

通過テストの解答

1 プログラムの書き方

1. ⑤④③⑥①② または ⑤④③⑥②①

```
2. package exercise;
   public class MyHello {
       public static void main(String [] args){
           System.out.println("プログラムのかたち");
       }
   }
```

3. B, E

【解説】 B は複数行に渡れない。 E は入れ子にできない

```
4. public class P4 {

    public static void main(String[] args) {
        System.out.println("何時ですか");
        System.out.println("12:30 分です。");
    }
}
```

2 いろいろなデータ型

- (1) double (2) char (3) short (4) float (5) int
(6) boolean (7) byte (8) long (9) String
- (1) String (2) char (3) long (4) float
(5) char (6) int (7) boolean (8) String
- ②先頭が数字 ③先頭が数字 ⑥-は使えない⑧#は使えない⑩予約語
- 4、8、9
- ① 10.5 は double なので代入できない
③ "true"は String なので代入できない
④ "A"は String なので代入できない
⑥ 宣言の中に char が2度書かれている。char ch1; char ch2;なら OK
⑧ 宣言する前に変数を使用している
⑨ 同じ変数名を2度宣言している
⑩ char c=ss;は char c = (char)ss; とキャストする必要がある
⑫ String は char にキャストできない
⑬ boolean はキャストできない

3 演算子と演算

- 1) $(2*a+3*b)*c$ 2) $2*a\%6$ 3) $-a*4*b$ 4) $a*a*a - 2*b*c$
- 1) $a +=b$; 2) $a--$; または $--a$; 3) $a *=(b-1)$ 4) $a=b=c=0$;
- $a + "$ と" + $b + "$ の合計は" + $(a+b) + "$ です"
- ① $10/3$ ② $11/3$ ③ `number is 2.510` ④ `number is 12.5`
- 1)② 2)④ 3)⑦ 4)⑦

【解説】 1) b には $c+1$ が代入されるので b は 2。 a も 2 になる。 c は 1 のまま
2) $a++$ なので式は $a=0$ で計算する。 a を評価した後で a の値は 1 増えて 1 になる
3) `int a=b=c=0;` は文法エラー。 `int a=0, b=0, c=0;` とすべきところ
4) $(b+1) +=a$; の部分は $(b+1) = (b+1) + a$; と展開できる。式の左辺は変数でなければならない。 $b+1$ は式であるため文法エラーになる

4 標準クラスの利用

- 1) `double x = Math.max(2*a, Math.pow(b,2));`
2) `double y = 2*a* Math.random() - a;`
3) `double z = Math.round((2*a+b)/c);`
- ① `Math.pow(b,2) - 4*a*c;`
② $(-b + \text{sq}) / (2*a)$
③ $(-b - \text{sq}) / (2*a)$
- ```
import lib.Input;
public class P3 {
 public static void main(String[] args) {
 double a,b,c;
 a = Input.getDouble();
 b = Input.getDouble();
 c = Input.getDouble();
 double x = Math.pow(a, 2) + Math.pow(b, 2) - 2*c;
 double y = Math.sqrt(a) + Math.sqrt(b) + 3*c;
 System.out.println("(1)="+x);
 System.out.println("(2)="+y);
 }
}
```

## 5 配列と for 文

- ```
import lib.Input;
public class P1 {
    public static void main(String[] args) {
        System.out.println("start");
        int m,n;
        for(int i=0; i<3; i++){
            m = Input.getInt();
            n = Input.getInt();
            System.out.println(n%m);
        }
        System.out.println("end");
    }
}
```
- ```
public class P2 {
 public static void main(String[] args) {
 int[] n = {102, 98, 122, -97, 88};
 double[] x = {12.5, 33.5, -12.7, 18.4, 13.33};
 String[] s = {"赤", "青", "黄", "白", "黒"};

 for(int i=0; i<5; i++){
 System.out.print(n[i]+"%t"); // %t はタブ
 }
 System.out.println(""); // 改行
 for(int i=0; i<5; i++){
 System.out.print(x[i]+"%t");
 }
 System.out.println(""); // 改行
 for(int i=0; i<5; i++){
 System.out.print(s[i]+"%t");
 }
 }
}
```
- ```
public class P3 {
    public static void main(String[] args) {
        int[]a = {10, 23, 13, 7 , 22};
        int[]b = {2, 3, 5, 3 , 4};
        for(int i=0; i<5; i++){
            System.out.println(a[i] + "x"
                               + b[i] + "=" + a[i]*b[i]);
        }
    }
}
```

4. (1) C --- System.out.println(n);は for 文の外なので 1 回しか実行されない
(2) E --- int i=0; i<4; i++ では i は 0,1,2,3 となるが配列は n[2] までしかない
ので実行時例外が発生してプログラムは停止する

6 配列と for 文

- ```
public class P1 {
 public static void main(String[] args) {
 double total = 0, mean = 0;
 double[] d = { 12.3, 13.5, 12.2, 13.0, 12.8, 12.3 };
 for (double x : d) {
 total += x;
 }
 mean = total/d.length;
 System.out.println("合計=" + total);
 System.out.println("平均=" + mean);
 for (double x : d) {
 System.out.print(x + " ");
 }
 }
}
```
- ```
public class P2 {
    public static void main(String[] args) {
        String[] name = {"田中", "中村", "鈴木", "山本", "本田"}; //(1)
        double[] language = {82, 85, 74, 90, 70}; //(2)
        double[] english = {70, 74, 88, 74, 82}; //(3)
        System.out.println(""); // 改行
        //(4)
        double total = 0;
        for(int i=0; i<name.length; i++){
            total += language[i];
        }
        System.out.println("国語平均=" + total/language.length);
        System.out.println(""); // 改行
        //(5)
        total = 0;
        for(int i=0; i<name.length; i++){
            total += english[i];
        }
        System.out.println("英語平均=" + total/english.length);
        System.out.println(""); // 改行
        //(6)
        for(int i=0; i<name.length; i++){
            System.out.println(name[i] + ": " + (language[i]+english[i]));
        }
    }
}
```

3. ① `int i=8` ② `i>=0`

【解説】最後の要素（番号は8）から初めて0番目まで出力します

4. (1) ③ 【解説】`i`は0から始めて8より小さい間、2ずつ増やしながら繰り返す。したがって、0 2 4 6 と出力される
- (2) ④ 【解説】`i`は`System.out.print(i++ + " ");`の中で1ずつ増加するが、表示の後で増加するので、開始は（`i`が0の時）0からで最後は（`i`が7の時）7まで表示する
- (3) ⑪ 【解説】 `ArrayIndexOutOfBoundsException`が発生する
`i`が7の時、`System.out.print(a[++i] + " ");`において、先にインクリメントされるので`i`は8になる。`a[8]`は存在しないため配列インデックスが境界を越えたという実行時エラーになる
- (4) ⑨ 【解説】 `i`の値が変化しない。常に1のみであるため無限に1を出力する
- (5) ⑩ 【解説】 `for`文に`{}`がないので`i++;`の部分は`for`文ではない。また、`i`は`for`文の中で宣言されているので、`for`文の外では存在しない。そのため`i++;`の部分で変数が宣言されていない（名前を解決できない）というコンパイルエラーが発生する
- (6) ⑩ 【解説】 最初の行で`ini i=0`として、`i`が宣言されているので、最初の`for`文ではこの`i`が使われ、問題なく動作する。これは、`for`文の外側で宣言した変数は`for`文の中でも有効であるためである。しかし、2つ目の`for`文では`for(int i=0; i<2; i++)`と、重ねて`i`を宣言しているのでここでコンパイルエラーになる。同じ名前の変数を重ねて宣言できないという文法違反にあたる

7

条件を書くための演算子

1. (1) `n>=100` (2) `n>=100 && n<500` (3) `n%2==0` (4) `(n%3==0 || n%2!=0) && n<100`
(5) `c>'t'+1` (6) `Math.sqrt(x)>2.0` (7) `s.equals("abcde")`
2. (1) ② (2) ③ (3) ① (4) ① (5) ③ (6) ② (7) ②
- 【解説】(5)は`int`の変数に`!`を付けている。`!`は`boolean`の値にのみ付けることが出来る
3. ①`true` ②`a=11 b=0`
- 【解説】`||`は短絡演算子なので①で`a++>b`が成立すると、後続の`a!=b++`をチェックしない。そのため`b`の値は増えない。
4. ①④⑥⑦
5. (1) ⑦ 【解説】`n+1`は`int`になる
- (2) ② 【解説】複合代入演算子`+=`は型を維持するのでコンパイルエラーではない
- (3) ⑤ 【解説】比較するだけなのでエラーにはならない。`c`は文字型なので正の数

8**while 文と電卓プログラム**

```
1. public class P1 {
    public static void main(String[] args) {
        int i=0;
        while(i<5){
            System.out.println(i+1+"回目");
            i++;
        }
    }
}
```

```
2. import lib.Input;
public class P2 {
    public static void main(String[] args) {
        String s;
        while((s=Input.getString())!=null){
            System.out.println(s);
        }
    }
}
```

```
3. import lib.Input;
public class P3 {
    public static void main(String[] args) {
        int n, total=0;
        while((n=Input.getInt())!=0){
            total += n;
        }
        System.out.println(total);
    }
}
```

```
4. public class P4 {
    public static void main(String[] args) {
        double[] d ={1.2, 3.3, 4.5, 10.0, 5.2, 1.3};
        int i=0;
        double total=0;
        while(i<d.length){
            total += d[i];
            i++;
        }
        System.out.println("合計"+total);
    }
}
```

```

5.
import lib.Input;
public class P5 {
    public static void main(String[] args) {
        double x, sum1=0, sum2=0;
        int n=0;
        while((x=Input.getDouble())!=0){
            sum1 += x;
            sum2 += Math.pow(x,2);
            n++;
        }
        System.out.println("合計="+sum1);
        System.out.println("平方和="+sum2);
        System.out.println("平均="+sum1/n);
    }
}

```

6.③

【解説】 **total** が **115** になった時点で反復が終了する。最後に **total** に入った **45** が取り消されることはない。**total** は **115** になったまま **while** 文を抜けて表示される

7. 1) ⑨ 【解説】 **while(true)** は常に条件が成り立つため無限ループになる
- 2) ⑥ 【解説】 **a** は **5,4,3,2,1** と減っていきます。**0** になった時条件不成立でループを出る
- 3) ④ 【解説】 **--a** なので、最初に **a** を **1** 減らしてから表示する。**a** は **5** から **1** までループを実行しますが、表示は **4** から **0** までになる
- 4) ⑪ 【解説】 **a[++i]** と前置の **++** なので、**i=0** のとき **a[1]** を出力する。ループの最後で **i** は **4** になり、**a[5]** を出力しようとするので実行時エラーになる。
- 5) ⑩ 【解説】 **b** が **do-while** の中で宣言されているため、**while(b>0)** でコンパイルエラーになる
- 6) ⑦ 【解説】 **{}** が無いので **a++** は **while** ループの外です。**while** 実行中は **a** は **1** のまま。**b** は **1** ずつ減っていき、**b=1** のとき **while** ループを出る。

9

if 文と投票集計プログラム

```

1.
import lib.Input;
public class P1 {
    public static void main(String[] args) {
        String s;
        while((s=Input.getString())!=null){
            if(s.equals("dog")){
                System.out.println("いぬ");
            }else if(s.equals("cat")){
                System.out.println("ねこ");
            }else if(s.equals("mouse")){
                System.out.println("ねずみ");
            }else if(s.equals("rabbit")){
                System.out.println("うさぎ");
            }else {
                System.out.println("?");
            }
        }
    }
}

```

【解説】 **Input.getString()** では、**Enter** キーをタイプすると **null** が入力されます (149 頁を参照してください)。

```

2. import lib.Input;
   public class P2 {
       public static void main(String[] args) {
           int m;
           while ((m = Input.getInt()) != 0) {
               if (m==12||m==1||m==2) {
                   System.out.println("冬");
               } else if (m==3||m==4||m==5) {
                   System.out.println("春");
               } else if(m==6||m==7||m==8){
                   System.out.println("夏");
               } else if(m==9||m==10||m==11){
                   System.out.println("秋");
               } else{
                   system.out.println("?");
               }
           }
       }
   }

```

```

3. public class P3 {
       public static void main(String[] args) {
           String[] name
           ={"田中","前田","鈴木","中村","田辺","浦川","島田","岩田"};
           int[] drink      = {7,0,2,4,3,0,0,6};
           int[] smoke      = {60,10,0,20,10,0,30,0};

           for(int i=0; i<name.length; i++){
               int d = drink[i];// そのままだと論理式が見にくくなるので
               int s = smoke[i];
               System.out.print(name[i] + " ");
               //
               if(d==0&&s==0){
                   System.out.println("安全");
               }else if(d==0&&s>0&&s<=20 ||d>0&&d<=3&&s==0){
                   System.out.println("注意");
               }else if(d>0&&d<=3&&s>0&&s<=20){
                   System.out.println("要指導");
               }else{
                   System.out.println("要検査");
               }
           }
       }
   }

```

4.②

【解説】 if に {} が使われていないので、else が作用するのは a=b だけ。b=0 は if 文の外なので常に実行される。したがって、b は 0 である。

5.①

【解説】 if(b=true) は b に true を代入しています。b==true ではないことに注意してください。そのため if 文は true となり画面にも "true" が表示されます。

10**switch 文と複数の場合分け**

```
1. import lib.Input;
public class P1 {
    public static void main(String[] args) {
        int n=Input.getInt();
        switch(n){
            case 1:
                System.out.println("java 言語講座");
                break;
            case 2:
                System.out.println("C++言語講座");
                break;
            case 3:
                System.out.println("web アプリ作成講座");
                break;
            default:
                System.out.println("?");
                break;
        }
    }
}
```

```
2. import lib.Input;
public class P2 {
    public static void main(String[] args) {
        char ch = Input.getChar();
        switch(ch){
            case 'a':
                System.out.println(100);
                break;
            case 'b':
            case 'c':
                System.out.println(200);
                break;
            case 'd':
                System.out.println(300);
                break;
            default:
                System.out.println(400);
                break;
        }
    }
}
```

3. ①13 【解説】 $2\%6$ は 2。case 2: で処理する

②4 【解説】 $6\%6$ は 0。case 0: で処理する

③6 【解説】 $10\%6$ は 4。case 4: で処理する

4. B, C

【解説】 A は case 文に変数 m を使っている。D は switch 文の $n\%3.0$ が double になり使えない B のキャストは有効。C で byte 型を使っているのも有効

11**break と continue**

```
1. import lib.Input;
public class P1 {
    public static void main(String[] args) {
        char c;
        while((c=Input.getChar())!=0){
            if(c=='e'){
                break; // while ループを脱出する
            }
            switch(c){
            case 'o':
                System.out.println("おはよう");
                break;
            case 'k':
                System.out.println("こんにちは");
                break;
            case 's':
                System.out.println("さようなら");
                break;
            default:
                System.out.println("?");
            }
        }
    }
}
```

```
2. public class P2 {
    public static void main(String[] args) {
        int[] dt = { 55, 60, 90, 45, 68, 70, 77, 80, 82, 95,
                    75, 170, 63, 65, 78, 50, 67, -90, 70, 55,
                    45, 68, 57, 65, 68, -72, 71, 66, 98, 48};
        int a=0,b=0,c=0,d=0,e=0;
        for(int i=0; i<dt.length; i++){
            if(dt[i]<0||dt[i]>100){
                continue;
            }
            if(dt[i]>=90){
                a++;
            }else if(dt[i]>=80){
                b++;
            }else if(dt[i]>=70){
                c++;
            }else if(dt[i]>=60){
                d++;
            }else{
                e++;
            }
        }
        System.out.println("A="+a);
        System.out.println("B="+b);
        System.out.println("C="+c);
        System.out.println("D="+d);
        System.out.println("E="+e);
    }
}
```

3. ①②③④⑤ 【解説】内側のループを、i が 0,1,2,3,4 までの 5 回繰り返した時に、break 文で for ループを脱出する
4. ⑥ 【解説】if についてのラベルは continue からの戻り先として指定できない
5. ②③⑥⑦

【解説】	i	j	j>2	《説明》
	0	1	×	i と j は左のように変化する。j>2 の場合、i と j の値は表示されないので、0:1 0:2 1:1 1:2 が表示される
	0	2	×	
	0	3	○	
	1	1	×	
	1	2	×	
	1	3	○	

12 配列の仕組み

- 1.A. String[] s1, s2;
 B. int[] n = new int[5];
 C. char[] ch=null; ch = new char[] { 'a', 'b', 'c'};

```

2.
import lib.Input;
public class P2 {
    public static void main(String[] args) {
        int n = Input.getInt();
        double[] x = new double[n];
        for( double a : x){
            System.out.println(a);
        }
    }
}

```

3. B,F

【解説】A-- m=0 はできない、C-- ch = { 'a', 'b', 'c'}; という代入は不可、D-- [4] のように要素数を指定できない、E-- [2] のように要素数を指定できない、G-- new {1,2,3} という書き方はない

4. C 【解説】str は初期化されていないので配列は存在しない
5. E 【解説】n には null が入っているので配列は存在しない

13**配列の操作**

```
1. import lib.Input;
public class P1 {
    public static void main(String[] args) {
        double[] d = new double[5];
        // 入力
        for(int i=0; i<d.length; i++){
            d[i] = Input.getDouble();
        }
        // 表示
        for(double x : d){
            System.out.print(x + "%t");
        }
        System.out.println(""); // 改行
        // 合計と平均
        double sum = 0;
        for(double x : d){
            sum += x;
        }
        System.out.println("合計=" + sum);
        System.out.println("平均=" + sum/d.length);
    }
}
```

```
2. public class P2 {
    public static void main(String[] args) {
        int[] n1 = {6, 9, 3, 1, 8};
        // コピー
        int[] n2 = new int[n1.length];
        for(int i=0; i<n2.length; i++){
            n2[i] = n1[i];
        }
        // 表示
        for(int n : n2){
            System.out.print(n + "%t");
        }
    }
}
```

```
3. import lib.Input;
public class P3 {
    public static void main(String[] args) {
        String[] names
            = {"鈴木", "田中", "浦", "中田", "杉山", "大野", "佐々木", "三浦"};
        String message="なし";
        String name = Input.getString();
        for(String s : names){
            if(s.equals(name)){
                message = "合致";
                break;
            }
        }
        System.out.println(message);
    }
}
```

```

4. import lib.Input;
   public class P4 {
       public static void main(String[] args) {
           double[] data = new double[100];
           int i = 0;
           double x;
           // 配列にデータを入れる
           while(i<data.length && (x=Input.getDouble())!=0){
               data[i] = x;
               i++;
           }
           // 最大値を探す
           double max;
           max = data[0];
           for(double d : data){
               if(d>max) max = d;
           }
           System.out.println("最大値="+max);
       }
   }

```

【解説】「配列にデータを入れる」では配列の容量を超えないように `i<data.length` を繰り返し条件に加えておきます。

- 5. E 【解説】 `data` には `null` が入っているので使用すると実行時例外がおこる
- 6. B 【解説】 `apple` と `banana` は同じ配列を共有している
- 7. D 【解説】 `n` は `int` の配列型、`x` は `double` の配列型。配列型は自動型変換されない

14

多次元の配列

- 1. A. `int[][] n = new int[][]{{1,2}, {3,4,5}};`
 B. `int[][] m = new int[10][];`

```

2. public class P2 {
   public static void main(String[] args) {
       int[][] n = {{1,2,3}, {4,5}, {6,7},{8}};
       for(int[] a : n){
           for(int m : a){
               System.out.print(m+"¥t");
           }
           System.out.println("");//改行
       }
   }
}

```

3.

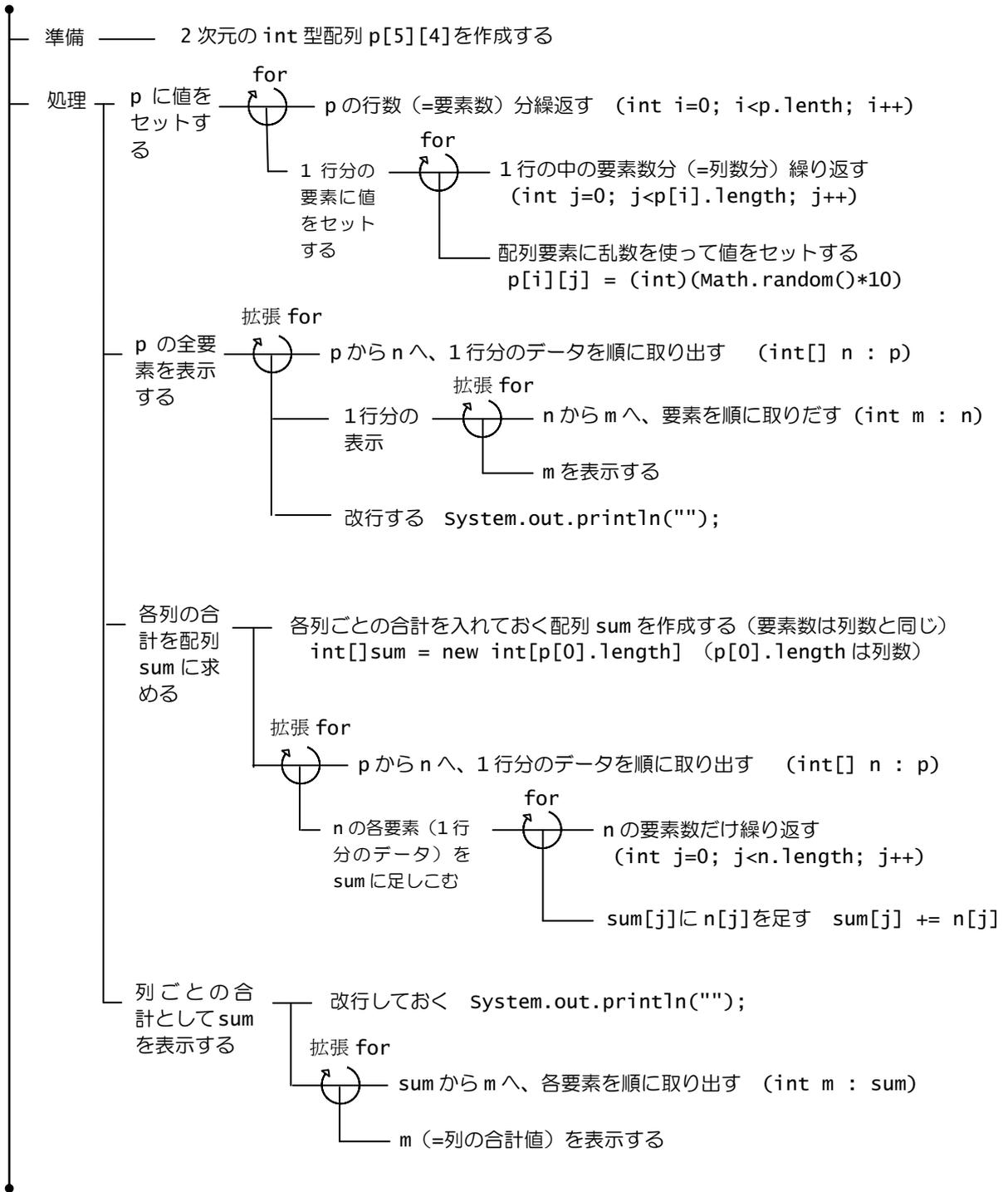
```
public class P3 {
    public static void main(String[] args) {
        int[][] n1 = { {10, 15, 22}, {8, 7, 12} };
        // コピー
        int[][] n2 = new int[n1.length][n1[0].length];
        for(int i=0; i<n2.length; i++){
            for(int j=0; j<n2[i].length; j++){
                n2[i][j] = n1[i][j];
            }
        }
        // 表示
        for(int[] m : n2){
            for(int k : m){
                System.out.print(k + "¥t");
            }
            System.out.println("");
        }
    }
}
```

4.

```
public class P4 {
    public static void main(String[] args) {
        int[][] p = new int[5][4];
        // 乱数でデータ作成
        for(int i=0; i<p.length; i++){
            for(int j=0; j<p[i].length; j++){
                p[i][j] = (int)(Math.random()*10);
            }
        }
        // データを表示
        for(int[] n : p){
            for(int m : n){
                System.out.print(m + "¥t");
            }
            System.out.println("");
        }
        // 列の合計を取る
        int[] sum= new int[p[0].length];
        for(int[] n : p){
            for(int j=0; j<n.length; j++){
                sum[j] += n[j];
            }
        }
        // 列の合計を表示
        System.out.println("");
        for(int m : sum){
            System.out.print(m + "¥t");
        }
    }
}
```

【解説】

やや複雑な処理なので、解答の SPD を次に示します。
プログラムの処理の意味を確認してください。



5. A, D

【解説】 B -- []は右側に書く、C -- int[][3]の部分は int[3][]である、E -- n と m は次元が違うので（つまり型が違う）ので参照値を代入できない、F -- new {} という書き方はない、G -- 初期化リストを与えているので int[2][]のように要素数を指定できない

6. B 【解説】 m は 2 次元配列になる。m[0] と m[1] は同じ配列を共有する

15

メソッド

- ```
import lib.Input;
public class P1 {
 public static void main(String[] args) {
 String s1 = Input.getString();
 String s2 = Input.getString();
 disp(s1, s2);
 }
 public static void disp(String s1, String s2){
 System.out.println(s1 + "は"+s2+"です");
 }
}
```
- ```
import lib.Input;
public class P2 {
    public static void main(String[] args) {
        double m = Input.getDouble("一日の予測通話分数は");
        int gaku = ryokin(m);
        System.out.println(m + "分--"+gaku+"円");
    }
    public static int ryokin(double m){
        if(m<2){
            return 1800;
        }else if(m<4){
            return 2300;
        }else if(m<8){
            return 3300;
        }else if(m<18){
            return 4800;
        }else{
            return 7300;
        }
    }
}
```

```

3. import lib.Input;
   public class P3 {
       public static void main(String[] args) {
           int n;
           while((n=Input.getInt())!=0){
               if(n<=0) continue; // 負の数は自然数ではない
               if(isSosu(n)){
                   System.out.println(n+"は素数");
               }else{
                   System.out.println(n+"は素数ではない");
               }
           }
       }
       public static boolean isSosu(int n){
           if(n==1) return false; // 1は素数ではない
           for(int i=2; i<=n/2; i++){
               if(n%i==0) return false;
           }
           return true;
       }
   }

```

4. ① char ② double b, string c ③ b、c

【解説】③ : a - double なので不可、b - char に変換される、c - char に変換される

5. B, C, E, G

【解説】仮引数の型に注目する。(cahr, int)で受け取れる実引数は A,B,D,F,H である。C や G は 152 や 0x98 のような整数を指定しているが、メソッド呼び出し時は char には自動型変換されないのが不可。ただ B は short で受けているので型変換エラーになる。結局 B,C,E,G がエラーになる

6. B 【解説】メソッド呼び出しでは、引数の 64 は short に自動型変換されない。なお、return s+k; では+は文字列連結演算子として働き、return するのは String になる

16**応用的なメソッド**

```
1. import lib.Input;

public class P1 {

    public static void main(String[] args) {
        double s1 = Input.getDouble("縦");
        double s2 = Input.getDouble("横");
        double s3 = Input.getDouble("高さ");
        double w = Input.getDouble("重量");
        if(!isOk(s1, s2, s3)){
            System.out.println("サイズオーバーで配送できない");
        }else{
            int charge = ryokin(s1,s2,s3,w);
            System.out.println("料金は" + charge + "円");
        }
    }
    // サイズチェック
    public static boolean isOk(double a, double b, double c) {
        if(a+b+c <= 180) return true;
        return false;
    }
    // 料金計算
    public static int ryokin(double a, double b, double c, double w){
        int charge;
        double size = a+b+c;
        if(size<=90){
            if(w<=5){
                charge = 500;
            }else if(w<=10){
                charge = 1000;
            }else{
                charge = 1500;
            }
        }else{
            if(w<=5){
                charge = 1000;
            }else if(w<=10){
                charge = 2000;
            }else{
                charge = 3000;
            }
        }
        return charge;
    }
}
```

2.

```
import lib.Input;
public class P2 {
    public static void main(String[] args) {
        int n = Input.getInt();
        if(isJanken(n)){
            System.out.print(n + ":");
            switch(janken(n)){//値を返すメソッド呼び出しを書いてよい
                case 0:
                    System.out.println("引き分け");
                    break;
                case 1:
                    System.out.println("勝ち");
                    break;
                case 2:
                    System.out.println("負け");
                    break;
            }
        }else{
            System.out.println("入力した値が正しくない");
        }
    }
    // 1,2,3 の乱数を返す
    public static int rand3(){
        int r = (int)(Math.random()*3+1);
        return r;
    }
    // 1,2,3 のどれかなら OK
    public static boolean isJanken(int n){
        if(n==1||n==2||n==3){
            return true;
        }else{
            return false;
        }
    }
    // 勝敗表を 2 次元の配列にしたので要素を選択するだけで勝敗が分かる
    // n と m が配列要素番号になっているところがポイント
    public static int janken(int n){
        int[][] tbl = {
            {9,9,9,9 },
            {9,0,2,1 },
            {9,1,0,2 },
            {9,2,1,0 }
        };
        int m = rand3();
        System.out.print(m + " ");
        return tbl[m][n];
    }
}
```

3.

```
public class P3 {
    public static void main(String[] args) {
        int[] dt = {5,7,9,14,7,4,2};
        graph(dt);
    }
    // グラフを描きます
    public static void graph(int[]a){
        for(int n : a){
            dot(n);
        }
    }
    // ひとつの横棒グラフを描きます
    public static void dot(int n){
        for(int i=0; i<n; i++){
            System.out.print("#");
        }
        System.out.println(""); // 改行
    }
}
```

4.

```
public class P4 {
    public static void main(String[] args) {
        double[] dt = {15.1, 1.5, 17.2, 7.3, 21.0, 3.8, 6.8};
        double[] x = Tokei(dt);
        System.out.println("合計=" + x[0]);
        System.out.println("平均=" + x[1]);
        System.out.println("標準偏差=" + x[2]);
    }
    public static double[]Tokei(double[] a){
        double[] ans= new double[3];
        ans[0] = total(a);
        ans[1] = mean(a);
        ans[2] = sd(a);
        return ans; // 配列 (の参照値) を返す
    }
    // 平均値を計算する
    public static double mean(double[] a){
        return total(a)/a.length;
    }
    // 合計を求める
    public static double total(double[]a){
        double total = 0;
        for(double d : a){
            total +=d;
        }
        return total;
    }
    // 偏差の2乗の合計
    public static double vt(double[] a){
        double mean = mean(a);
        double vt = 0;
        for(double x : a){
            vt += Math.pow(x-mean,2);
        }
        return vt;
    }
    // 標準偏差
    public static double sd(double[] a){
        double vt = vt(a);
        return Math.sqrt(vt/a.length);
    }
}
```

17 コマンドラインの操作

1. ①F ②L ③J ④G ⑤D ⑥A

2. (1) javac exam/P2.java

(2) java exam/P2 102 36

```
3. public class P3 {
    public static void main( String[] args ) {
        // コマンドラインからデータを受け取る
        if(args.length!=3){
            System.out.println("3つの引数を指定して下さい");
            System.exit(1);
        }
        double a = Double.parseDouble(args[0]); // doubleに変換
        double b = Double.parseDouble(args[1]);
        double c = Double.parseDouble(args[2]);

        //面積を計算する
        if(isvalid(a,b,c)){
            System.out.println("面積="+heron(a,b,c));
        }else{
            System.out.println("3角形ができない");
        }
    }

    // 公式を計算する
    public static double heron(double a, double b, double c){
        double s=s(a,b,c);
        return Math.sqrt(s*(s-a)*(s-b)*(s-c));
    }

    // データが正しいか調べる
    public static boolean isvalid(double a, double b, double c){
        if(a<0) return false;
        if(b<0) return false;
        if(c<0) return false;
        double s = s(a,b,c);
        if(s-a<0) return false;
        if(s-b<0) return false;
        if(s-c<0) return false;
        return true;
    }

    // 部分的な計算
    public static double s(double a, double b, double c){
        return (a+b+c)/2;
    }
}
```

18**その他の演算子**

1. 問 1

- A. `p = p > 100 ? p * 1.10 : (p > 50 ? p * 1.05 : p);`
- B. `c = a % 3 == 0 || a % 3 == 1 ? Math.sqrt(a) : Math.sqrt(a + 1);`
- C. `b = b > 0 ? b >> 1 : b >> 1;`

問 2 6 問 3 -6

2. 問 1

$A_2: 1111100000000000$ $B_2: 0000011101111010$ $C_2: 0011001100110000$

$A_{16}: 0xF000$ $B_{16}: 0x077A$ $C_{16}: 0x3330$

問 2 65536 種類 (2^{16})

3. A. `0x0A` B. `0x40` C. `0xff` D. `11010110`

【解説】 A. の問題で `0x0A+a&b` は演算子の優先順位から `(0x0A+a) &b` という計算になる

この問題は第 3 刷時点で以下のように訂正になったものです。3 刷以前の本をご覧の方は正誤表をご覧ください。

誤) A. `0x2A & a + 0x2A | b`

正) A. `0x0A + a & b`

1.2**オブジェクト指向入門**

```
public class Dice {
    int max; // サイコロの目の数の上限値
    public Dice(int m){
        max = m;
    }
    // 対戦
    public void match(){
        int man = randomNumber(); // 人のサイコロの目
        int com = randomNumber(); // コンピュータのサイコロの目
        disp(man, com); // 勝敗を表示
    }
    // 結果表示
    public void disp(int man, int com){
        System.out.println("あなた:"+man+"-- コンピュータ:"+com+" ");
        if(man>com){
            System.out.println("あなたの勝ち");
        }else if(man==com){
            System.out.println("引き分け");
        }else{
            System.out.println("コンピュータの勝ち");
        }
    }
    // 整数乱数を返す
    public int randomNumber(){
        int n = (int)(Math.random()*max) +1;
        return n;
    }
}
```

```
public class PlayDice {
    public static void main(String[] args) {
        Dice d = new Dice(6); // オブジェクト作成
        d.match();           // ゲーム実行
    }
}
```