● 50 問× 2 点 = 100 点 ●持ち込み一切不可

(注) 以下の問題で，同じアルファベットの空欄は同じ内容である (例えば[m]と[m]の内容は必ず同じ)。しかし，アルファベットの異なる空欄は，同じ内容かもしれないし，異なる内容かもしれない (例えば[m]と[n]は同じ内容かもしれないし，異なる内容かもしれない)。

1. 計 5 問

次の文において，空欄 A〜E に入れる語として適切なものを，回答欄 A〜E に書け。

Java をはじめとする多くのオブジェクト指向言語では，事物の[A]を表現するために[B]を定義する。たとえば，“ 鳥 ” という[A]を表すために，Bird という名前の[B]を定義することができる。

[B]によって事物の[A]を表すとき，その事物の持つ「振る舞い」と「情報」を表現することが重要になる。Java の場合，「振る舞い」は[C]によって表現し，「情報」は[D]と呼ばれる変数によって表現する。[C]や[D]のように，[B]の定義の中で定義されたものを “ [B]の一員 ” という意味で[E]と呼ぶ。

2. 計 6 問

次の文において，空欄 F〜K に入れる語として適切なものを，回答欄 F〜K に書け。

クラスは，ユーザ (プログラマ) が定義できるオリジナルの「データ型」である。一般に，「型」情報から生成された実体を，[F]と呼ぶ。たとえば，「double 型の変数 d は，double 型の[F]である」などと言う。特に，[B]型の[F]のことを，[G]と呼ぶ。

それぞれの[G]は，その[D]の内容によって，異なる状態を持つことができる。言い換えれば，[B]が事物の概念を表しているのに対し，[G]は個々の事物を表しているのである。たとえば，[B]Cat は「猫」という概念を表し，Cat 型の[G]Tama や[G]Mike は，それぞれ「タマ」や「ミケ」といった個々の猫を表していることになる。

一般に，プログラムが正しく動作するためには，[G]を生成するとき，[G]の[D]を初期化したり，特別な前処理を行う必要がある。そのために，Java では[D]を初期化するための特別な[C]を定義することができる。この特別な[C]のことを，[H]と呼ぶ。

端的に言って，オブジェクト指向とは，事物の[A]を[B]で表現して，その[F]としての[G]を生成し，それぞれの[G]がお互いにメッセージの交信をすることによってプログラムが動作する，という考え方である。

また，オブジェクト指向の重要な考えに[B]の[I]がある。たとえば，「乗り物」を表す Vehicle という[B]を[J]とし，その[K]として「車」を表す Car という[B]を定義することができる。このとき，Car は Vehicle の[E]を[I]することになる。

3. 計 4 問

次の実行可能な Java プログラムを完成させるために，空欄 L〜O 部分に必要な内容を回答欄 L〜O に書け。

```
[ L ] TestClass {
  public [ M ] main( [ N ] ) {
    [ O ]( "hello!" ); // hello! と画面に出力する
  }
}
```

4. 計 4 問

問題 3 のプログラムのソースファイル名はどのようなものにすべきか。ソースファイル名を回答欄 P に書け。また，問題 3 のソースファイルをコンパイルするには，どのようにコマンドを入力すればよいか。ソースファイル名を含む完全なコマンドを回答欄 Q に書け。

問題 3 のソースファイルをコンパイルすると，クラスファイルが生成される。このクラスファイル名を完全な形で回答欄 R に書け。問題 3 から生成されたクラスファイルを実行するには，どのようなコマンドを入力すればよいか。回答欄 S に，クラス名を含む完全なコマンドを書け。

5. 計 4 問

次のプログラムを完成させるために必要な各空欄 T〜W の内容を，それぞれ回答欄 T〜W に書け。なお，プログラムの仕様については，プログラム内のコメントに書かれているので注意すること。

```
1  [ L ] Elevator { // エレベーターを表すクラス Elevator
2    // 以下の floor，n は，クラスの外部からアクセスできない
3    [ T ] int floor; //現在の階を表す floor
4    [ T ] int n;     //現在の総乗員数を表す n
5
6    //以下のコンストラクタとメソッドはどこからでもアクセス可能
7    //floor と n を初期化するコンストラクタ Elevator
8    [ U ] Elevator( int floor, int n ) {
9      [ V ].floor = floor; [ V ].n = n;
10   }
11   [ U ] int getFloor() { return floor; }
12   [ U ] void moveTo( int floor ) { [ V ].floor = floor; }
13   [ U ] int getN() { return n; }
14   [ U ] void setN( int n ) { [ V ].n = n; }
15   [ U ] void pickUp( int num ) {
16     n = n + num;
17     [ O ]( "現在の乗員数は"+n+"人です");
18   }
19   [ U ] [ M ] main ( [ N ] ) {
20     //1 階で乗客の乗っていないエレベーターオブジェクト e を生成し，
21     //3 階に移動して，客を 2 人載せる
22     Elevator e = [ W ]
23     e.moveTo( 3 ); e.pickUp( 2 );
24   }
25 }
```

6. 計 12 問

次の文章の空欄 X〜Z，a〜i に当てはまる適切な内容を回答欄 X〜Z，a〜i に書け。

一般的にソフトウェアを作成する場合，関連するデータと機能をひとつにまとめて，[X]と呼ばれる再利用可能な「部品」にまとめる。クラスは，最小の[X]であると言える。Java では関連するクラスを集めて，[Y]と呼ばれるより大きい[X]を作成することができる。このようにプログラムの構成要素を部品化 ([X]化) することで，プログラムの見通しが良くなるだけでなく，プログラムの設計もしやすくなる。また，[X]ごとに独立して開発することも可能になる。

[X]を作成する際に重要になるのは，[X]内の構成要素へのアクセスを適切に制御することである。[X]内の構成要素に対する外部からのアクセスを無制限に許せば，[X]の機能が正常に動作することは期待はできなくなる。なぜならば，外部から[X]内の構成要素に対して不正な操作が行われるかもしれないからである。

そのために，Java ではクラスのメンバへのアクセスを制御できるようになっている。たとえば，問題 5 のプログラムでも各メンバに対して，アクセスの指定を行っている。なお，一般に[D]は，クラス外部からアクセス[Z]ようにすべきである，とされている。そのため，問題 5 の 11〜14 行目のように，必要ならば[D]へのアクセスを行うための専用の[C]を用意する。このような[C]を[a]と呼ぶ。特に，[D]の値を返す[a]を[b]と呼び，[D]の値を設定する[a]を[c]と呼ぶ。問題 5 のプログラムでは，[d]行目と[e]行目の[a]が[b]であり，[f]行目と[g]行目の[a]が[c]である。

[X]の外部からアクセスできない構成要素は，その[X]の[h]部分となる。一方，その[X]の外部からアクセスできる構成要素は，その[X]の[i]部分になる。[h]部分は変更を自由に行うことができる。なぜなら，外部から利用されることがないので，変更してもその[X]の外部に影響が出ることはないからである。それに対し[i]部分は，外部から利用される部分であり，あまり変更はできない。変更すると，外部からその[i]部分を利用している箇所を変更に合わせて書き換えなくてはならなくなり，これは大変な作業量になってしまうためである。

7. 計 2 問

次のソースプログラムは，tuisjava.com というドメイン名を持つ会社が作成したものであり，クラス Compressor を定義している。空欄[j]にあてはまる適切な内容を回答欄[j]に書け。通常，このクラス Compressor をパッケージ com.tuisjava.Tools の一部として使用するソースプログラムの冒頭部分にはインポート宣言を書かなければならない。このインポート宣言を完全な形で回答欄[k]に書け。

```
[ j ] com.tuisjava.Tools;
[ L ] Compressor { // データ圧縮機能を持つクラス Compressor
  // データを圧縮する compress()
  byte[] compress( byte[] rawData ) { /* 略 */ }
  // 圧縮されたデータを元に戻す decompress()
  byte[] decompress( byte[] compressedData ) { /* 略 */ }
}
```

● 2 枚目の問題用紙に続く

8. 計 4 問

次のプログラムを完成させるために必要な各空欄 l〜o の内容を，それぞれ回答欄 l〜o に書け。なお，プログラムの仕様については，プログラム内のコメントに書かれているので注意すること。

```
1   [ l ] class Animal { // 動物を表す抽象クラス Animal
2     [ T ] double x, y;   // 居場所を表す x, y
3
4     // コンストラクタ Animal()
5     [ U ] Animal( double xpos, double ypos ) {
6       x = xpos; y = ypos;
7     }
8     // 鳴き声をあげる行為を表す抽象メソッド makeVoice()
9     [ U ] [ l ] void makeVoice();
10  }
11  class Cat [ m ] Animal { // 猫を表す Animal のサブクラス Cat
12    [ U ] Cat( double x, double y ) { [ n ]( x, y ); }
13    [ U ] void makeVoice() {
14      [ O ]( "にゃあ" ); // 「にゃあ」となく（画面に表示する）
15    }
16  }
17
18  class Dog [ m ] Animal { // 犬を表す Animal のサブクラス Cat
19    [ U ] Cat( double x, double y ) { [ n ]( x, y ); }
20    [ U ] void makeVoice() {
21      [ O ]( "わん" ); // 「わん」と吠える（画面に表示する）
22    }
23  }
24
25  class Zoo { // 動物園を表すクラス Zoo
26    // 動物に鳴いてもらうメソッド letItMakeVoice()
27    void letItMakeVoice( [ o ] a ) { // 左の空欄の o は小文字の o
28      // オブジェクト a の makeVoice() を呼んで鳴いてもらう
29      a.makeVoice();
30    }
31    public [ M ] main( [ N ] ) {
32      Zoo aZoo = new Zoo(); // 動物園オブジェクトを生成
33      // 位置 (0.0, 0.0) に猫オブジェクトを生成
34      Cat aCat = new Cat( 0.0, 0.0 );
35      // 位置 (5.0, 5.0) に犬オブジェクトを生成
36      Dog aDog = new Dog( 5.0, 5.0 );
37      aZoo.letItMakeVoice( aCat ); // 猫に鳴いてもらう
38      aZoo.letItMakeVoice( aDog ); // 犬に鳴いてもらう
39    }
40  }
```

9. 計 2 問

次のプログラムを完成させるために必要な各空欄 p〜q の内容を，それぞれ回答欄 p〜q に書け。なお，プログラムの仕様については，プログラム内のコメントに書かれているので注意すること。

```
1   // “ 表示することが出来る ” という意味を持つインタフェイス Showable
2   [ p ] Showable { void show(); }
3
4   [ l ] class Document { //書類を表す抽象クラス Document
5     [ T ] String title;   //書類のタイトルを表す title
6     [ U ] Document( String theTitle ) { title = theTitle; }
7     // 書類をファイルに保存する抽象メソッド save()
8     [ U ] [ l ] void save();
9     // その他の抽象メソッドはここでは省略
10  }
11
12  // テキスト書類を表すクラス TextDocument
13  [ U ] class TextDocument [ m ] Document [ q ] Showable {
14    [ U ] TextDocument( String theTitle ) {
15      [ n ]( theTitle );
16    }
17    [ U ] void save() { /* このテキスト書類を保存する */ }
18    [ U ] void show() { /* このテキスト書類を表示する */ }
19    // その他のメソッドはここでは省略
20  }
```

10. 計 5 問

次の文章の空欄 r〜v に当てはまる内容を，回答欄 r〜v に書け。

オブジェクト指向で最も重要な考え方は，[r] 関係を表現する「継承」である。「B [r] A」とは，「B は A の一種」という関係を表す。たとえば，「楕円 [r] 2 次元図形」は「楕円は 2 次元図形の一種」という意味であり，同様に「矩形 [r] 2 次元図形」は「矩形は 2 次元図形の一種」という意味である。Java では「サブクラス [r] [s]」という [r] 関係が成立していると考えて，サブクラスのオブジェクトは，[s] のオブジェクトとして扱うことが可能になっているのである。問題 8 のプログラムでは，「猫 [r] 動物」という関係を表すために，動物を表す Animal という [s] を作り，そのサブクラスとして猫を表す Cat を定義している。

また，サブクラスでは，[s] から継承したメンバだけでなく，サブクラス独自のメンバを追加することもできる。また，サブクラスでメソッドを [t] することで，[s] とは異なった独自の振る舞いをサブクラスにさせることができる。

重要なのは，ソフトウェアの多くの部分を，[s] で記述することである。[s] で記述された部分は，サブクラスのオブジェクトを一括して扱うことができ，サブクラスが増えたとしても変更する必要がない。そのおかげで，汎用性・拡張性に富んだソフトウェアを作成することができるようになる。

問題 8 のプログラム例では，27〜30 行目のクラス Zoo の letItMakeVoice() メソッドは，[s] Animal だけで記述されていて，Cat 型オブジェクトや Dog 型オブジェクトを扱うことができる。また，このプログラムで新たに「牛」を扱う必要がでてきた場合は，Animal のサブクラスとして「牛」を表す Cattle を新しく定義すればよい。こうして，Animal のサブクラスが新しく定義されても，[s] Animal を使って記述された letItMakeVoice() メソッドは変更する必要がない。

なお，多くの場合，[s] は，メソッドの詳細な定義が不可能なことが多い。これは，たいていの [s] が，抽象度の高い一般概念を表しているためである。問題 8 の例では，「動物」は一般概念であり，その具体的な振る舞い (メソッド) を定義することは難しい。たとえば，「動物」がどのように「鳴き声を上げる」かは，具体的に定義できない。一方，「猫」や「犬」という概念は，「にゃあとなく」，「わんと吠える」というように，具体的な振る舞いを定義できる。

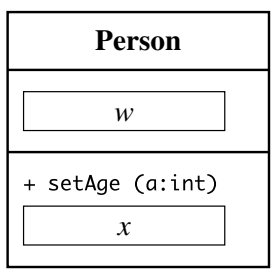
前述のように，一般概念を表すクラスは，振る舞い (メソッド) を適切に定義できない。その場しのぎで何かしらの定義を行っても，そのクラスのオブジェクトが生成されて，メソッドが呼ばれた場合に，不都合が起きる可能性がある。そこで，[u] クラスという一般概念を表すクラスが用意されている。[u] クラスのオブジェクト生成は，禁止される。また，[u] クラスでは，[u] メソッドという定義部分のない特別なメソッドを宣言できるようになっている。[u] メソッドは，サブクラスで [t] することによって，定義を与えることができる。

ところで，[r] とともに重要なクラス間関係が，[v] 関係である。「B [v] A」は，「B は A を持っている」という意味である。たとえば，「車 [v] エンジン」とは，「車はエンジンを持っている」という意味である。「クラス B [v] クラス A」という [v] 関係をプログラム上で表すには，A 型オブジェクトをクラス B のメンバとして持たせる。これを，「包含」という。たとえば，問題 9 の Document クラスは，タイトルを表すメンバ title を持っている。これは，「書類はタイトルを持っている」という [v] 関係を表しているのである。[r] 関係と [v] 関係の違いをはっきりと認識して，[r] 関係を表すには「継承」を使い，[v] 関係を表すには「包含」を用いるのが重要である。

11. 計 2 問

右下の UML クラス図が，左下のクラス定義を表すように，空欄 w〜x に当てはまる内容を回答欄 w〜x に書け。各型情報や，各メンバのアクセス特性も含めること。ただし，メソッドの動作内容などは書かなくて良い。

```
class Person {
  private int age;
  public void setAge( int a ) {
    age = a;
  }
  public int getAge() {
    return age;
  }
}
```



A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

X

Y

Z

a

b

c

d

e

f

g

h

i

j

k

l

m

n

o

p

q

r

s

t

u

v

w

x