この講義の目的

モデリングとシミュレーション

2020年度

私たちはいつもモデル化を行っている

- 心のモデル
  - 生まれてすぐに、身近な大人を見ながらモデル化を開始
  - 他人の行動、心理を読もうとする

私たちはいつもモデル化を行っている

- 周囲のモデル化
  - 物の動きの予想
  - 社会の動向の予想
- 現実と常に調整を実行

科学とは

- 対象物を記述する
  - 分類学、博物学
- 現象を記述する
  - 現象学
- 対象物、現象を一般化する
- 常に、現象を参照することが重要

厳密な科学へ

- 対象を要素に分解し、それらの相互作用として現象や対象物を記述する
- 一般原理から、現象を説明する

- 具体から一般へ、一般から具体へ
- 文章による記述から、数理モデルへの発展
  - 一般性、厳密性

20世紀末からの科学
コンピュータ・ネットワークの普及

- 計算モデル→シミュレーション
  - 今ならば、PCで十分
- 様々な分野でのモデル化
  - 分野横断的研究
- 重要なスキル
  - 好奇心
  - 数学、物理学、統計学、プログラミング

(数理)モデル化の要点

- 注目する対象、量の切り出し
  - 名詞に注目
- 対象の動作・操作の切り出し
  - 動詞に注目
- 適切な名前付け
- 関係の書き出し
  - 階層構造、包含関係、相互作用

この講義の目標

- 微分方程式や線形代数を利用して、モデル化し、それを解く。
- 確率的現象を理解し、確率を利用したシミュレーションを行う。
- シミュレーション結果を可視化し、結果を説明する。

この講義と「実験」との関係

- この講義では
  - モデルの提示
  - 理論的解析
- 「実験」では
  - シミュレーションの実行
  - データ解析
- 単位認定は連動
- 今年度が最後の開講

Java入門

モデリングとシミュレーション

10

2020年度

オブジェクト指向
Object Oriented Programming

- 対象をモノ(object)と、その運動・変化として捉える
- 対象システムを記述
  - 名詞をクラスとして捉える
  - 動詞をオブジェクトの操作、動作として捉える

クラス(class)とは

- クラス：モノを抽象化したもの
  - 普通名詞相当
  - 属性：数値や文字列など
  - メソッド：操作

メソッド(method)とは

- モノ(クラス)の動き
  - モノの操作
  - モノの変化
- メソッドとして記述
- 名詞に付随した述語

クラスインスタンス(instance)とは

- クラスの具体化したもの
  - クラス⇔クラスインスタンス
  - 一般名詞⇔固有名詞
- instance : a particular example or case of something

インスタンスに属さない一般化された述語

- 副作用を起こさないモノ
- 数学的関数
  - 引数の値に応じて値を返す
- サブルーチン
  - 引数の値を加工する
- 述語を集めたクラス(ライブラリ)のスタティックメソッド

クラスとインスタンスの例

- 学生を学籍番号と氏名で登録する
  - 学生というクラス
  - 属性としての学籍番号と氏名
- 具体的な学生がクラスインスタンス
  - 具体的に学籍番号と氏名が設定される

StudentSampleクラス

```
1.  package studentSample0;
2.  public class Student {
3.      public final String name;//氏名
4.      public final int studentID;//学籍番号


5.      //コンストラクタ、クラスインスタンスを生成する
6.      public Student (String name, int StudentID){
7.          //thisはコンストラクタで生成された「この」インスタンス
8.          this.name = name;
9.          this.studentID = StudentID;
10.     }
11. }
```

main()から起動

```
1.  package studentSample0;
2.  public class Main {
3.      //ここから実行が始まる
4.      public static void main(String[] args) {
5.          // Studentクラスインスタンスの生成
6.          //ここでは、クラスインスタンスの配列を生成
7.          Student studentArray []= {new Student("Aoyama",1),
8.                                    new Student("Asou",2) };
9.          for (Student student : studentArray) {
10.             System.out.println(student.name);    }
11.     }
12.}
```

Javaの特徴

- 文法はほぼC++と同じ
  - ポインタが無い
- すべてクラスで記述する
  - クラスインスタンスは参照型

Javaの特徴

- mainメソッドを持つクラスから起動
- OS非依存
  - “Write once, run anywhere”

- mainメソッドを有するクラスを起動する
- mainメソッドの役割
  - クラスインスタンスを生成して起動する
  - ここに、処理の本体を書かないこと
  - 修飾子staticの意味を考える

良いプログラムに向けて

- 一貫した名前の付け方：後述
- 適切なモジュール、クラスへの分割
- メソッドは短く
  - 一つのメソッドに一つの機能

Javaの命名規則

- 読みやすいプログラムを書くための重要な要素
  - 標準的な命名規則に従う
- クラス名は大文字で始める
- 変数名、インスタンス名、メソッド名は小文字で始める
  - 例外：定数は大文字
- パッケージ名は小文字で始める

Javaの命名規則２

- クラスのインスタンスは、クラス名の最初を小文字にすると分かりやすい
  - 例： Student student;
- 一文字の変数名はごく局所的に
- Camel記法
  - 複数の英単語を結ぶ名称の場合、単語の最初を大文字としてスペースを入れずにつなぐ
  - 例：StudentRecord

Javaの命名規則３ setterとgetter

- クラスフィールドへのアクセスメソッドの標準的命名規則
- 値の設定
  - setフィールド名
- 値の取得
  - getフィールド名
- Netbeansの機能に従う