■ 問題１：Java 言語の基礎知識

Java 言語について以下の設問に答えよ。

＜設問 1-1＞

以下の文章はクラスの継承について述べたものである。空欄に入れるべき適切な語句を**解答欄 1-1** に記入せよ。

A．Java ではクラスを継承して新しいクラスを作ることができる。このとき継承元のクラスを［ ① ］、継承先のクラスを［ ② ］と呼ぶ。［ ② ］には新たにフィールドやメソッドを追加することができる。

B．クラスを継承するにはクラス宣言で「［ ③ ］スーパークラス名」と記述する。

C．スーパークラスと同じ仕様のメソッドをサブクラスに定義することができる。これをメソッドの［ ④ ］と呼ぶ。

D．スーパークラス型の変数でサブクラスのインスタンスを指すことができる。ただし使えるのは［ ⑤ ］のメンバのみとなる。

E．スーパークラスのメソッドをサブクラスでオーバーライドすると、スーパークラス型の変数を使ってサブクラス側のメソッドを呼び出すことができるようになる。スーパークラス型の変数が指すサブクラスによって異なる動作をさせることができる。このように呼び出し方が同じでも異なる動作をすることを［ ⑥ ］と呼ぶ。

F．サブクラスでオーバーライドしたスーパークラスのメソッドを（サブクラス内から）呼び出すには［ ⑦ ］と記述する。

G．［ ⑧ ］とはプログラムコードを実装しないメソッドのことで、メソッドの宣言に［ ⑨ ］を、プログラムコードを定義する中カッコの代わりに［ ⑩ ］を付ける。

H．［ ⑪ ］を持つクラスを［ ⑫ ］と呼び、class の前に［ ⑬ ］を付ける。［ ⑫ ］型の［ ⑭ ］を作ることはできるが［ ⑮ ］を作ることはできない。

I．クラス宣言の class を［ ⑯ ］と記述するとインターフェイスになる。インターフェイスにするとすべてのメソッドが［ ⑰ ］になる。インターフェイスを実装するにはクラス宣言で「［ ⑱ ］インターフェイス名」と記述する。

J．継承できるクラスは［ ⑲ ］であるが、実装できるインターフェイスは［ ⑳ ］である。

解答欄 1-1

① スーパークラス	② サブクラス	③ extends	④ オーバーライド
⑤ スーパークラス	⑥ ポリモーフィズム	⑦ Super.メソッド名	⑧ 抽象メソッド
⑨ Abstract	⑩ セミコロン	⑪ 抽象メソッド	⑫ 抽象クラス
⑬ Abstract	⑭ 變數	⑮ インスタンス	⑯ Interface
⑰ 抽象メソッド	⑱ Implements	⑲ 一つだけ	⑳ 複數可能

＜設問 1-2＞

以下の文章は Java の様々な機能およびプログラムの設計手法について述べたものである。空欄に入れるべき適切な語句を**解答欄 1-2** に記入せよ。

A. プログラムコードのつづりの間違いや文法の誤りなどは［ ① ］と呼ぶ。これに対しプログラム実行時に発生するエラーを［ ② ］と呼ぶ。この実行時に生じるトラブルを Java では［ ③ ］と呼ぶ。［ ③ ］が発生する可能性がある場合に適切な対応をするには［ ④ ］文を使う。

B. ランタイムエラー以外にも［ ⑤ ］文を使って例外を発生させることができる。またメソッドの宣言に「［ ⑥ ］＋発生する例外」を記述すると、メソッドの呼び出し元に例外処理を投げることもできる。

C. プログラムの処理の流れの単位を［ ⑦ ］と呼び、複数の［ ⑦ ］を同時に実行するプログラムを［ ⑧ ］プログラムと呼ぶ。新しい［ ⑦ ］を追加するには［ ⑨ ］クラスを継承するか、［ ⑩ ］インターフェイスを実装する。

D. メモリ領域には［ ⑪ ］と［ ⑫ ］の 2 種類ある。［ ⑪ ］にはメソッドの中で宣言された変数のように一時的に必要な情報が格納される。［ ⑫ ］にはクラスのインスタンスが格納される。

E. Java には不要になったインスタンスを自動で削除する機能がある。これを［ ⑬ ］と呼ぶ。

F. クラスやメソッドの中でクラスを宣言することができる。これを［ ⑭ ］と呼ぶ。

G. Java ファイル（*.java）には複数のクラスを宣言することができる。ただし［ ⑮ ］修飾子をつけることができるのはひとつだけで［ ⑯ ］と同じ名前にしなければならない。

H. ［ ⑰ ］とはデータ構造（管理方法）のひとつで、先に入ったデータから先に取り出される。これに対し［ ⑱ ］は後に入ったデータから先に取り出される。

I. デザインパターンとはソフトウェアの設計ノウハウを再利用しやすいようにパターン化したものである。ひとつのインスタンスしか生成できないようにする［ ⑲ ］パターンや委譲を使ってアルゴリズムを切り換える［ ⑳ ］パターンなどがある。

解答欄 1-2

① コンパイルエラー	② ランタイムエラー	③ 例外	④ try～catch
⑤ Throw	⑥ Throws	⑦ スレッド	⑧ マルチスレッド
⑨ Thread	⑩ Runnable	⑪ スタック	⑫ ヒープ
⑬ ガベージコレクション	⑭ 内部クラス	⑮ Public	⑯ ファイル名
⑰ キュー	⑱ スタック	⑲ Singleton	⑳ Strategy

■ 問題２：Java プログラミング基礎

以下で指示された処理について**実行結果**を参考に**リスト 2** の［　A　］の部分にプログラムコードを作成せよ。

リスト 2：Java プログラミング基礎

```
public class exam21 {
    public static void main(String[] args) {
        int[] num = {50, 21, 87, 13, 91, 1234, 62, 74, 38, 45};
        A
    }
}
```

① 配列 num の最大値を表示する処理を作成せよ（配列の中をちゃんと調べること）。

実行結果①

```
最大値は 1234 です！
```

② 配列 num の各要素を 3 で割ったときのあまりを表示する処理を作成せよ（配列の中をちゃんと調べること）。なお、割り切れたときは「Fizz」と表示すること。

実行結果②

```
3 で割ったときのあまりを表示します！
あまりは 2 です！
Fizz
Fizz
あまりは 1 です！
あまりは 1 です！
あまりは 2 です！
あまりは 2 です！
あまりは 2 です！
Fizz
```

③ 配列 num の中で 3 を含む数値を表示する処理を作成せよ（配列の中をちゃんと調べること）。

実行結果③

```
3 を含む数字を表示します！
13
1234
38
```

■ 問題３：シネコン ECC

リスト 3 はシネコン ECC の発券処理である。実行したところ異なる窓口で購入すると同じ番号のチケットが発券されることがわかった（**実行結果 3：Before**）。異なる窓口で購入しても番号が重ならないように Singleton パターンを適用せよ（**クラス図 3、実行結果 3：After**）。

リスト 3：シネコン ECC の発券処理

```
public class ECCCinema {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        Window[] w = { new Window(), new Window(), new Window()};
        while(true) {
            System.out.print("何番の窓口で購入しますか？>");
            int n = Integer.parseInt(in.next());
            if (n < 0) break;
            w[n].issueTicket();
        }
    }
}
// 発券機
class TicketMaker {
    private int ticket = 0;
    public int next() { return ++ticket; }
}
// 窓口
class Window {
    private TicketMaker t = new TicketMaker();
    public void issueTicket() { System.out.println("あなたの座席番号は" + t.next() + "番です。");}
}
```

実行結果 3：Before

何番の窓口で購入しますか？>**0**
あなたの座席番号は１番です。
何番の窓口で購入しますか？>**0**
あなたの座席番号は２番です。
何番の窓口で購入しますか？>**1**
あなたの座席番号は１番です。
何番の窓口で購入しますか？>**2**
あなたの座席番号は１番です。
何番の窓口で購入しますか？>**-1**

実行結果 3：After

何番の窓口で購入しますか？>**0**
あなたの座席番号は１番です。
何番の窓口で購入しますか？>**0**
あなたの座席番号は２番です。
何番の窓口で購入しますか？>**1**
あなたの座席番号は１番です。
何番の窓口で購入しますか？>**2**
あなたの座席番号は１番です。
何番の窓口で購入しますか？>**-1**

クラス図 3