Ｊａｖａプログラミング Ⅱ

８回目　抽象クラスとインタフェース

抽象クラス

抽象メソッドとは　処理内容が定義されない空のメソッド

宣　　言　　メソッドの修飾子に *abstract* を加える
メソッドの処理内容は書かない
※メソッドの本体部はブロック”{}”ではなくセミコロン”;”を書く

abstract　戻り値の型　ﾒｿｯﾄﾞ名(引数ﾘｽﾄ);

抽象クラスとは　０または１個以上の抽象メソッドをメンバーにもつクラス

宣　　言　　クラスの修飾子に *abstract* を加える
１個以上の抽象メソッドをメンバーにもつ場合
➔ 必ず *abstract* を記述
※クラスが抽象メソッドを含むことを明示
０個の抽象メソッドをメンバーにもつ場合
➔ 修飾子 *abstract* により抽象メソッドをもたない抽象クラスとなる

```
abstract　class　ｸﾗｽ名{
    メンバー
}
```

特　　徴　　抽象クラスのオブジェクトを直接は作成できない

利　　用　　継承してすべての抽象メソッドをオーバーライドする
オーバーライドされない場合 ➔ サブクラスは抽象クラスとなる

抽象クラス型の変数や配列としての利用は可能

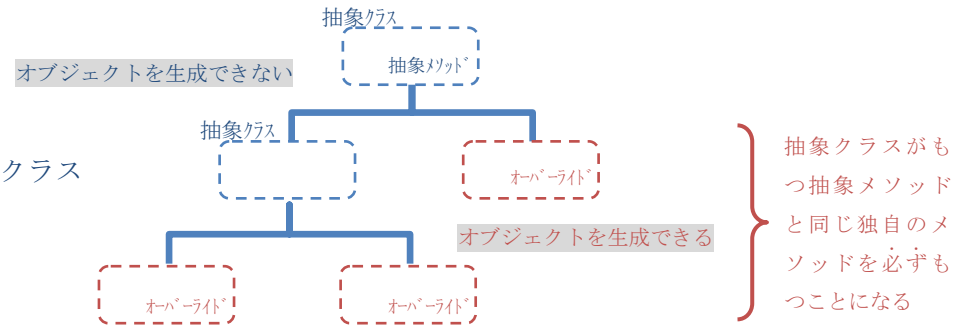

図１　抽象クラス

instanceof 演算子

instanceof 演算子　オブジェクトのクラスを調べる

書　　式　　参照　*instanceof*　クラス名

演算結果
true　←　左辺のオブジェクトのクラスが
　　右辺のクラス、またはそのサブクラス
false　←　それ以外

インタフェース

インタフェースとは　抽象メソッド、定数をメンバーにもつ型（インタフェース型という）
※その他、クラスやインタフェースをメンバーにもつことができる

宣　　言　　キーワード interface を用いてメンバーを指定する

interface インタフェース名{
　メンバー
}

※インタフェースとそのメンバーは暗黙的に次の修飾子となる
　interface　→　*abstract*
　メソッド　→　*public abstract*
　変　　数　→　*public static final*

※変数はすべて定数のためコンストラクタの必要はない
　インタフェース → コンストラクタの宣言はできない
　抽象クラス → コンストラクタの宣言はできる

特　　徴　　インタフェース型のオブジェクトを直接は作成できない

利　　用　　実装してすべての抽象メソッドをオーバーライドする
オーバーライドされない場合→実装されたクラスは抽象クラスとなる

インタフェース型の変数や配列としての利用は可能

定数は インタフェース名.変数名 でアクセス
※クラス変数と同じアクセス方法

インタフェースの実装

インタフェースの実装とは　クラスと組み合わせること
クラスはインタフェースがもつメンバーを受け継ぐ

宣　　言　　キーワード implements を用いてインタフェースを指定する

class クラス名 *implements* インタフェース 1, インタフェース 2, ・・・{
　　メンバー
}

インタフェースは多重継承の仕組みを実現する
同じ抽象メソッド名がある場合は１つとみなされる
・クラスの継承　　→　単一継承
・インタフェースの実装 →　多重継承

インタフェースの拡張

インタフェースは、クラスと同じように継承による拡張が可能

宣　　言　　*interface* サブインタフェース *extends* スーパーインタフェース 1, スーパーインタフェース 2, ・・{
　　メンバー
}

インタフェースは多重継承の仕組みを実現する
同じ抽象メソッド名がある場合は１つとみなされる

抽象メソッドのオーバーロード／オーバーライドは同様になされる