapter: 6	Section: 2	Question type: single-choice
Subject: Inheritance in C++			Question Number: 4
Question: Which of the following statements is correct when a class is inherited privately?			
A) Public members of the base class become protected members of derived class. B) Public members of the base class become private members of derived class. C) Private members of the base class become private members of derived class. D) Public members of the base class become public members of derived class.			

Chapter: 6	Inheritance		
Section: 7	Friendship in the “C++” world.		
C++ Associate (CPA)	Chapter: 6	Section: 7	Question type: single-choice
Subject: Friend keyword			Question Number: 5
Question: If class A is friend of class B and if class B is friend of class C, which of the following is true?			
A) class C is friend of class A B) class A is friend of class C C) class A and class C do not have any friend relationship D) None of the above			

ANSWER KEY

Correct answers: **Q1 - C**
Explanation: no explanation
Correct answers: **Q2 - D**
Explanation: no explanation
Correct answers: **Q3 - B**
Explanation: no explanation
Correct answers: **Q4 - B**
Explanation: no explanation
Correct answers: **Q5 - C**
Explanation: no explanation

AUTHOR:	ERNESTO RODRIGUEZ REINA ERREINA@GMAIL.COM
AUTHOR'S BIO:	Ernesto Rodriguez Reina is a software developer with more than 10 years of experience coding in C++. LinkedIn: http://www.linkedin.com/profile/view?id=219680343

Chapter: [6]	Inheritance		
Section: [3]	Polymorphism and virtual methods		
C++ Associate (CPA)	Chapter: [6]	Section: [3]	Question type: [Multiple-choice]
Subject: Constructors and virtual methods.			Question Number: [1]

Question: What is the output of the following code?

```
#include <cstdlib>
#include <iostream>

struct A{
  A(){
    f();
  }

  virtual void f(){
    std::cout << "A" << std::endl;
  }
};

struct B: A{
  B(){
    f();
  }

  void f() override{
    std::cout << "B" << std::endl;
  }
};

int main(){
  B b;
  return EXIT_SUCCESS;
}
```

Answers:

1. A
2. B
3. AA
4. AB
5. BB
6. BA

ANSWER KEY

Correct answers:
Q1 - AB

Explanation: Before constructing any object, its bases are constructed first. Even when function *f* is virtual when constructing A the function *f* has not yet being overridden, that happen just after the constructor of A returns. The constructor of A executes the function defined in A and later the constructor of B executes the overridden function *f* defined in B.

AUTHOR:	SRAJAN RATTI	MSFREEDOM911@GMAIL.COM
AUTHOR'S BIO:	I am a Final year student of Amity University India. I am doing my Engineering in BTECH CSE.I love to learn new languages. My hobbies are Playing keyboard, making music ,animation.	

Chapter: [6]	Inheritance		
Section:[2]	Classes, inheritance and type compatibility		
C++ Certified Associate Programmer (CPA)	Chapter: [6]	Section:[2]	Question type: [Multiple-choice]
Subject: [Size of a class]			Question Number: [1]

Question: What is the output of the following program?

```
//assume size of integer to be 4 bytes
#include<iostream>
class Desktop
{
        int i,j;
public:
        static int k;

};
int Desktop::k;

class Xbox :public Desktop {

        int tt;

};

int main()

{       std::cout << sizeof(Xbox);
    return 0;
}
```

Answers:

A. 8.
B. 12 .
C. 16.
D. 4

Chapter: [6]	Inheritance		
Section: [3]	Polymorphism and virtual methods		
C++ Certified Associate Programmer (CPA)	Chapter: [6]	Section:[3]	Question type: [Multiple-choice]
Subject: [Virtual functions]			Question Number: [2]

Question: What is the output of the following code?

```
#include<iostream>

class Shape {

public:
virtual void Parameters()=0;

};
class Rectangle: public Shape {
int a, b;

public:
Rectangle(int aa = 0, int bb = 0) :a(aa), b(bb){}
void Parameters();

};
void Rectangle::Parameters()
{
std::cout <<a << " " << b;
}


int main()
{

Shape *s;
Rectangle *rr=new Rectangle(5,5);
s = rr;
s->Parameters();

}
```

Answers:

A. 5 5 .
B. Garbage Value.
C. Error
D. 0 0.

Chapter: [6]	Inheritance		
Section: [3]	Polymorphism and virtual methods		
C++ Certified Associate Programmer (CPA)	Chapter: [6]	Section:[3]	Question type: [Match the Following]
Subject: [Virtual Destructors]			Question Number: [3]

Question: Match the following in correct order of execution?

```
#include<iostream>
class Shape {
public:
virtual void Parameters()=0;
Shape(){
std::cout << "Constructing Shape\n";
}
virtual ~Shape()
{ std::cout << "~Destroying Shape\n"; }

};

class Rectangle:public Shape {
int a, b;

public:
Rectangle(int aa = 0, int bb = 0) :a(aa), b(bb)
{ std::cout << "Constructing Rectangle\n"; }
~Rectangle()
{
std::cout << "~Destroying Rectangle\n";
}
void Parameters();

};
void Rectangle::Parameters()
{
std::cout<< a << " " << b << "\n";
}

int main()
{Shape* s = new Rectangle(5, 4);
s->Parameters();
delete s;
}
```

Answers:

Constructing Shape	1
~Destroying Shape	3
Constructing Rectangle	4
~Destroying Rectangle	2
5 4	5
	Will Not Execute

Chapter: [6]	Inheritance		
Section: [7]	Friendship in the “C++” world.		
C++ Certified Associate Programmer (CPA)	Chapter: [6]	Section: [7]	Question type: [Multiple-choice]
Subject: [friend functions]			Question Number: [4]

Question: What is the output of the following code?

```
#include <iostream>

class Ams;

class ster {
	ster() { std::cout << "HELLO"; }

public:
	friend ster Ams::createB();
};

class Ams {
public:
	Ams() { std::cout << "RUN"; }

	ster createB() { return ster(); }
};

int main() {
	Ams a;
	ster b = a.createB();
}
```

Answers:

A. HELLORUN.
B. HELLO.
C. RUN.
D. Error

Chapter: [6]	Inheritance		
Section:[2]	Classes, inheritance and type compatibility		
C++ Certified Associate Programmer (CPA)	Chapter: [6]	Section:[2]	Question type: [Multiple-choice]
Subject: [Copy Constructors]			Question Number: [5]

Question: How many times the copy constructor would be called?

```
#include<iostream>
class A
{
public:
A(){ std::cout << "Normal Constructor Called\n"; }

A(const A&ac)//copy constructor

{ std::cout << "Copy Constructor called\n"; }

~A(){ std::cout << "Destructor Called\n"; }
};

void func(A &a)
{
std::cout << "Function CAlled\n";
}

int main()
{A ax;
A y = ax;
func(y);
ax = y;}
```

Answers:

A. 0 .
B. 1 .
C. 2.
D. 3.

Chapter: [6]	Inheritance		
Section: [3]	3-4		
C++ Certified Associate Programmer (CPA)	Chapter: [6]	Section:[3-4]	Question type: [Multiple-choice]
Subject: [Virtual Functions]			Question Number: [6]

Question: What is the output of the following program?

```
#include<iostream>
#include<exception>
class ABC{
public :
virtual void func(){ std::cout << "CBA"; }
};
class BB :public ABC
{
public:
void func(){ std::cout << "HMM"; }
};

int main()
{
ABC *a = new ABC();
try
{
BB&b = dynamic_cast<BB&>(*a);
b.func();
}
catch (std::bad_cast &e)
{
std::cout << "Dynamic Cast Failed";
}}
```

Answers:

A. CBA .
B. HMM .
C. Dynamic Cast Failed.
D. Garbage Value.

Chapter: [6]	Inheritance		
Section:[2-3]	2-3		
C++ Certified Associate Programmer (CPA)	Chapter: [6]	Section:[2-3]	Question type: [Multiple-choice]
Subject: [Incrementing Pointers]			Question Number: [7]
Question: Which statement is not true?			
1.We can create objects of an abstract class. 2.Abstact class can be inherited. 3.Abstract class should have one or more pure virtual functions. 4 .Abstract class is an example of run time polymorphism.			

Chapter: [6]	Inheritance		
Section:[2]	Classes, inheritance and type compatibility		
C++ Certified Associate Programmer (CPA)	Chapter: [6]	Section:[2]	Question type: [Multiple-choice]
Subject:[Constuctors]			Question Number: [8]

Question: What is the output of the following code?

```
#include<iostream>
class A
{
public:
        A(){ std::cout << "Cons A\n"; }
        ~A(){ std::cout << "Destu A"; }
};
class B :public A
{
public:
        B(){ std::cout << "Cons B\n"; }
        ~B(){ std::cout << "Destu B\n"; }
};
class C :public B
{
public:
        C(){ std::cout << "Cons C\n"; }
        ~C(){ std::cout << "Destu C\n"; }
};
int main()
{
        C c;
}
```

Answers:

A. Cons A
Destu A
Cons B
Destu B
Consu C
Destu C

B. Cons A
Cons B
Cons C
Destu A
Destu B
Destu C

C. Cons A
Cons B
Cons C
Destu C
Destu B
Destu A

D. Cons C
Cons B
Cons A
Destu C
Destu B
Destu A

ANSWER KEY

Correct answers: **Q1 - B.**
Explanation: static data members do not contribute in size of an object
Correct answer: **Q2 - A.**
Explanation: -
Correct answer: **Q3 – see below:**
Correct Sequence: Constructing Shape , Constructing Rectangle, 5 4, ~Destroying Rectangle, ~Destroying Shape
Correct Answer: **Q4 – D**
Explanation: There is a compilation error when attempting to declare Ams::createB() a friend of ster. To declare Ams::createB() a friend of ster, the compiler needs to know that that function exists. Since it has only seen the declaration of Ams so far, not the full definition, it cannot know this.
Correct answers: **Q5 - B.**
Explanation: -
Correct answers: **Q6 - C.**
Explanation: - Since the Base object does not contain a complete Child object this conversion will fail.
Correct answers: **Q7 - A.**
Explanation: -
Correct answers: **Q8 - C.**
Explanation: -