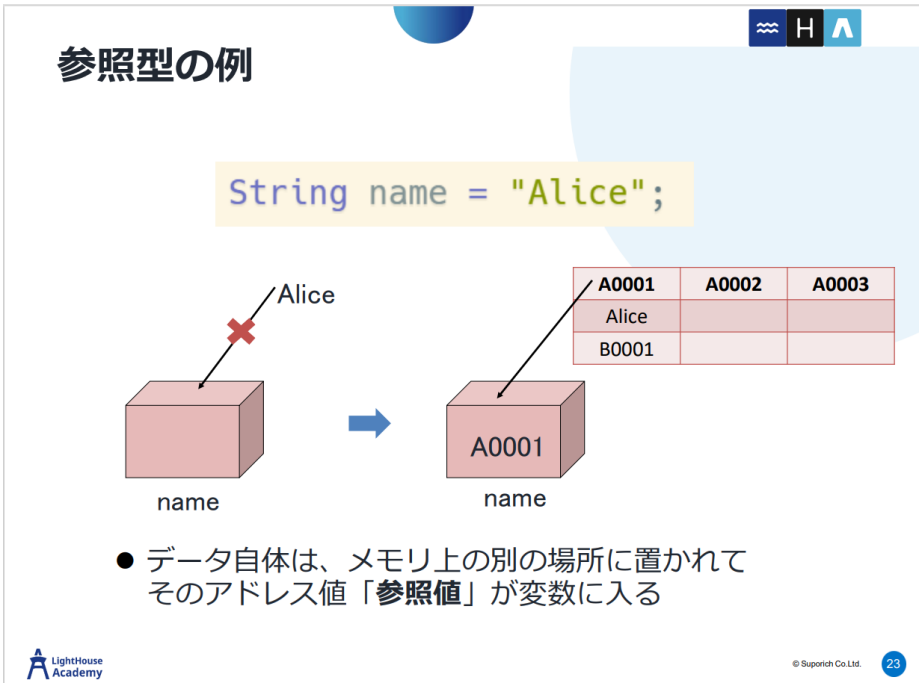
基本データ型・参照型　ok
アクセス修飾子（カプセル化）ok
オーバーロード・オーバーライド(アノテーション)ok
オブジェクト指向（メリット）
インスタンス（フィールド・メンバ変数・コンストラクタ・メソッド）
This/super
継承　is-a
ポリモーフィズム
インタフェース
抽象クラス
Try-catch
リストと配列の違い・連想配列
API
ラムダ式

基本データ型・参照型
変数定義などで用いる型は大きく分け 2 種類存在し、それぞれ基本データ型と参照型と呼ばれる。

基本データ型
数値のような基本的なデータを格納する型。変数そのものに格納される

参照型
文字列や配列、自分で定義したクラス等のデータタイプ。基本データ型と違い変数そのものではなくメモリにデータが格納され、変数にはデータが格納されている番地（アドレス）が格納される。※String も厳密にいえばクラス



カプセル化
変数定義やメソッド定義を行う際、閲覧・編集できるクラスを限定したい状況などがある。
その場合に用いるのがカプセル化である。

カプセル化はアクセス修飾子をアクセス制限したいものの頭につけることで実装することができる

アクセス修飾子の種類と範囲

修飾子	本クラス	本パッケージ	スブクラス	外部パッケージ
public	○	○	○	○
protected	○	○	○	×
(default)	○	○	×	×
private	○	×	×	×

※上記の図では protected は外部パッケージからは参照できないと書かれているが、外部パッケージのクラスが protected を使用しているクラスを継承するサブクラスだった場合のみ参照することが可能

オーバーロードとオーバーライド
この二つは名前も似ておりどちらもメソッドに作用するものなので注意が必要。

二つの違い

オーバーロード
オーバーロードは、既に存在するメソッドのパラメータを変更することができる。
この際、戻り値・メソッド名などはオーバーロードを行う対象と全く同じでなければならない。

オーバーライド
オーバーライドは継承などを行った際に、スーパークラスのメソッドをサブクラスにあった形式で使用したい場合などに用いるものであり、メソッドの内容を変更し使用することができる。
※マナーとしてアノテーションを必ずつける

二つともアクセス修飾子は条件の緩いほうにのみ変更することができる。

オブジェクト指向
様々なクラスから、役割を持ったオブジェクトを呼び出し実行していくコードの書き方。

行うメリット
継承やインスタンス化など様々な方法でデータの受け渡しを行うことで、一つのクラスにコードが集中せずスッキリとしたコードになる。