
MPC
Club
rogramming
uroran

今日やること

1. カプセル化
2. 継承
3. オーバーライド

今日やること

1.カプセル化

2.継承

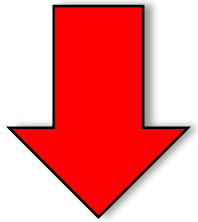
3.オーバーライド

カプセル化

kaimai.hp ＝ -1;

ありえない値を設定
できてしまった！！

フィールドを直接操作してほしくない



カプセル化

カプセル化とは?

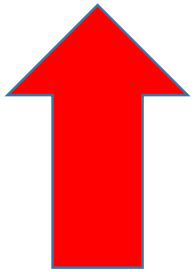
隠蔽

- getter：値を取得するメソッド
- setter：値を設定するメソッド

フィールドを守る!!

メソッドを使うことでありえない
値を設定されること防げた？？

kaimai.hp ＝ -1;



まだ参照できる・・・

アクセス修飾子

アクセス修飾子

private	同じクラスからアクセス可能
なし	同じパッケージからアクセス可能
protected	同じパッケージ・継承先のクラスからアクセス可能
public	全クラスからアクセス可能

private int hp;

カプセル化完了!!

今日やること

1. カプセル化
2. 継承
3. オーバーライド

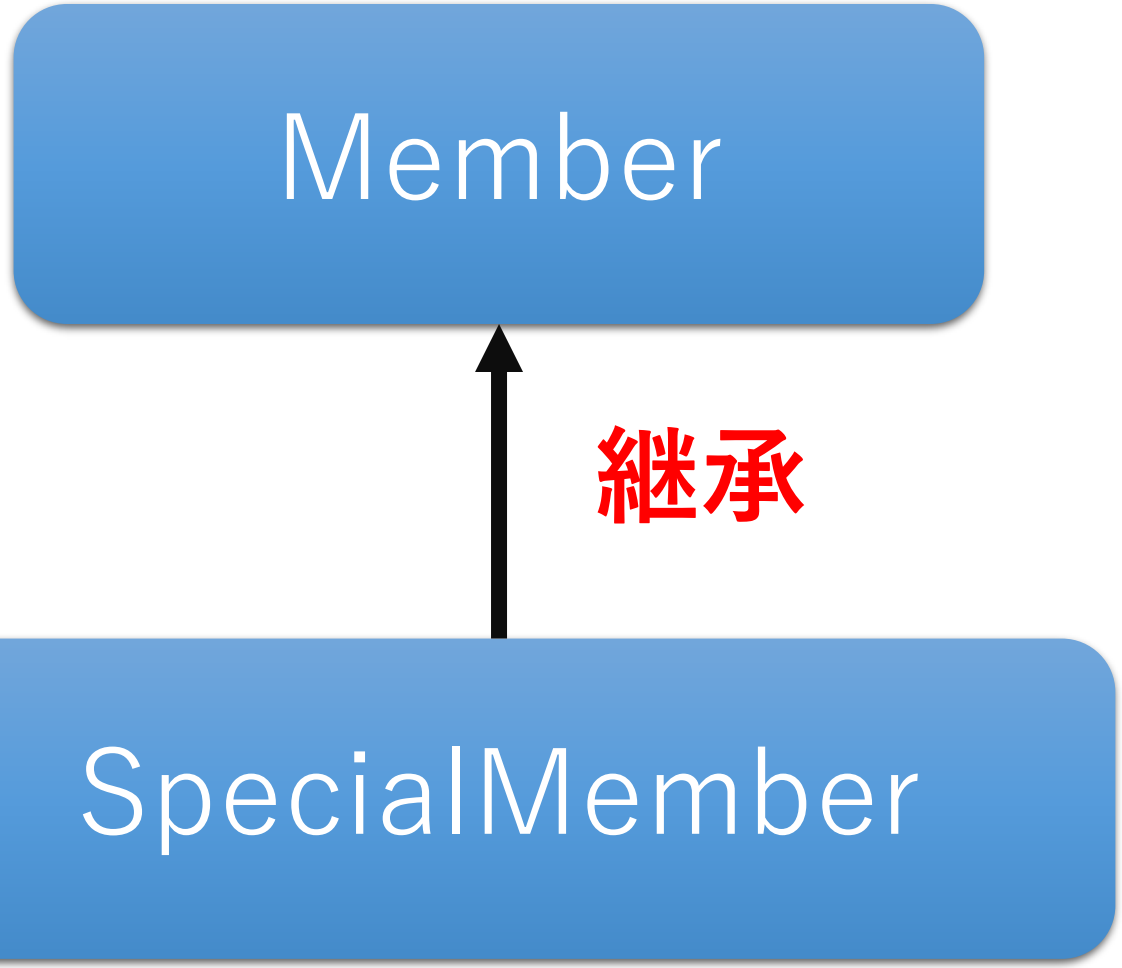
継承

継承とは?

2つのクラスに親子関係を持たせる

子は親のメンバを引き継ぐことができる

親クラス



子クラス

継承

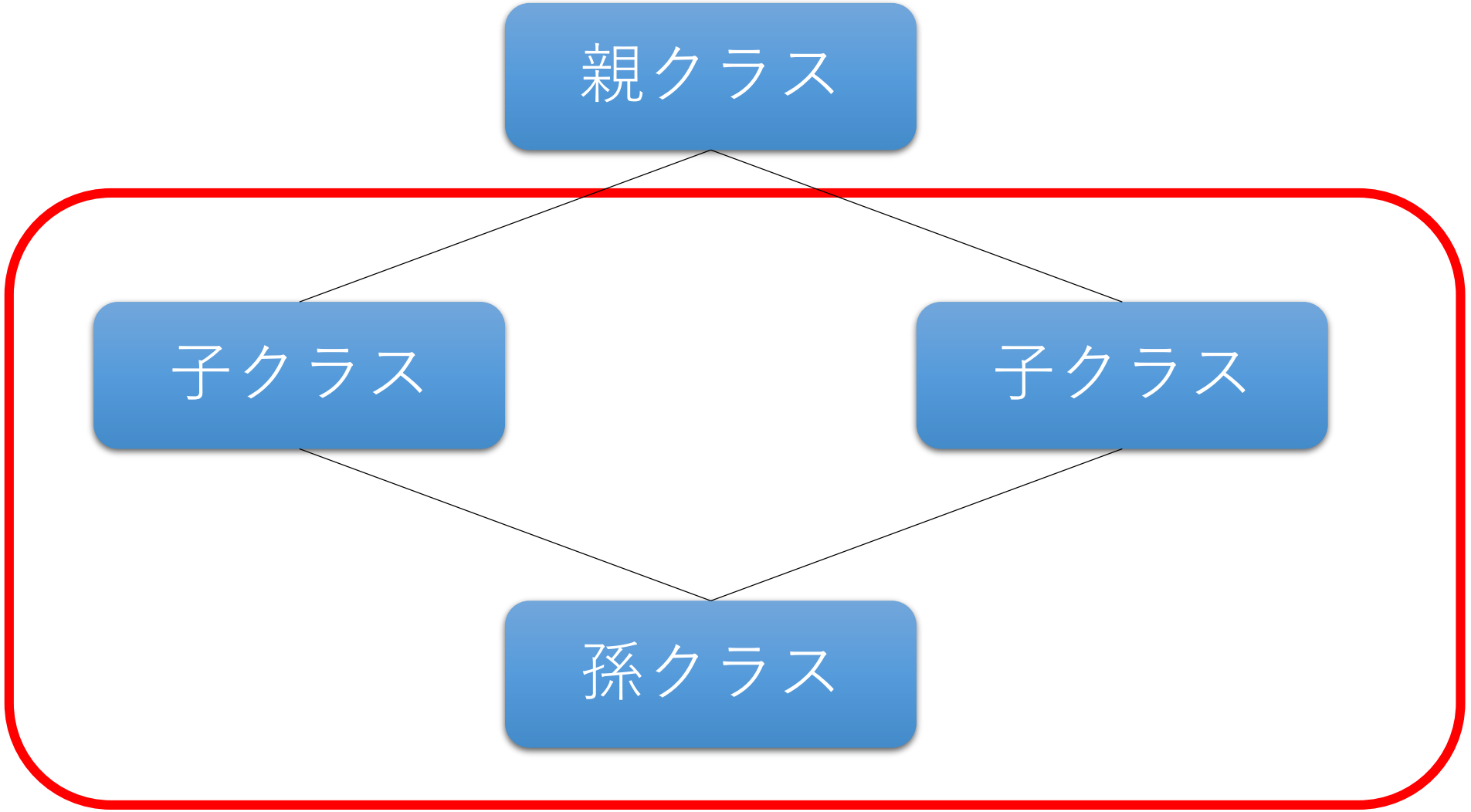
class　子クラス　**extends**　親クラス{…}

アクセス修飾子

private	同じクラスからアクセス可能
なし	同じパッケージからアクセス可能
protected	同じパッケージ・継承先のクラスからアクセス可能
public	全クラスからアクセス可能

多重継承の禁止

多重継承



多重継承

今日やること

1. カプセル化
2. 継承
3. オーバーライド

オーバーライド

オーバーライドとは?

子クラスで親クラスと同名の
関数を定義できる

子クラスで処理を上書き
(**オーバーライド**)できる

オーバーライド

親クラスのメソッドをオーバーライドしなかった場合

```
親クラス.method();　　//親クラスの処理
子クラス.method();　　//親クラスの処理
```

親クラスのメソッドをオーバーライドした場合

```
親クラス.method();　　//親クラスの処理
子クラス.method();　　//子クラスの処理
```

演習問題