第10回　コンストラクタ

（１）第6章10.

次のクラスに定義するコンストラクタとして、正しいものを選びなさい。(1つ選択)

1. public class Account {
2. private int balance;
3. }

　A. public void Account (int balance) {

this.balance = balance;

}

　B. **public Account(){**

this.balance = 0;

}

　C. public static Account(){

this.balance = 0;

}

　D. public Constructor(int balance) {

this.balance = balance;

}

［　B　］

（２）第8章03.

次のコードをコンパイル、実行したときの結果として、正しいものを選びなさい。(1つ選択)

1. public class Train {

2. static String name = "none";

3.

4. public **Train**(String name) {　　　　←　コンストラクタ

5. this.name = name;

6. }

7. public static void main(String[] args) {

8. Train t1 = new Train(); ←　引数なし・・・存在しない

9. Train t2 = new Train("aline"); ←　引数あり

10. System.out.print(t1.name + " " + t2.name);

11. }

12. }

A. 「none aline」と表示される

B. 「null aline」と表示される

C. 「aline aline」と表示される

**D. コンバイルエラーになる**

E. 実行時に例外が発生する

［　　D　　］

（３）第8章08.

次のコードのコンパイルを成功させ、実行結果が「refresh L」となるようにしたい。空欄にあてはまるコードを選びなさい。(1つ選択)

1. public class Photo {

2. private String size;

3. private String title;

4.

5. Photo() {

6. \_\_\_\_\_\_\_\_\_\_　←ｵｰﾊﾞｰﾛｰﾄﾞしたｺﾝｽﾄﾗｸﾀを最初に呼ぶ this()

7. }

8. Photo(String title) {

9. this.title = title;

10. }

11. void printInfo() {

12. System.out.println(title + " " + size);

13. }

14.

15. public static void main(String[] args) {

16. Photo p = new Photo();

17. p.printInfo();

18. }

19. }

A. this.size = "L";

this("refresh");

B. Photo("refresh");

this.size = "L";

**C. this("refresh");**

this.size = "L";

D. this.size = "L";

Photo("refresh");

E. this.Photo("refresh");

this.size = "L";

［　　C　　］

（４）第8章09.

次のプログラムをコンパイル、実行したときの結果として、正しいものを選びなさい。(1つ選択)

1. public class SuperClass {

2. protected int num;

3. public SuperClass() {

4. this.num = 1;

5. }

6. public SuperClass(int num) {

7. this.num = num;

8. }

9. }

1. public class SubClass extends SuperClass {

2. private int a;

3. private int b;

4. public SubClass(int a) {

super();　が自動的に追加される

5. this.a = a;

6. }

7. public SubClass(int a, int b) {

8. this(a);

9. this.b = b;

10. }

11. public static void main(String[] args) {

12. SubClass sub = new SubClass(2, 3);

13. System.out.println(sub.num + ":" + sub.a + ":" + sub.b);

14. }

15. }

A. 「1:2:3」と表示される

B. 「0:2:3」と表示される

C. コンパイル工ラーになる

D. 実行時に例外がスローされる

［　　A　　］

（５）第8章59.

コンストラクタに関する説明として、正しいものを選びなさい。(2つ選択)

A. サブクラスはスーパークラスのコンストラクタを引き継がない

B. コンストラクタの戻り値型はvoidにすることができる

C. コンストラクタはprivateで修飾できる

D. コンストラクタは、クラス内のすべてのフィールドを初期化しなければならない

［　　A　　］ ［　　C　　］

第11回　オーバーロードとオーバーライド

（１）第8章02.

次のsampleクラスを継承したサブクラスを定義するときに、サブクラスに定義したメソッドのうち、Sampleクラスのメソッドを正しくオーバーライドしているものを選びなさい。(2つ選択)

1. public class Sample {

2. void methodA(){}

3. void methodB(int a) {}

4. int methodC(int a, int b) {

5. return 0;

6. }

7. int methodD(int a) {

8. return 0;

9. }

10. }

A. public void methodA() {}

B. public void methodB(long a) {} **引数の型が異なるのでオーバーライドではない**

C. public int methodC(char a, int b) { **引数の数と型が異なるのでオーバーライドではない**

return 0;

}

D. public int methodD(int i) {

return 1;

}

［　　A　　］ ［　　D　　］

（２）第8章35.

以下の中から、メソッドをオーバーロードしているクラスを選びなさい。(1つ選択)

A. class Triangle {

public void setup(int height, int base) { }

public void setup(int height) { } **引数の個数が異なるのでOK**

}

B. class Triangle {

public setup(int height, int base) { } **戻り値の型宣言がないのでエラー**

public void setup(int height) { }

}

C. class Triangle {

public int setup(int height, int base) { } **戻り値の型のみが異なるのは**

public void setup(int height, int base) { } **オーバーロードとは言えない**

｝

D. class Triangle {

public void setup(int height, int base) { }

public void setdown() { } **メソッド名が異なるので別のもの**

｝

［　　A　　］

（３）第8章40.

次のプログラムをコンパイル、実行したときの結果として、正しいものを選びなさい。(1つ選択)

1. public class Employee {

2. public void disp() {

3. System.out.println("Employee");

4. }

5. }

1. public class Manager extends Employee {

2. public void disp() {

3. System.out.println("Manager");

4. }

5. public static void main(String[] args) {

6. Manager m = new Manager();

7. Employee e = m;

8. e.disp(); **Managerクラスのインスタンスで参照される**

9. }

10. }

A. 「Employee」が表示される

B. 「Manager」が表示される

C. 「Employee」「Manager」の順に表示される

D. 「Manager」「 Employee」の順に表示される

E. コンパイル工ラーが発生する

F. 実行時に例外がスローされる

［　　B　　］

（４）第8章41.

メソッドのシグニチャを構成する要素として正しいものを選びなさい。(4つ選択)

A. 引数の数

B. 引数の名前

C. アクセス修飾子

D. 引数の型

E. メソッド名

F. 戻り値の型

G. 引数の順番

［　　A　］ ［　　D　　］ ［　　E　　］［　　G　　］

（５）第8章48.

次のコードをコンパイル、実行したときの結果として、正しいものを選びなさい。(1つ選択)

1. public class Calculator {

2. int beforeTaxes(int price) {

3. return(int)(price \* 1.08);

4. }

5. double beforeTaxes(int price) {

6. return price \* 1.08;

7. }

8.

9. public static void main(String[] args) {

10. int p = 40;

11. Calculator calc = new Calculator();

12. System.out.println("payment = " + calc.beforeTaxes(p));

13. }

14. }

A. 「payment = 40」と表示される

B. 「payment = 43」と表示される

C. 「payment = 43.2」と表示される

D. コンパイル工ラーになる

**両方とも同じ型の引数を宣言しているので同一メソッドととして認識されるから**

［　　D　　］

第12回　抽象クラスとインタフェース

（１）第8章05.  
次のプログラムをコンパイル、実行したときの結果として、正しいものを選びなさい。(1つ選択)

1. public interface Sample {

2. void test();

3. }

1. public class A implements Sample {

2. public void test() {

3. System.out.println("A");

4. }

5. }

1. public class B extends A {

2. public void test() {

3. System.out.println("B");

4. }

5. }

1. public class Main {

2. public static void main(String[] args) {

3. Sample[] samples = {new A(), new B() };

4. for(Sample s : samples) {

5. s.test();

6. }

7. }

8. }

A. 「A」「B」の順に表示される

B. 「B」「A」の順に表示される

C. 「A」「A」の順に表示される

D. 「B」「B」の順に表示される

E. Bクラスでコンパイル工ラーが発生する

F. Mainクラスでコンパイル工ラーが発生する

［　A　　］

（２）第8章24.

以下の中から、ボリモーフィズムに関係が深いものを選びなさい。(2つ選択)

A. インタフェースの継承

B. インタフェースの実装

C. メソッドのオーバーロード

D. メソッドのオーバーライド

E. アクセス修飾子protectedの利用

［　　B　　］ ［　　D　　］

（３）第8章50.

抽象クラスに関する説明として、正しいものを選びなさい。(1つ選択)

A. 抽象クラスに定義されるメソッドは、暗黙的にpublic abstract である

B. 抽象クラスに定義されるフィールドは、暗黙的にstatic finalである

C. 抽象クラスはサブクラス化できない

D. 抽象クラスはインスタンス化できない

E. 抽象クラスは抽象メソッドを持たなければいけない

［　　D　　］

（４）第8章56.

インタフェースの定義として、正しいものを選びなさい。(1つ選択)

A. public interface App {

public String id = "A001";

void execute(String param);　　　→　public abstract とみなされるのでOK

}

B. public interface App {

private String id = "A001"; → publicでないとNG

public void execute(String param);

}

C. public interface App {

public String id; → 初期値が必要

abstract void execute(String param);

}

D. public interface App {

abstract String id = "A001"; → abstractは使えない

void execute(String param);

}

［　A　　］

（５）第8章57.

次のコードが正常にコンバイルされるようにしたい。samplelmplクラスの2行目「//insert code here」に入るコードとして、正しいものを選びなさい。(1つ選択)

1. abstract class Sample {

2. void print() {

3. System.out.println("sample");

4. }

5. public abstract void test();

6. }

1. public class SampleImpl extends Sample {

2. //insert Code here

3. }

A. void test() {

System.out.println("impl");

}

B. public abstract void test() {

System.out.println("impl");

}

C. public void test() {

System.out.println("impl");

}

D. public void print() {

System.out.println("impl");

}

E. void print() {

System.out.println("impl");

}

［　　C　　］

第13回　static

（１）第8章21.   
次のコードをコンバイル、実行したときの結果として、正しいものを選びなさい。(1つ選択)

1. public class Test {

2. private static int a;

3. private int b;

4.

5. public static int countUpA() {

6. return ++a;

7. }

8. public int doMethod() {

9. return countUpA();

10. }

11.

12. public static void main(String[] args) {

13. Test test = new Test();

14. System.out.print(" " + test.doMethod());

15. System.out.print(" " + test.countUpA());

16. }

17. }

A. 「01」と表示される

B. 「12」と表示される

C. 9行目でコンパイル工ラーになる

D. 15行目でコンパイル工ラーになる

E. 実行時に例外が発生する

［　　B　　］

（２）第8章37.

次のコードをコンパイル、実行したときの結果として、正しいものを選びなさい。(1つ選択)

1. public class News {

2. static int id;

3. String name;

4.

5. static void printInfo() {

6. System.out.println(id + ": " + name);

7. }

8.

9. public static void main(String[] args) {

10. News n = new News();

11. n.printInfo();

12. }

13. }

A、何も表示されない

B. 「0:」と表示される

C. 「0:null」と表示される

D. コンパイル工ラーになる

E. 実行時に例外が発生する

［　　D　　］

（３）第8章42.

次のコードをコンパイル、実行したときの結果として、正しいものを選びなさい。(1つ選択)

1. public class Counter {

2. static int count = 0;

3.

4. Counter() {

5. ++ count;

6. }

7.

8. public static void main(String[] args) {

9. Counter c = new Counter();

10. c = new Counter();

11. System.out.println(c.count);

12. }

13. }

A. 0が表示される

B. 1が表示される

C. 2が表示される

D. コンパイル工ラーになる

E. 実行時に例外が発生する

［　　C　　］

（４）第8章55.

次のコードのコンバイルを成功させ、実行結果が「2」となるようにしたい。空欄にあてはまるコードを選びなさい。(1つ選択)

1. public class Main {

2. public static void main(String[] args) {

3. int a = subtract(5, 3);

4. System.out.println(a);

5. }

6. \_\_\_\_\_\_ int subtract(int a, int b) {

7. return a - b;

8. }

9. }

A. static

B. final

C. public

D. void

［　　A　　］

（５）第8章58.

次のコードをコンパイル、実行したときの結果として、正しいものを選びなさい。(1つ選択)

1. public class Point {

2. public static void main(String[] args) {

3. int x = 5;

4. Point p = new Point();

5. p. move(x);

6. }

7. public void move(int y) {

8. System.out.println(x++ + " " + --y);

9. }

10. }

A. 「05」と表示される

B. 「15」と表示される

C. 「04」と表示される

D. コンパイル工ラーになる

E. 実行時に例外が発生する

［　　D　　］