Javaバイブルシリーズ

オブジェクト指向プログラミングの教科書（別冊）

SCC

●本書は、以下の開発環境で動作確認を行っております。

・Windows10
・Java10.0.2
・MySQL8.0.12

別冊

確認問題解答例

プログラムについてはこれが唯一の解答というわけではありません。同様の結果になる異なったプログラムがありますので、是非オリジナルを考えてみてください。

確認問題 01

Q1 a） 現実世界

b） モデリング

c） OOA

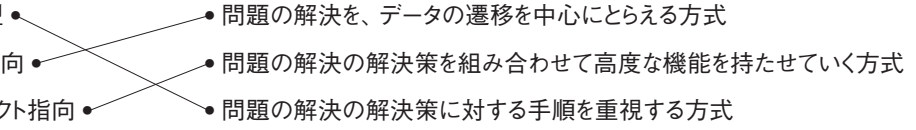
d） 設計工程

e） OOT

f） OOP

Q2 オブジェクト同士の相互作用として、システムの振る舞いをとらえる考え方

Q3



Q4 a. 手　b. オ　c. 手　d. デ

Q5 a. インスタンス

b. スーパークラス

c. クラス

d. 差分プログラミング

e. オブジェクト

Q6 イ

Q7 ア

［解説］イは継承の効果、ウは差分プログラミングについて、エはポリモルフィズムと呼ばれる機能の効果について記述されている。

Q8 • プログラム品質の向上

• プログラム生産性の向上

Q9 完成品である部品を組み合わせることで生産性が向上し、十分なテストを経た高品質の部品を使用することで、プログラムの品質が向上するから。

確認問題 02

Q1 a） クラス

b） フィールド

c） メソッド

Q2
- mainメソッドを持つクラス名とファイル名が異なっている
- mainはmain(String[] args)として引数を受け取らなければならない
- z = x + yに;（セミコロン）が必要
- javaにはprintf()というメソッドはない
- }が不足している

Q3

```
public class Confirm02Q3{
    public static void main(String[] args){
        System.out.println("Javaの勉強がんばります！！");
    }
}
```

確認問題 03

Q1 a） 基本データ

b） 参照（オブジェクト）

c） byte

d） short

e） long

f） double

g） true（真）

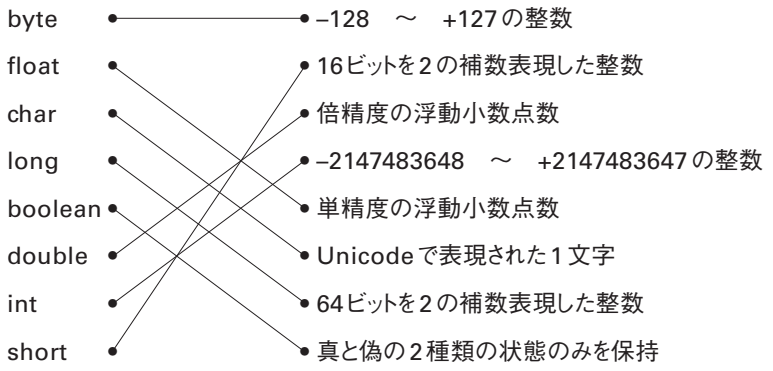
h） false（偽）

Q2 a） class

b） main

c） x－y

Q3



Q4 8行目

［解説］int型をshort型に代入しようとしているため精度のエラーが発生する

Q5

```
public class Confirm03Q5{
    public static void main(String[] args){
        double x;
        float  y;

        x = 12.5;
        y = (float)x;

        System.out.println( x );
        System.out.println( y );
    }
}
```

確認問題 04

Q1

```
public class Confirm04Q1{
    public static void main(String[] args){
        char a;
        String str;
        str = new String( );

        a = 'ね';
        str = "言ったとおりでしょ！！";

        System.out.println( a );
        System.out.println( str );
    }
}
```

Q2

```
public class Confirm04Q2{
    public static void main(String[] args){
        String str = "こんにちは";
        int x;
        x = str.length();
        System.out.println(str + "の文字数は" + x + "文字です");
    }
}
```

確認問題 05

Q1

```
public class Confirm05Q1{
    public static void main(String[] args){
        int[] array;
        array = new int[3];

        array[0] = 5;
        array[1] = 8;
        array[2] = array[0] + array[1];
        System.out.println(array[2]);
    }
}
```

Q2

```
class Confirm05Q2{
    public static void main(String[] args){
        int[] array;
        int i;
        array = new int[4];

        array[0] = 100;
        array[1] = 200;
        array[2] = array[0] + array[1];
        array[3] = array[2] * 2;

        for(i=0; i<4; i++){
        System.out.println(array[i] );
        }
    }
}
```

Q3

```
public class Confirm05Q3{
    public static void main(String[] args){
        int[] array;
        int element;

        array = new int[10];

        element = array.length;
        System.out.println(element);
    }
}
```

Q4 “null”は、0や空白とは異なり「何も入っていない状態」のことである。

確認問題 06

Q1

```
public class Confirm06Q1{
    public static void main(String[] args){
        int x=5, y=10, z;
        z = x;
        x = y;
        y = z;

        System.out.println( "入れ替え後のx:"+x );
        System.out.println( "入れ替え後のy:"+y );
    }
}
```

Q2 10

9

8

10

9

9

8

Q3 c = a + b / 2;　を　c = (a + b) / 2;に修正

Q4 ○

○

×（x,yどちらかが正でどちらかが負だと0未満になりfalse）

○

Q5

```
public class Confirm06Q5{
    public static void main(String[] args){
        int x = -64;
        System.out.println( "xの4倍："+(x<<2) );
        System.out.println( "xの1/8："+(x>>3) );
    }
}
```

Q6

```
if(y != 10){
    x = 3;
}
else{
    x = 10;
}
```

確認問題 07

Q1

```
public class Confirm07Q1{
    public static void main(String[] args){
        int x;
        x = -30;

        if(x>=0){
            System.out.println( "変数xは正の数です" );
        }else{
            System.out.println( "変数xは負の数です" );
        }
    }
}
```

Q2

```
public class Confirm07Q2{
    public static void main(String[] args){
        int score;
        score = 80;

        if( score >= 80 ){
            System.out.println( "優です" );
        }else if( score >= 65 ){
            System.out.println( "良です" );
        }else if( score >= 50 ){
            System.out.println( "可です" );
        }else{
            System.out.println( "不可です" );
        }
    }
}
```

Q3

```
public class Confirm07Q3{
    public static void main(String[] args){
        int month;
        month = 10;

        switch( month ){
            case 3:
            case 4:
            case 5:
                System.out.println( "春です" );
                break;
            case 6:
            case 7:
            case 8:
                System.out.println( "夏です" );
                break;
            case 9:
            case 10:
            case 11:
                System.out.println( "秋です" );
                break;
            case 12:
            case 1:
            case 2:
                System.out.println( "冬です" );
                break;
            default:
                System.out.println( "範囲外です" );
                break;
        }
    }
}
```

Q4 a： 初期化処理

b： 繰り返しを継続する条件

c： 繰り返し1回ごとの処理

d： 繰り返しを継続する条件

e： i = 0

f： i < 10

g： i++

Q5

```
public class Confirm07Q5{
    public static void main(String[] args){
        int i;
        int total = 0;

        for(i = 2 ; i <= 100 ; i += 2 ){
            total += i;
        }
        System.out.println("合計は" + total + "です。");
    }
}
```

Q6

```
public class Confirm07Q6{
    public static void main(String[] args){
        int i;
        int total = 0;

        i = 2;
        while( i <= 100){
            total += i;
            i += 2;
        }
        System.out.println("合計は" + total + "です。");
    }
}
```

Q7

```
public class Confirm07Q7{
    public static void main(String[] args){
        int i,j;

        for( i = 1 ; i <= 9 ; i++ ){
            for( j = 1 ; j <= 9 ; j++){
                System.out.print( i * j + "¥t" );
            }
            System.out.println( );
        }
    }
}
```

Q8 a: i = 1 ; i <= 5 ; i++

b: j = 1 ; j <= i ; j++

c: System.out.print("¥n")

Q9 解答例1（while構文を使用した場合）

```
public class Confirm07Q9{
    public static void main(String[] args){
        int[] score = {57,60,89,32,66,77,92,45};
        int max = 0;
        int min = 100;
        int i=0;

        while( i < score.length){
            if( max < score[i] ){
                max = score[i];
            }
            if( min > score[i]){
                min = score[i];
            }
            i++;
        }
        System.out.println("最高点は" + max + "点です。");
        System.out.println("最低点は" + min + "点です。");
    }
}
```

解答例2（for構文を使用した場合）

```
public class Confirm07Q9{
    public static void main(String[] args){
        int[] score = {57,60,89,32,66,77,92,45};
        int max = 0;
        int min = 100;

        for( int i = 0 ; i < score.length ; i++ ){
            if( max < score[i] ){
                max = score[i];
            }
            if( min > score[i] ){
                min = score[i];
            }
        }
        System.out.println("最高点は" + max + "点です。");
        System.out.println("最低点は" + min + "点です。");
    }
}
```

解答例3（拡張for構文を使用した場合）

```
public class Confirm07Q9{
    public static void main(String[] args){
        int[] score = {57,60,89,32,66,77,92,45};
        int max = 0;
        int min = 100;

        for( int n : score ){
            if( max < n ){
                max = n;
            }
            if( min > n ){
                min = n;
            }
        }
        System.out.println("最高点は" + max + "点です。");
        System.out.println("最低点は" + min + "点です。");
    }
}
```

確認問題 08

Q1

```
public class Confirm08Q1{

    public static void main(String[] args){
        Confirm08Q1   instA;

        instA = new Confirm08Q1();

        instA.morningGreet();
    }

    public void morningGreet(){
        System.out.println("おはようございます！");
    }
}
```

Q2

```
public class Confirm08Q2{

    public static void main(String[] args){
        Confirm08Q2   instA,instB,instC;

        instA = new Confirm08Q2();
        instB = new Confirm08Q2();
        instC = new Confirm08Q2();

        instA.morningGreet();
        instB.daytimeGreet();
        instC.nightGreet();
    }

    public void morningGreet(){
        System.out.println("おはようございます！");
    }

    public void daytimeGreet(){
        System.out.println("こんにちは！");
    }
```

続く

```
    public void nightGreet(){
        System.out.println("こんばんは！");
    }
}
```

※以降、public classで始まるプログラムソースは別ファイルにします

確認問題 09

Q1

```
public class Confirm09Q1{

    public static void main(String[] args){
        Dish1   instA;

        instA = new Dish1();

        instA.breakfast();
    }
}
```

```
public class Dish1{

    public void breakfast(){
        System.out.println("目玉焼きとみそ汁です。");
    }
}
```

Q2

```
public class Confirm09Q2{

    public static void main(String[] args){
        Dish2   instA,instB,instC;

        instA = new Dish2();
        instB = new Dish2();
        instC = new Dish2();

        instA.breakfast();
        instB.lunch();
        instC.dinner();
    }
}
```

```
public class Dish2{

    public void breakfast(){
        System.out.println("目玉焼きとみそ汁です。");
    }

    public void lunch(){
        System.out.println("とんかつ定食です。");
    }

    public void dinner(){
        System.out.println("焼肉食べ放題です。");
    }
}
```

確認問題 10

Q1

```
public class Confirm10Q1{

    public static void main(String[] args){
        int           result;
        int           num = 100;
        Confirm10Q1   instA;

        instA = new Confirm10Q1();

        result = instA.duplicate(num);
        System.out.println("渡した数値[" + num +"]の倍数は" + result);
    }

    public int duplicate(int num){
        return (num * 2);
    }
}
```

Q2

```
public class Confirm10Q2{

    public static void main(String[] args){
        int           num = 100;
        int           result;
        Confirm10Q2   instA;

        instA = new Confirm10Q2();

        result = instA.duplicate(num);
        System.out.println("渡した数値[" + num +"]の倍数は" + result);
        result = instA.triplicate(num);
        System.out.println("渡した数値[" + num +"]の3倍の数は" + result);
    }

    public int duplicate(int num){
        return (num * 2);
    }
```

続く

```
    public int triplicate(int num){
        return (num * 3);
    }
}
```

Q3

```
public class Confirm10Q3{

    public static void main(String[] args){
        int           num;
        int           result;
        Confirm10Q3   instA;

        instA = new Confirm10Q3();

        num = Integer.parseInt(args[0]);
        result = instA.duplicate(num);
        System.out.println("渡した数値[" + num +"]の倍数は" + result);
        result = instA.triplicate(num);
        System.out.println("渡した数値[" + num +"]の3倍の数は" + result);
    }

    public int duplicate(int num){
        return (num * 2);
    }

    public int triplicate(int num){
        return (num * 3);
    }
}
```

確認問題 11

Q1

```
public class CalcStatic{

    public static int add(int no1,int no2){
        return (no1 + no2);
    }

    public static int subtract(int no1,int no2){
        return (no1 - no2);
    }
}
```

```
public class CalcNomal{

    public int add(int no1,int no2){
        return (no1 + no2);
    }

    public int subtract(int no1,int no2){
        return (no1 - no2);
    }
}
```

```
public class Confirm11Q1{

    public static void main(String[] args){
        int           result;
        CalcNomal     instA;

        instA = new CalcNomal();

        result = instA.add(100,50);
        System.out.println("Nomalの加算結果 :" + result);

        result = CalcStatic.add(100,50);
        System.out.println("Staticの加算結果:" + result);

        result = instA.subtract(100,50);
```

続く

```
        System.out.println("Nomalの減算結果 :" + result);

        result = CalcStatic.subtract(100,50);
        System.out.println("Staticの減算結果:" + result);
    }
}
```

Q2

```
public class Confirm11Q2{

    public static void main(String[] args){
        int         result,no1,no2;
        CalcNomal   instA;

        instA = new CalcNomal();

        no1 = Integer.parseInt(args[0]);
        no2 = Integer.parseInt(args[1]);

        result = instA.add(no1,no2);
        System.out.println("Nomalの加算結果 :" + result);

        result = CalcStatic.add(no1,no2);
        System.out.println("Staticの加算結果:" + result);

        result = instA.subtract(no1,no2);
        System.out.println("Nomalの減算結果 :" + result);

        result = CalcStatic.subtract(no1,no2);
        System.out.println("Staticの減算結果:" + result);
    }
}
```

確認問題 12

Q1

```
public class VariableCheck{

    public static    int  classVariable=0;
    public           int  instanceVariable=0;

    public void plus(){
        classVariable++;
        instanceVariable++;
    }

    public void minus(){
        classVariable--;
        instanceVariable--;
    }
}
```

```
public class Confirm12Q1{

    public static void main(String[] args){
        VariableCheck  instA,instB;
        int            i;

        instA = new VariableCheck();
        instB = new VariableCheck();

        for(i=0;i<50;i++){
            instA.plus();
            instB.minus();
        }

        System.out.println("プラス側のクラス変数        :"+instA.classVariable);
        System.out.println("プラス側のインスタンス変数  :"+instA.instanceVariable);
        System.out.println("マイナス側のクラス変数      :"+instB.classVariable);
        System.out.println("マイナス側のインスタンス変数:"+instB.instanceVariable);
    }
}
```

Q2

```
public class Confirm12Q2{

    public int distribution[]={0,0,0,0,0,0,0,0,0,0};

    public static void main(String[] args){
        Confirm12Q2 instA,instB;
        int         i;

        instA = new Confirm12Q2();
        instB = new Confirm12Q2();

        instA.randomNum();
        instB.randomNum();

        for(i=0;i<10;i++){
            System.out.print("[" + i + "]が出た回数:");
            System.out.println(instA.distribution[i]
                                +instB.distribution[i]);
        }
    }

    public void randomNum(){
        int ret,i;

        for(i=0;i<10;i++){
            ret = (int)(Math.random() * 10);
            distribution[ret]++;
        }
    }
}
```

確認問題 13

Q1

```
public class GamePlayer{

    public void play(String name){
        System.out.println(name + "で遊びました");
    }

    public void play(String name,int time){
        System.out.println(name + "を" + time + "時間鑑賞しました");
    }
}
```

```
public class Confirm13Q1{

    public static void main(String[] args){
        GamePlayer    instA;

        instA = new GamePlayer();

        instA.play("格闘ゲーム");
        instA.play("カンフー映画",2);

    }
}
```

Q2

```
public class Arithmetic{

    public int calculate(int no1){
        return (no1 * no1);
    }

    public int calculate(int no1,int no2){
        return (no1 - no2);
    }

    public int calculate(int no1,int no2,int no3){
        return (no1 + no2 + no3);
    }
}
```

```
public class Confirm13Q2{

    public static void main(String[] args){
        Arithmetic instA;

        instA = new Arithmetic();

        System.out.println("数値を1つ[5]渡した結果      :" + instA.calculate(5));
        System.out.println("数値を2つ[8,5]渡した結果    :" + instA.calculate(8,5));
        System.out.println("数値を3つ[11,8,5]渡した結果:" + instA.calculate(11,8,5));
    }
}
```

確認問題 14

Q1

```
public class Car{

    public void drive(int num){
        System.out.println( num + "Km走りました");
    }
}
```

```
public class Bus extends Car{

    public void put(int no){
        System.out.println( no + "人乗せました");
    }

    public void drop(int no){
        System.out.println( no + "人降ろしました");
    }
}
```

```
public class PatrolCar extends Car{

    public void siren(){
        System.out.println("サイレンを鳴らしました");
    }
}
```

```
public class Confirm14Q1{

    public static void main(String[] args){
        Bus        instBus;
        PatrolCar  instPCar;

        instBus = new Bus();
        instPCar = new PatrolCar();

        instBus.put(5);
        instBus.drive(20);
        instBus.drop(3);
```

続く

```
        instPCar.siren();
        instPCar.drive(10);
    }
}
```

Q2

```
public class AddSubtract{

    int result;

    public void add(int no1,int no2){
        result = no1 + no2;
    }

    public void subtract(int no1,int no2){
        result = no1 - no2;
    }

    public void display(){
        System.out.println("現在の結果は[" + result + "]です");
    }
}
```

```
public class FourArithmetic extends AddSubtract{

    public void multiply(int no1,int no2){
        result = no1 * no2;
    }

    public void divide(int no1,int no2){
        result = no1 / no2;
    }
}
```

```
public class Confirm14Q2{

    public static void main(String[] args){
        FourArithmetic   instA;

        instA = new FourArithmetic();
```

続く

```
        instA.add(10,5);
        instA.display();

        instA.subtract(10,5);
        instA.display();

        instA.multiply(10,5);
        instA.display();

        instA.divide(10,5);
        instA.display();
    }
}
```

確認問題 15

Q1

```
public class Bird{

    public void move(){
        System.out.println( "飛んで移動します。" );
    }
    public void eat(){
        System.out.println( "エサを食べます。" );
    }
}
```

```
public class Sparrow extends Bird{

    public void eat(){
        System.out.println( "昆虫などを食べます。" );
    }
}
```

```
public class Penguin extends Bird{

    public void move(){
        System.out.println( "泳いで移動します。" );
    }
```

続く

```
    public void eat(){
        System.out.println( "魚を食べます。" );
    }
}
```

```
public class Confirm15Q1{

    public static void main(String[] args){
        Sparrow   instSparrow;
        Penguin   instPenguin;

        instSparrow = new Sparrow();
        instPenguin = new Penguin();

        System.out.println( "スズメは" );
        instSparrow.move();
        instSparrow.eat();

        System.out.println( "ペンギンは" );
        instPenguin.move();
        instPenguin.eat();
    }
}
```

Q2

```
public class AdultSparrow extends Sparrow{

    public void eat(){
        super.eat();
        System.out.println( "大人になると主食が種子に変化します。" );
    }
}
```

```
public class Confirm15Q2{

    public static void main(String[] args){
        AdultSparrow    instSparrow;
        instSparrow = new AdultSparrow();

        System.out.println( "スズメは" );
        instSparrow.move();
```

続く

```
        instSparrow.eat();
    }
}
```

確認問題 16

Q1

```
public abstract class Arbeit{

    int    wages;

    public abstract void calculate(int hour);

    public void display(){
        System.out.println( wages + "円入手しました。" );
    }
}
```

```
public class Convenience extends Arbeit{

    public void calculate(int hour){
        System.out.println( "コンビニで" +hour + "時間働きました。" );
        wages = hour * 1000;
    }
}
```

```
public class CDshop extends Arbeit{

    public void calculate(int hour){
        System.out.println( "CDショップで" +hour + "時間働きました。);
        wages = hour * 850;
    }
}
```

```
public class GasStation extends Arbeit{

    public void calculate(int hour){
        System.out.println( "スタンドで" + hour + "時間働きました。" );
        wages = hour * 1000 + 500;
    }
}
```

続く

```
public class Confirm16Q1{

    public static void main(String[] args){
        GasStation     instA = new GasStation();
        Convenience    instB = new Convenience();
        CDshop         instC = new CDshop();

        instA.calculate(3);
        instA.display();
        instB.calculate(4);
        instB.display();
        instC.calculate(2);
        instC.display();
    }
}
```

Q2

```
public abstract class Student{

    public void study(int hour){
        System.out.println( "自宅で" + hour + "時間勉強しました。" );
    }

    public abstract void work(int hour);
}
```

```
public class Pattern1 extends Student{

    public void work(int hour){
        Convenience inst = new Convenience();
        inst.calculate(hour);
        inst.display();
    }
}
```

```
public class Pattern2 extends Student{

    public void work(int hour){
        System.out.println( "お手伝いを" + hour + "時間しました。");
    }
}
```

続く

```
public class Confirm16Q2{

    public static void main(String[] args){
        Pattern1 tomoko       = new Pattern1();
        Pattern1 tatsuhiro    = new Pattern1();
        Pattern2 nobuyasu     = new Pattern2();

        System.out.println( "tomokoの生活です。" );
        tomoko.work(3);
        tomoko.study(3);
        System.out.println( "tatsuhiroの生活です。" );
        tatsuhiro.work(5);
        tatsuhiro.study(1);
        System.out.println( "nobuyasuの生活です。" );
        nobuyasu.work(2);
        nobuyasu.study(5);
    }
}
```

確認問題 17

Q1

```
public interface DataSort{

    public void sort(int[] data);

    public void display();
}
```

```
public class SelectionSort implements DataSort{

    int[] data;

    public void sort(int[] data){
        int i,j;     //添え字
        int tmp;     //swap用変数

        //並べ替え(選択法)
        for( i = 0 ; i < data.length - 1 ; i++ ){
            for( j = i ; j < data.length ; j++ ){
                //swap
```

続く

```
                if( data[i] > data[j] ){
                    tmp = data[i];
                    data[i] = data[j];
                    data[j] = tmp;
                }
            }
        }

        //フィールドに格納
        this.data = new int[data.length];
        for (i = 0 ; i < data.length ; i++){
            this.data[i] = data[i];
        }
    }

    public void display(){
        for ( int i = 0 ; i < data.length ; i++){
            System.out.print(data[i] + "  ");
        }
        System.out.println();
    }
}
```

```
public class BubbleSort implements DataSort{

    int[] data;

    public void sort(int[] data){
        int i,j;    //添え字
        int tmp;    //swap用変数

        //並べ替え(隣接変換法)
        for( i = data.length - 1 ; i > 0 ; i-- ){
            for( j = 0 ; j < i ; j++ ){
                //swap
                if( data[j] > data[j+1] ){
                    tmp = data[j];
                    data[j] = data[j+1];
                    data[j+1] = tmp;
                }
```

続く

```
            }
        }

        //フィールドに格納
        this.data = new int[data.length];
        for (i = 0 ; i < data.length ; i++){
            this.data[i] = data[i];
        }
    }

    public void display(){
        for ( int i = 0 ; i < data.length ; i++){
            System.out.print(data[i] + "  ");
        }
        System.out.println();
    }
}
```

```
public class Confirm17Q1{

    public static void main(String[] args){
        int[] data = {95,66,39,85,60,28,58,19,18,29,77,33,24};
        SelectionSort    sort1 = new SelectionSort();
        BubbleSort       sort2 = new BubbleSort();

        System.out.println( "選択法" );
        sort1.sort(data);
        sort1.display();

        System.out.println( "隣接交換法" );
        sort2.sort(data);
        sort2.display();
    }
}
```

Q2

```
public class Phone{

    public String no;    //電話番号
```

続く

```
    public void tell(){
        System.out.println("電話できます。");
    }
}
```

```
public interface Music{

    public void listen();
}
```

```
public class Smartphone extends Phone implements Music{

    public String address;  //メールアドレス

    public void mail(){
        System.out.println("メールできます。");
    }

    public void listen(){
        System.out.println("音楽が聴けます。");
    }
}
```

```
public class Confirm17Q2{

    public static void main(String[] args){
        Smartphone inst = new Smartphone();

        inst.tell();
        inst.mail();
        inst.listen();
    }
}
```

確認問題 18

Q1

```
public class BirdFeature{

    public void showName(){
        System.out.println("鳥の名前を表示します。");
    }

    public void about(){
        System.out.println("特徴や習性を表示します。");
    }
}
```

```
public class Kestrel extends BirdFeature{

    public void showName(){
        System.out.println("タカ目ハヤブサ科チョウゲンボウ");
    }

    public void about(){
        System.out.println("素早く羽ばたいてホバリングを行った後、急降下
        して獲物を捕らえる。");
    }
}
```

```
public class Merlin extends BirdFeature{

    public void showName(){
        System.out.println("タカ目ハヤブサ科コチョウゲンボウ");
    }

    public void about(){
        System.out.println("日本でも田園なので見ることができる冬鳥。");
        System.out.println("ハトくらいの大きさ。");
    }
}
```

```
public class Dictionary{

    public void showBirdFeature(BirdFeature birdFeature){
```

続く

```
        birdFeature.showName();
        birdFeature.about();
    }
}
```

```
public class Confirm18Q1{

    public static void main(String[] args){
        Dictionary dic  = new Dictionary();
        Kestrel kestrel = new Kestrel();
        Merlin  merlin  = new Merlin();

        dic.showBirdFeature(kestrel);
        dic.showBirdFeature(merlin);
    }
}
```

Q2

```
public interface Communication{

    public void communicate();
}
```

```
public class Tiger implements Communication{

    public void communicate(){
        System.out.println("トラと触れ合うのは危険です。");
        System.out.println("遠くからエサを投げましょう。");
    }
}
```

```
public class Mustang implements Communication{

    public void communicate(){
        System.out.println("野生馬は警戒心が強いです。");
        System.out.println("エサを置いたら遠くで見守りましょう。");
    }
}
```

```
public class Dolphin implements Communication{
```

続く

```
    public void communicate(){
        System.out.println("イルカはひとなつっこいです。");
        System.out.println("頭をなでることができます。");
    }
}
```

```
public class JavaSafari{

    public void challenge(Communication animal){
        animal.communicate();
    }
}
```

```
public class Confirm18Q2{

    public static void main(String[] args){
        JavaSafari safari = new JavaSafari();
        Tiger tiger       = new Tiger();
        Mustang mustang   = new Mustang();
        Dolphin dolphin   = new Dolphin();

        safari.challenge(tiger);
        safari.challenge(mustang);
        safari.challenge(dolphin);
    }
}
```

確認問題 19

Q1

```
public class Fruit{

    String name;
    int price;

    public Fruit(String name,int price){
        this.name = name;
        this.price = price;
    }

    public int buy(int cnt){
        return price * cnt;
    }
}
```

```
public class Confirm19Q1{

    public static void main(String[] args){
        int total = 0;
        Fruit f1,f2,f3;

        f1 = new Fruit("りんご",100);
        f2 = new Fruit("みかん",50);
        f3 = new Fruit("ぶどう",200);

        total = f1.buy(3) + f2.buy(6) + f3.buy(2);
        System.out.println( "合計は" + total + "円です。" );
    }
}
```

Q2

```
public class Country{

    String name;
    double population;

    public Country(){
        name = "日本";
        population = 1.3;
    }

    public Country(String name,double population){
        this.name = name;
        this.population = population;
    }

    public void show(){
        System.out.println( name + "の人口は" + population + "億人
        です。" );
    }
}
```

```
public class Confirm19Q2{

    public static void main(String[] args){
        Country  inst1,inst2,inst3;

        inst1 = new Country();
        inst2 = new Country("中国",13.2);
        inst3 = new Country("インド",11.0);

        inst1.show();
        inst2.show();
        inst3.show();
    }
}
```

確認問題 20

Q1

```
package Confirm20;

public class HobyCls{

    String hobby;

    public HobyCls(String hobby){
        this.hobby = hobby;
    }

    public void display(){
        System.out.println("趣味は[" + hobby + "]です");
    }
}
```

```
package Confirm20;

public class NameCls{

    String name;

    public NameCls(String name){
        this.name = name;
    }

    public void display(){
        System.out.println("名前は[" + name + "]です");
    }
}
```

```
public class Confirm20Q1{

    public static void main(String args[]){
        Confirm20.NameCls   instNameCls;
        Confirm20.HobyCls   instHobyCls;

        instNameCls = new Confirm20.NameCls("ＳＣＣ太郎");
        instHobyCls = new Confirm20.HobyCls("盆栽");
```

続く

```
        instNameCls.display();
        instHobyCls.display();
    }
}
```

Q2

```
package introduce;

public class HobyCls{

    String hobby;

    public HobyCls(String hobby){
        this.hobby = hobby;
    }

    public void display(){
        System.out.println("趣味は[" + hobby + "]です");
    }
}
```

```
package introduce;

public class NameCls{

    String name;

    public NameCls(String name){
        this.name = name;
    }

    public void display(){
       System.out.println("名前は[" + name + "]です");
    }
}
```

```
public class Confirm20Q2{

    public static void main(String[] args){
        introduce.NameCls   instNameCls;
```

続く

```
        introduce.HobyCls   instHobyCls;

        instNameCls = new introduce.NameCls("ＳＣＣ太郎");
        instHobyCls = new introduce.HobyCls("盆栽");

        instNameCls.display();
        instHobyCls.display();
    }
}
```

確認問題 21

Q1

```
import introduce.HobyCls;
import introduce.NameCls;

public class Confirm21Q1{

    public static void main(String[] args){
        NameCls   instNameCls;
        HobyCls   instHobyCls;

        instNameCls = new NameCls("ＳＣＣ太郎");
        instHobyCls = new HobyCls("盆栽");

        instName.display();
        instHobyCls.display();
    }
}
```

確認問題 22

Q1

```
public class Empl1{

    public int     emplNo;
    public String  depart;
    public String  name;
    public int     salary;

    Empl1(int emplNo,String depart,String name,int salary){
        this.emplNo = emplNo;
        this.depart = depart;
        this.name = name;
        this.salary = salary;
    }
}
```

```
public class Confirm22Q1{

    public static void main(String[] args){
        Empl1  instA;

        instA = new Empl1(1,"営業部","ＳＣＣ二郎",200000);

        System.out.println("社員番号[" + instA.emplNo + "]");
        System.out.println("部署名  [" + instA.depart + "]");
        System.out.println("名前    [" + instA.name + "]");
        System.out.println("基本給  [" + instA.salary + "]");
    }
}
```

Q2

```
public class Empl2{

    private int     emplNo;
    private String  depart;
    private String  name;
    private int     salary;

    Empl2(int emplNo,String depart,String name,int salary){
```

続く

```
        this.emplNo = emplNo;
        this.depart = depart;
        this.name = name;
        this.salary = salary;
    }

    public int getEmplNo(){
        return emplNo;
    }

    public String getDepart(){
        return depart;
    }

    public String getName(){
        return name;
    }

    public int getSalary(){
        return salary;
    }
}
```

```
public class Confirm22Q2{

    public static void main(String[] args){
        Empl2  instA;

        instA = new Empl2(1,"営業部","ＳＣＣ二郎",200000);

        System.out.println("社員番号[" + instA.getEmplNo() + "]");
        System.out.println("部署名  [" + instA.getDepart() + "]");
        System.out.println("名前    [" + instA.getName() + "]");
        System.out.println("基本給  [" + instA.getSalary() + "]");
    }
}
```

Q3 4種類

Q4 private、修飾子なし（デフォルト）、protected、public

Q5

private	• 自クラス内からのみアクセス可能。
修飾子なし	• 同一パッケージ内からアクセス可能。
protected	• 同一パッケージ内、および継承して作られたサブクラスからアクセス可能。
public	• どのクラスからもアクセス可能。

Q6 private、修飾子なし（デフォルト）、protected、public

Q7 修飾子なし（デフォルト）、public

Q8

abstract	• 抽象クラスとも呼ばれ、このクラスから直接インスタンスを生成することはできない。
final	• 継承（主としてオーバーライド）できないことを意味する。このクラスをもとにインスタンス化は可能。

Q9 静的メソッドと呼ばれ、インスタンスの生成を行わずに使用することができるメソッド。

Q10 静的変数と呼ばれ、インスタンスの生成を行わずに使用することができる。またインスタンスが生成された場合には、そのクラスとすべてのインスタンス間でただひとつの領域として共有される。

Q11

- ローカル変数

  メソッドの内側で定義する。そのメソッド内でのみ有効となる。

- インスタンス変数

  すべてのメソッドの外側で定義する。各インスタンスごとに確保され、クラス内のすべてのメソッドから参照できる。

- クラス変数

  すべてのメソッドの外側で定義し、かつstatic修飾された変数。そのクラスおよび生成されたインスタンス全体で共通の領域となる。

確認問題 23

Q1

```
public class Confirm23Q1{

    public static void main(String[] args){
        Confirm23Q1 instA;
        int result;

        instA = new Confirm23Q1();

        result = instA.multiply(Integer.parseInt(args[0]),Integer.parseInt(args[1]));

        System.out.println("数値1[" + args[0] + "]");
        System.out.println("数値2[" + args[1] + "]");
        System.out.println("乗算結果[" + result + "]");
    }

    private int multiply(int no1,int no2){
        return (no1 * no2);
    }
}
```

Q2

```
public class Confirm23Q2{

    public static void main(String[] args){
        Confirm23Q2 instA;
        int result;

        instA = new Confirm23Q2();

        result = instA.multiply(Integer.parseInt(args[0]),Integer.parseInt(args[1]));

        System.out.println("数値1[" + args[0] + "]");
        System.out.println("数値2[" + args[1] + "]");
        System.out.println("乗算結果[" + result + "]");
    }

    private int multiply(int no1,int no2){
        return (no1 * no2);
```

続く

```
    }
}
```

Q3

```
public class Confirm23Q3{

    public static void main(String[] args){
        Confirm23Q3 instA;
        int result=0;

        instA = new Confirm23Q3();

        try{
            result = instA.multiply(Integer.parseInt(args[0]),Integer.parseInt(args[1]));
        }catch(Exception e){
            System.out.println("数字ではない値が渡されました");
        }

        System.out.println("数値1[" + args[0] + "]");
        System.out.println("数値2[" + args[1] + "]");
        System.out.println("乗算結果[" + result + "]");
    }

    private int multiply(int no1,int no2){
        return (no1 * no2);
    }
}
```

確認問題 24

Q1

```
public class GreetThread1 extends Thread{

    GreetThread1(String name){
        super(name);
    }

    public void run(){
        int i;

        for(i=0;i<20;i++){
            System.out.println(this.getName() + "：" + i);
        }
    }
}
```

```
public class Confirm24Q1{

    public static void main(String[] args){
        GreetThread1   thread1,thread2;

        thread1 = new GreetThread1("おはようThread ");
        thread2 = new GreetThread1("こんにちはThread");

        thread1.start();
        thread2.start();
    }
}
```

Q2

```
public class GreetThread2 extends Thread{

    GreetThread2(String name){
        super(name);
    }

    public void run(){
        int i;
```

続く

```
        for(i=0;i<20;i++){
            System.out.println(this.getName() + "：" + i);
            try{
                Thread.sleep(200);
            }catch(Exception e){

            }
        }
    }
}
```

```
public class Confirm24Q2{

    public static void main(String[] args){
        GreetThread2   thread1,thread2;

        thread1 = new GreetThread2("おはようThread ");
        thread2 = new GreetThread2("こんにちはThread");

        thread1.start();
        thread2.start();
    }
}
```

Q3

```
public class MorningThread extends Thread{

    String str;

    MorningThread(String str){
        this.str = str;
    }

    public void run(){
        int i;

        for(i=0;i<20;i++){
            System.out.println(str + "：" + i);
            try{
                Thread.sleep(200);
```

続く

```
            }catch(Exception e){

            }
        }
    }
}
```

```
public class DaytimeThread implements Runnable{

    String str;

    DaytimeThread(String str){
        this.str = str;
    }

    public void run(){
        int i;

        for(i=0;i<20;i++){
            System.out.println(str + "：" + i);
            try{
                Thread.sleep(200);
            }catch(Exception e){

            }
        }
    }
}
```

```
public class Confirm24Q3{

    public static void main(String[] args){
        MorningThread   thread1;
        DaytimeThread   instA;
        Thread          thread2;

        thread1 = new MorningThread("おはようThread ");
        instA = new DaytimeThread("こんにちはThread");
        thread2 = new Thread(instA);
```

続く

```
        thread1.start();
        thread2.start();
    }
}
```

確認問題 25

Q1

```
import java.util.LinkedList;

public class QueueList<T>{

    LinkedList<T>  list;

    QueueList(){
        list = new LinkedList<T>();
    }

    public int enqueue(T element){
        list.addLast(element);
        System.out.println("追加した要素:" + element);
        return list.size();
    }

    public T dequeue(){
        T ret =list.getFirst();
        System.out.println("取り出した要素:" + ret);
        list.removeFirst();
        return ret;
    }
}
```

```
public class Confirm25Q1{

    public static void main(String[] args){
        QueueList<Integer>   list;

        list = new QueueList<Integer>();

        list.enqueue(1);
```

続く

```
        list.enqueue(2);
        list.enqueue(3);

        list.dequeue();
        list.dequeue();
        list.dequeue();
    }
}
```

Q2

```
import java.util.LinkedList;

public class StackList<E>{

        LinkedList<E> list;

        StackList(){
             list = new LinkedList<E>();
        }

        public int push(E element){
            list.addLast(element);
            System.out.println("追加した要素:" + element);
            return list.size();
        }

        public E pop(){
            E ret = list.getLast();
            System.out.println("取り出した要素:" + ret);
            list.removeLast();
            return ret;
        }
}
```

```
public class Confirm25Q2{

    public static void main(String[] args){

        QueueList<Integer>   quelist;
        StackList<Integer>   stacklist;
        int                  i;
```

続く

```
        quelist = new QueueList<Integer>();
        stacklist = new StackList<Integer>();

        //データの格納(3要素)
        System.out.println("キューにデータを追加します");
        quelist.enqueue(1);
        quelist.enqueue(2);
        quelist.enqueue(3);

        System.out.println("スタックにデータを追加します");
        stacklist.push(1);
        stacklist.push(2);
        stacklist.push(3);

        //データの取り出し
        System.out.println("キューからデータを取り出します");
        for(i=0;i<3;i++){
            quelist.dequeue();
        }

        System.out.println("スタックからデータを取り出します");
        for(i=0;i<3;i++){
            stacklist.pop();
        }
    }
}
```

Q3

```
public class ConfirmClass1{

    public int val;

    ConfirmClass1(int val){
        this.val = val;
    }
}
```

```
public class ConfirmClass2{
```

続く

```
    private int val;

    ConfirmClass2(int val){
        this.val = val;
    }

    public int getVal(){
        return val;
    }
}
```

```
public class ConfirmClass3{

    public void display(){
        System.out.println("確認クラス３です");
    }
}
```

```
import java.util.ArrayList;

public class Confirm25Q3{

    public static void main(String[] args){
        ArrayList<Object>  list;

        list = new ArrayList<Object>();

        list.add(new ConfirmClass1(10));
        list.add(new ConfirmClass2(20));
        list.add(new ConfirmClass3());

        ConfirmClass1   class1 = (ConfirmClass1)list.get(0);
        System.out.println("確認クラス１のval：" + class1.val);
        ConfirmClass2   class2 = (ConfirmClass2)list.get(1);
        System.out.println("確認クラス２のval：" + class2.getVal());
        ConfirmClass3   class3 = (ConfirmClass3)list.get(2);
        class3.display();
    }
}
```

確認問題 26

Q1

```
import java.io.*;

public class KeyboardInput{
    String buff = null;
    BufferedReader inputKB;

    public String inputKeyboard(){
        inputKB = new BufferedReader(
                        new InputStreamReader(System.in));

        try{
            buff = inputKB.readLine();
        }catch(IOException e){
            System.out.println("KBエラー");
        }
        return buff;
    }
}
```

```
public class Confirm26Q1{

    public static void main(String[] args){

        KeyboardInput   input;
        String          name,times;
        int             i;

        input = new KeyboardInput();

        System.out.print("名前を入力してください->");
        name  = input.inputKeyboard();
        System.out.print("回数を入力してください->");
        times = input.inputKeyboard();

        System.out.println("*** 今から表示します ***");
        for(i=0;i<Integer.parseInt(times);i++){
            System.out.println(name);
```

続く

```
        }
        System.out.println("***表示を終了しました***");
    }
}
```

Q2

```
import java.io.*;

public class Confirm26Q2{

    public static void main(String[] args){

        BufferedWriter buffw = null;
        KeyboardInput  input;
        String         name,times;
        int            i;

        try{
            buffw = new BufferedWriter(
                            new FileWriter("confirm26.txt"));

            input = new KeyboardInput();

            System.out.print("名前を入力してください->");
            name = input.inputKeyboard();
            System.out.print("回数を入力してください->");
            times = input.inputKeyboard();

            System.out.println("*** 今から表示します ***");
            buffw.write("*** 今から表示します ***");
            buffw.newLine();
            for(i=0;i<Integer.parseInt(times);i++){
                System.out.println(name);
                buffw.write(name);
                buffw.newLine();
            }
            System.out.println("***表示を終了しました***");
            buffw.write("***表示を終了しました***");
            buffw.newLine();
        }catch(IOException e){
```

続く

```
            System.out.println("書き込みエラーです");
        }finally{
            try{
                buffw.flush();
                buffw.close();
            }catch(Exception e){
                System.out.println("終了処理エラーです");
            }
        }
    }
}
```

Q3

```
import java.io.*;

public class Confirm26Q3{

    public static void main(String[] args){

        KeyboardInput   input;
        String    val;
        int       no1,no2,work,total,start,main;

        input = new KeyboardInput();

        try{
            System.out.print("1つ目の数値を入力してください->");
            val  = input.inputKeyboard();
            no1 = Integer.parseInt(val);
            System.out.print("2つ目の数値を入力してください->");
            val = input.inputKeyboard();
            no2 = Integer.parseInt(val);

            if(no1>no2){
                work = no1;
                no1 = no2;
                no2 = work;
            }

            start = no1;
```

続く

```
            for(total=0;start<=no2;start++){
                total += start;
            }
            System.out.println("[" + no1 + "]から" + "[" + no2 + "]までの合計:" + total );
        }catch(Exception e){
            System.out.println("数値以外が入力されました");
        }
    }
}
```

確認問題 27

Q1

```
import java.sql.*;
import java.util.ArrayList;

public class DBSelect{

  static final String URL =
    "jdbc:mysql://localhost/gradedb?useSSL=false&serverTimezone=JST";

  ArrayList<Student> selectAll(String USERNAME, String PASSWORD){
    ArrayList<Student> listRet = null;
    int       score = 0;
    String    number,name;
    listRet = new ArrayList<Student>();

    try (Connection connection = DriverManager.getConnection
      (URL, USERNAME, PASSWORD);
      Statement stmt = connection.createStatement(); ){
      String sql = "SELECT * FROM ENGLISH";
      ResultSet rslt = stmt.executeQuery(sql);
      while(rslt.next()){
        number = rslt.getString("STUDENTNO");
        name = rslt.getString("STUDENTNAME");
        score = rslt.getInt("SCORE");
        listRet.add(new Student(number,name,score));
      }
    } catch (Exception e) {
```

続く

```
      e.printStackTrace();
    }
    return listRet;
  }
}
```

```
public class Student{

    private String number;
    private String name;
    private int    score;

    public Student(String number,String name,int score){
        this.number = number;
        this.name = name;
        this.score = score;
    }

    public String getNumber(){
        return number;
    }

    public String getName(){
        return name;
    }

    public int getScore(){
        return score;
    }
}
```

```
import java.util.ArrayList;

public class Confirm27Q1{

  static String USERNAME = null;   // ユーザ名
  static String PASSWORD = null;   // パスワード

  public static void main(String[] args){
    ArrayList<Student> result = new ArrayList<Student>();
```

続く

```
    DBSelect  sqlSelect = new DBSelect();

    GetParam getParam = new GetParam();
    USERNAME = getParam.getKeyin(" ユーザ名    : ");
    PASSWORD = getParam.getKeyin(" パスワード  : ");
    System.out.println();

    result = sqlSelect.selectAll(USERNAME,PASSWORD);

    for(int i = 0;i<result.size();i++){
      Student inst;
      inst = result.get(i);
      System.out.print(inst.getNumber() + "¥t");
      System.out.print(inst.getName() + "¥t");
      System.out.println(inst.getScore());
    }
  }
}
```

Q2

```
import java.sql.*;

public class DBInsert{
  static final String URL =
    "jdbc:mysql://localhost/gradedb?useSSL=false&serverTimezone=JST";

  public void insert(String USERNAME, String PASSWORD, Student student){
    try (Connection connection = DriverManager.getConnection
                                    (URL, USERNAME, PASSWORD);
      Statement statement = connection.createStatement();) {
      String sql = "INSERT INTO english (studentno, studentname, score)"
        + " VALUES (" + student.getNumber() + ", '" + student.getName()
        + "', " + student.getScore() + ");";
      int result = statement.executeUpdate(sql);
      System.out.println("挿入件数:" + result);
    } catch (SQLException e) {
        e.printStackTrace();
    }
  }
}
```

続く

```
public class StudentCreate{
    Student  student;
    public Student inputStudent(){
        String   number,name;
        int      score;
        KeyboardInput  instKbin = new KeyboardInput();
        System.out.print("学生番号-->");
        number = instKbin.inputKeyboard();
        System.out.print("氏名-->");
        name = instKbin.inputKeyboard();
        System.out.print("点数-->");
        score = Integer.parseInt(instKbin.inputKeyboard());
        return (new Student(number,name,score));
    }
}
```

```
import java.util.ArrayList;

public class Confirm27Q2{

  static String USERNAME = null;   // ユーザ名
  static String PASSWORD = null;   // パスワード

  public static void main(String[] args){
    ArrayList<Student> result = new ArrayList<Student>();
    DBSelect  sqlSelect = new DBSelect();
    DBInsert  sqlInsert = new DBInsert();
    StudentCreate instCreate = new StudentCreate();

    GetParam getParam = new GetParam();
    USERNAME = getParam.getKeyin(" ユーザ名    : ");
    PASSWORD = getParam.getKeyin(" パスワード  : ");
    System.out.println();

    result = sqlSelect.selectAll(USERNAME, PASSWORD);
    for(Student inst : result){
      System.out.print(inst.getNumber() + "¥t");
      System.out.print(inst.getName() + "¥t");
      System.out.println(inst.getScore());
```

続く

```
    }
    System.out.println();

    Student instStudent = instCreate.inputStudent();
    sqlInsert.insert(USERNAME, PASSWORD, instStudent);

    System.out.println();

    result = sqlSelect.selectAll(USERNAME, PASSWORD);
    for(Student inst : result){
      System.out.print(inst.getNumber() + "¥t");
      System.out.print(inst.getName() + "¥t");
      System.out.println(inst.getScore());
    }
    System.out.println();
  }
}
```

確認問題 28

Q1

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

public class Confirm28Q1 extends JFrame implements ItemListener,ActionListener{
    private FlowLayout    flow;
    private JTextField    text;
    private JButton       button;
    private ButtonGroup   btngrp;
    private JRadioButton  rbtn1,rbtn2,rbtn3;
    private String        str;
    private Container     cntnr;

    Confirm28Q1(String title){
        super(title);
        setBounds(200,200,350,200);
    }

    public static void main(String[] args){
        Confirm28Q1   frame;

        frame = new Confirm28Q1("Confirm28Q1");

        frame.defineVal();
        frame.setVisible(true);
    }

    void defineVal(){
        cntnr = getContentPane();
        flow = new FlowLayout();
        cntnr.setLayout(flow);

        btngrp = new ButtonGroup();
        rbtn1  = new JRadioButton("和食",false);
        rbtn2  = new JRadioButton("フレンチ",false);
        rbtn3  = new JRadioButton("イタリアン",false);
```

続く

```
        text   = new JTextField(20);
        button = new JButton("今選択されているのは？");

        btngrp.add(rbtn1);
        btngrp.add(rbtn2);
        btngrp.add(rbtn3);
        cntnr.add(rbtn1);
        cntnr.add(rbtn2);
        cntnr.add(rbtn3);
        cntnr.add(button);
        cntnr.add(text);

        setDefaultCloseOperation(EXIT_ON_CLOSE);
        rbtn1.addItemListener(this);
        rbtn2.addItemListener(this);
        rbtn3.addItemListener(this);
        button.addActionListener(this);
    }

    public void itemStateChanged(ItemEvent e){
        str = ((JRadioButton)e.getItemSelectable()).getText();
    }

    public void actionPerformed(ActionEvent e){
        if(str != null){
            text.setText("よし！" + str + "にしよう！");
        }else{
            text.setText("う〜ん　何にしよう？");
        }
    }
}
```

確認問題 29

下記のコードは、旧誌の記述です。サンプルとして記載を残していますが、アプレットの実行はJavaやWebブラウザのバージョンに依存するため、実行は保証しておりません。

Q1

```
import java.applet.Applet;
import java.awt.Graphics;
import java.awt.Button;
import java.awt.Choice;
import java.awt.Color;
import java.awt.event.ActionListener;
import java.awt.event.ItemListener;
import java.awt.event.ActionEvent;
import java.awt.event.ItemEvent;

public class Confirm29Q1 extends Applet implements ItemListener,ActionListener{

    private Choice  houseChoice;
    private Button  architect;
    private Button  clear;
    private String[] itemHouse = {"煙突","屋根","壁","窓"};
    private int x[] = { 175,75,275 };
    private int y[] = { 90,200,200 };
    private int index=-1;
    private boolean clearFlg = false;

    public void init(){
        int i;

        houseChoice = new Choice();
        architect = new Button("建築");
        clear = new Button("クリア");

        for(i=0;i<4;i++){
            houseChoice.addItem(itemHouse[i]);
        }

        add(houseChoice);
        add(architect);
        add(clear);
```

続く

```
        architect.addActionListener(this);
        clear.addActionListener(this);
        houseChoice.addItemListener(this);
    }

    public void paint(Graphics g){
        if(clearFlg == true){
            g.setColor(getBackground());
            g.fillRect(0,0,getSize().width,getSize().height);
            clearFlg = false;
        }else{
            switch(index){
                case 0:                              //煙突
                    g.fillRect(220,130,20,50);
                    break;
                case 1:                              //屋根
                    g.setColor(Color.red);
                    g.fillPolygon(x,y,3);
                    break;
                case 2:                              //壁
                    g.setColor(Color.blue);
                    g.fillRect(100,200,150,100);
                    break;
                case 3:                              //窓
                    g.setColor(Color.white);
                    g.fillRect(150,225,50,40);
                    g.setColor(Color.black);
                    g.drawLine(175,225,175,265);
                    g.drawLine(150,245,200,245);
                    break;
                default:
                        break;
              }
         }
     }

     public void update(Graphics g){
         paint(g);
    }
```

続く

```
    public void actionPerformed(ActionEvent e){
        if(e.getSource() == architect){
            if(index == -1){
                index = 0;
            }
        }else{
            clearFlg = true;
        }
        repaint();
    }

    public void itemStateChanged(ItemEvent e){
        index = ((Choice)e.getSource()).getSelectedIndex();
    }
}
```

```
<HTML>

<applet code="Confirm29Q1.class" width=500 height=400>
</applet>

</HTML>
```