JAVA Programming Language Homework Ⅰ - OO concept

Student ID: Name:

1. Which of the following techniques can be used to prevent the instantiation of a class by any code outside of the class?

A. Declare all constructors with a void return type.

B. Declare all constructors using the private access modifier.

C. Do not declare any constructors inside a class definition.

D. Do not include a return statement in the constructor.

E. None of the above.

Answer: B

當建構子被定義成 private，任何的類別，除了此建構子本身的類別，誰都無法透過此建構子建立一個新的物件。

2. Which of the following statements are true?

A. A constructor can invoke the constructor of the direct superclass using the superclass constructor invocation statement “super”.

B. By using constructor invocation statement “this”, a constructor can invoke another constructor of the same class.

C. The constructor invocation statement, “this”, can legally appear anywhere in the constructor body.

D. By using the constructor invocation statement “this”, a constructor can invoke itself.

E. None of the above.

Answer: A, B

關鍵字 this 是指本身的類別，所以可用來呼叫本類別中其他的建構子，但卻不允許用 this 來呼叫本身(建構子)。

呼叫建構子的 this 敘述，必須放在陳述中的第一行。

因為 super 代表父類別或 superclass，所以可用來呼叫父類別的建構子。

3. Given the following Java code:

```
public class Hello {
          String title=””;
          int value;
          public Hello( ) {
            title = title + " World";
            System.out.print(title);
        }
        public Hello(int value) {
            this.value = value;
            title = "Hello";
            this( );
        }
        public static void main(String[] args){
                Hello b = new Hello (5);
        }
}
```

What is the result?

A. Hello

B. Hello World

C. Compilation fails

D. Hello World 5

E. Hello Hello

Answer: C

呼叫建構子的 this 敘述，必須放在陳述中的第一行。

4. Given the following Java code:

```
class Num {
        public static String b( ) { return "One"; }
        public static String b( int i ) { return "Two"; }
        public static String b( int i, int j ) throws Exception { return "Three"; }
        public static void main( String[] args ) {
                System.out.println( b(2) );
        }
}
```

What is the result?

A. One

B. Two

C. Compilation fails

D. Three

E. None of the above

Answer: B

因為 System.out.println(b(2))中的 b(2)和 public static String b(int i)

中的參數列相同。

5. Given the following Java code:

Exhibit:

```
public class SimpleCalc {
        public int value;
        public void calculate( ) { value += 7; }
}
```

And:

```
Public class MultiCalc extends SimpleCalc {
        public void calculate( ) { value -= 3; }
        public void calculate( int multipier) {
                calculate( );
                super.calculate( );
                value *= multipier;
        }
        public static void main(String[] args) {
                MultiCalc calculator = new MultiCalc( );
                calculator.calculate(2);
                System.out.println(" Value is: " + calculator.value);
        }
}
```

What is the result?

A. Value is: 8

B. Compilation fails.

C. Value is: 12

D. Value is: -12

E. The code runs with no output.

Answer: A

從 MultiCalc 類別的 main() 開始，建構出 MultiCalc()，
無傳參數的建構子，呼叫 calculate(int) 方法
再呼叫本類別的 calculate()，使 value = 0 – 3 = -3
再呼叫父類別的 calculate()，使 value = -3 + 7 = 4
最後將 value x 傳遞的參數值 2 = 4 * 2 = 8
再度返回 main() 方法，將該 value 屬性印出 8。

6. Given the following Java code:

```
public class Base {
    public static final String FOO = “foo”;
    public static void main(String[] args) {
        Base b = new Base( );
        Sub s = new Sub( );
        System.out.println(Base.FOO);
        System.out.println(Sub.FOO);
        System.out.println(b.FOO);
        System.out.println(s.FOO);
        System.out.println(((Base)s).FOO);
    }
}
class Sub extends Base {public static final String FOO=”bar”;}
```

What is the result?

A. foofoofoofoofoo

B. foobarfoobarbar

C. foobarfoofoofoo

D. foobarfoobarfoo

E. foofoofoobarbar

Answer: D

Base.FOO 是 class Base 的 static 成員，所以印出 "foo"

Sub.FOO 是 class Sub 的 static 成員，所以印出 "bar"

b.FOO 是 class Base 物件，可以存取 static 成員，所以印出 "foo"

s.FOO 是 class Sub 物件，可以存取 static 成員，所以印出 "bar"

(Base)s 之後成為 Base 物件，呼叫 Base 的 FOO，所以印出 "foo"

7. Given the following Java code:

```
public class TestPoly {
    public static void main(String[] args) {
        Parent p = new Child( );
    }
}

class Parent {
    public Parent( ) {
        super();
        System.out.println(“instantiate a parent”);
    }
}

class Child extends Parent {
    public Child( ) {
        System.out.println(“instantiate a child”);
    }
}
```

What is the result?

A. instantiate a child

B. instantiate a parent

C. instantiate a child
instantiate a parent

D. instantiate a parent
instantiate a child

E. Compilation fails

Answer: D

在 Child 建構子中的程式執行之前，類別 Child 的建構子會先呼叫類別 Parent 的建構子，當 Parent 建構子的程式執行時，會列印出第一行，然後再把控制項回傳給 Child 的建構子。

8. Given the following Java code:

```
public class TestPoly {
    public static void main(String[] args) {
        Parent p = new Child( );
    }
}

class Parent {
    public Parent( ) {
        super();
        System.out.println(“instantiate a parent”);
    }
}

class Child extends Parent {
```

```
15.         public Child( ) {
16.                 System.out.println("instantiate a child");
17.                 super( );
18.         }
19. }
```

What is the result?

A. instantiate a child

B. instantiate a parent

C. instantiate a child
instantiate a parent

D. instantiate a parent
instantiate a child

E. Compilation fails

Answer: E

Line 17 程式會使編譯程式失敗，對 super()的呼叫必須放在建構子的第一行陳述式中。

9. Given the following Java code:

```
class C {
        public static void main(String[] args) {
                A tmp = new B( );
                tmp.m1( );
                tmp.m2( );
                ((B)tmp).m1( );
                ((B)tmp).m2( );
        }
}
class A {public void m1( ) { System.out.println ("A");}}
class B extends A {
        public void m1( ) { System.out.println ("B1");}
        public void m2( ) { System.out.println ("B2");}
        public void m3( ) { System.out.println ("B3");}
        public void m4( ) { System.out.println ("B4");}
}
```

What is the result?

A. AB2B1B2

B. B1B2B1B2

C. Compiler Error

D. Runtime Error

E. None of the above

Answer: C

tmp 的型態為 A，因為類別 A 中沒有函式 m2，所以函式的多型不能應用，亦即無法使用型類為 A 的參照來呼叫類別 B 中的函式 m2。

10. Given the following Java code:

```
1. public class Bootchy {
2.          int botch;
3.          String snootch;
4.          public Bootchy() {
5.                   this(“snootchy”);
6.                   System.out.print(“first ”);
7.          }
8.          public Bootchy(String snootch) {
9.                   this(420, “snootchy”);
10.                  System.out.print(“second ”);
11.         }
12.         public Bootchy(int bootch, String snootch) {
13.                  this.bootch=botch;
14.                  this.snootch = snootch;
15.                  System.out.print(“third ”);
16.         }
17.         public static void main(String[] args){
18.                  Bootchy b = new Bootchy();
19.                  System.out.print(b.snootch +” “+ b.bootch);
20.         }
21. }
```

What is the result?

(A) snootchy 420 third second first

(B) snootchy 420 first second third

(C) first second third snootchy 420

(D) third second first snootchy 420

(E) third first second snootchy 420

Answer: D

建構出 b，以無傳參數的建構子建構 public Bootchy()，

this("snootchy")呼叫本類別建構子 public Boothchy(String snootch)，

this(420, "snootchy")呼叫本類別建構子 public Bootchy(int bootch, String snootch) 然後 bootch = 420; snootch = "snootchy"，印出 "third"，
再度返回 public Boothchy(String snootch)，印出 "second "，
再度返回 public Bootchy()，印出 "first"，
最後回到 main()，印出 "snootchy" 與 420。

11. Given the following Java code:

```
1. class A {
2.         private static int tmp = 1;
3.         static void m(int i) { tmp++; i++;}
4.         public void n(int i) { tmp = tmp + 2;}
5.         static void n() { tmp = tmp + 2;}
6.         public static void main(String[] args) {
7.                 int tmp2 = 3;
8.                 m(tmp2);
9.                 System.out.println(tmp + “,” + tmp2);
10.        }
11. }
```

What is the result?

A. 1, 3

B. 2, 3

C. 1, 4

D. 2, 4

E. Compiler Error

Answer: B

變數 tmp 和方法 m 都被定義為 static，所以能直接被其他類別所存取，第 7 行將 tmp2 設為 3，第 8 行呼叫 m 並將 tmp 由 1 增加到 2，而函式的參數 i 並不影響 tmp2 的值，所以結果為 2, 3。

12. Which of the following are legal identifiers?

A. _3variable

B. 3_variable

C. this

D. super

E. *variable

Answer: A

B. 變數字首不可為數字、C. this 為關鍵字、D. super 為關鍵字、E. 變數字首不可為「*」。

13. Which are not primitive types in Java?

A. float

B. Boolean

C. short

D. Double

E. long

Answer: B, D

Boolean 和 Double 分別為「boolean」與「double」之 Wrapper Class，並非 primitive 資料型態。

14. Given the following Java code:

```
interface Count {
    short counter = 0;
    void countUp( );
}
public class TestCount implements Count {

    public static void main(String[] args) {
        TestCount t = new TestCount( );
        t.countUp( );
    }
    public void countUp( ) {
        for (int x = 6; x > counter; x--, ++counter) {
        System.out.println(" " + counter);
        }
    }
}
```

What is the result?

A.0 1 2

B. 1 2 3

C. 0 1 2 3

D. 1 2 3 4

E. Compiler error

Answer: E

由於 counter 變數是介面變數，其預設為 final static，因此程式碼無法編譯：當第 12 行的程式碼試圖增加 counter 時，編譯器便會發生錯誤。

15. Given the following Java code:

```
1. public class ConstOver {
2.         public ConstOver(int x, int y, int z) {
3.         }
4. }
```

Which two overload the ConstOver constructor?

A. ConstOver( ){}

B. Protected int ConstOver( ){}

C. Private ConstOver(int z, int y, int x){}

D. public Object ConstOver(int x, byte y, byte z){}

E. public void ConstOver(byte x, byte y, byte z){}

Answer: A、C

B:有回傳值 int，是 method 不是建構子。

D:有回傳值 Object，是 method 不是建構子。

E:回傳值是 void，void 是沒有回傳值的 method。

16. Given the following Java code:

```
1. interface foo {
2.         int k = 0;
3. }
4. public class ExamA015 implements foo{
5.         public static void main(String[] args) {
6.                 int i;
7.                 ExamA015 test = new ExamA015( );
8.                 i = test.k;
9.                 i = ExamA015.k;
10.                i = foo.k;
11.        }
12. }
```

What is the result?

A. Compilation succeeds.

B. An error at line 2 causes compilation to fail.

C. An error at line 9 causes compilation to fail.

D. An error at line 10 causes compilation to fail.

E. An error at line 11 causes compilation to fail.

Answer: A

在 interface 所定義的變數其實就是 final static 變數，可直接取用。

17. Given the following Java code:

```
public class foo {
        public static void main (String[] args) {
                String s;
                System.out.println(“s=” + s);
        }
}
```

What is the result?

A. The code compiles and “s=” is printed.

B. The code compiles and “s=null” is printed.

C. The code does not compile because string s is not initialized.

D. The code does not compile because string s cannot be referenced.

E. There is a runtime error.

Answer: C

String 變數在使用前必須先給定初始值。Java 語言中只有 static 變數、陣列與 Primitive Type 會自動加入初始內容值，題中 s 屬於非陣列的參考資料型別 (Reference Type)。

18. Which two statements are true about has-a and is-a relationships? (choose two)
A. Inheritance represents an is-a relationship.
B. Inheritance represents a has-a relationship.
C. Interfaces must be used when creating a has-a relationship.
D. Instance variables can be used when creating a has-a relationship.

Answer: A、D
A: 繼承關係表現出 is-a 的關係:機車(繼承)車，所以機車 is-a 車子，正確
B: 繼承關係表現出 has-a 的關係:機車(繼承)車，機車 has-a 車子是錯的
C: 建立一個 has-a 的關係必須用到介面:不一定，因為機車 has-a 引擎，並不需要實做任何介面，因為引擎是另一種物件。
D: 建立一個 has-a 的關係可以用到實體變數:正確，因為機車 has-a 引擎的實體變數。

19. Which two statements are true? (choose two)
A. A final method in class X can be abstract if and only if X is abstract.
B. A protected method in class X can be overridden by any subclass of X.
C. A private static method can be called only within other static methods in class X.
D. A non-static public final method in class X can be overridden in any subclass of X.
E. A public static method in class X can be called by a subclass of X without explicitly referencing the class X.
Answer: B, E
A: final 方法不可能也同時是抽象方法，所以錯誤。
B: 一個 protected 方法是可以被子類別進行 override，正確。
C: 一個 private 的 static 方法只可以被同類別中的 static 方法所呼叫使用？這是錯誤的。因為其他同類別的 non-static 方法也可以呼叫使用。
D: 一個 final non-static 的 public 方法，可以被該類別的子類別所 overridden？這是錯誤的，因為該方法已經是 final，不可以被任何方式進行 overridden。
E: 一個 public static 方法可以被該類別之子類別直接參考使用，正確。