

情報処理技法 (Javaプログラミング) I

第1回
人がコンピュータに命令するには?
(プログラムの基本原理と書き方、実行方法)
人間科学科コミュニケーション専攻
白銀 純子

Copyright (C) Junko Shimomura, Tokyo Woman's Christian University 2015. All rights reserved.

第1回の内容

- ✧ オリエンテーション
- ✧ ファイルシステムの復習
- ✧ ターミナルの使い方
- ✧ プログラミングの概略

Copyright (C) Junko Shimomura, Tokyo Woman's Christian University 2015. All rights reserved.

授業目標

- ✧ 人がコンピュータに命令をし、コンピュータが動作する際の基本原理を理解すること
- ✧ コンピュータに命令をする際の基本的な文法を理解すること
- ✧ 自分で命令を組み立てることができるようになること

Copyright (C) Junko Shimomura, Tokyo Woman's Christian University 2015. All rights reserved.

プログラミングを習得するためには?

- ✧ 自分で試行錯誤すること
 1. 教科書などに載っているプログラムを写して実行してみる
 2. そのプログラムの一部を変更して実行してみる
- この繰り返しが一番重要&効果的!!

Copyright (C) Junko Shimomura, Tokyo Woman's Christian University 2015. All rights reserved.

学習上の注意事項

- ✧ 講義中だけでなく、手を動かすこと
 - ✧ 講義内容を次の講義までに復習し、練習問題を必ずやっておくこと
- ✧ 疑問点やわからないことをそのままにしないこと
 - ✧ 必ず次の講義までに解決するように!
- ✧ 授業を休んだときは、次に授業までに、必ず授業のWebページを見て内容を勉強しておくこと
 - ✧ 特にプログラミングの授業は、1回休むと全くついていけないこともあるので要注意!
 - ✧ わからないことは聞くこと

Copyright (C) Junko Shimomura, Tokyo Woman's Christian University 2015. All rights reserved.

教科書・連絡先・資料置き場

- ✧ 教科書
 - ✧ 基礎講座 Java, 白銀純子, 毎日コミュニケーションズ, 2010
- ✧ 連絡先
 - 研究室: 8号館4階8413室
 - メールアドレス: junko@lab.twcu.ac.jp
 - ※ 質問は、メールが研究室にどうぞ
- ✧ 授業Webページ
 - <http://www.cis.twcu.ac.jp/~junko/Programming/>
 - ✧ 講義中に見せる資料
 - ✧ 授業内容とその補足を載せたページ

Copyright (C) Junko Shimomura, Tokyo Woman's Christian University 2015. All rights reserved.

成績評価とレポート

- ⊗ 成績評価
 - 出席: 30%, レポート+期末試験: 70%
- ⊗ 出席
 - ⊗ 前回授業の復習問題への解答で出席としてカウント
 - ⊗ 正当な理由での欠席の場合、何らかの証明ができるものがあれば、出席を考慮
- ⊗ レポート
- ⊗ 期末試験(持ち込み全て可で、実技試験がメイン)

Copyright (C) Jinko Shimomura, Tokyo Woman's Christian University 2015. All rights reserved.

レポート・試験の採点基準

- ⊗ 提出されたプログラム1つ1つについて...
 1. コンパイル・実行結果が正しい: A相当の評価
 2. 1. に加え、細やかな配慮ができている: S相当の評価
 - ⊗ 必要ところで必要な情報提供をしている(入力・出力時のガイドをしている, etc.)
 - ⊗ エラーになりそうな入力に対する処理をしている
 - ⊗ etc.

Copyright (C) Jinko Shimomura, Tokyo Woman's Christian University 2015. All rights reserved.

ファイルシステムの復習

Copyright (C) Jinko Shimomura, Tokyo Woman's Christian University 2015. All rights reserved.

ファイルとフォルダ

- ⊗ ファイル: 文書や絵などを書いた紙
 - ⊗ 1つ1つのファイルは名前を付けて区別
- ⊗ フォルダ: ファイルを整理する箱



Copyright (C) Jinko Shimomura, Tokyo Woman's Christian University 2015. All rights reserved.

拡張子のおはなし

- ⊗ 拡張子: ファイル名の最後の「.」以降の部分

abc.txt
拡張子

def.html
拡張子

ghi.png
拡張子
- ⊗ 拡張子が何であるかで、ファイルの種類を区別

txt: テキストファイル	html: Webページのファイル
docx: Microsoft Wordファイル	png, jpg, gif: 静止画のファイル

Javaプログラムファイルは? ➡ java

Copyright (C) Jinko Shimomura, Tokyo Woman's Christian University 2015. All rights reserved.

ターミナルの使い方

Copyright (C) Jinko Shimomura, Tokyo Woman's Christian University 2015. All rights reserved.

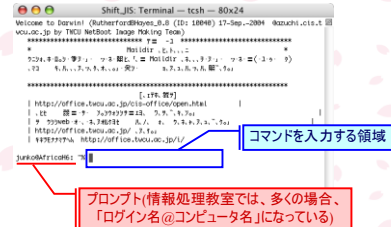
「ターミナル」って何?

- ⊛ ソフトウェアの名前(+α)を入力することで、ソフトウェアを使うための道具
- ⊛ 普通、ソフトウェアを使うときには、そのソフトウェアのアイコンをダブルクリックすると、ソフトウェアが起動
- ⊛ ターミナルでは、ソフトウェアの名前(+α)を入力し、「Return」キーを押すと、ソフトウェアが起動
- ⊛ Javaプログラミング1で使うターミナル: 「Finder」→「アプリケーション」→「ユーティリティ」フォルダを開き、その中の「ターミナル」をダブルクリック

Copyright (C) Jyuku Shingane, Tokyo Woman's Christian University 2015. All rights reserved.

コマンド入力の基本(1)

- ⊛ コマンドは、「プロンプト」の後ろに半角英数で入力



Copyright (C) Jyuku Shingane, Tokyo Woman's Christian University 2015. All rights reserved.

コマンド入力の基本(2)

- ⊛ コマンドの形

- 「コマンド名」がソフトウェアの名前に相当
- コマンド名 引数
- 必ず「コマンド名」を最初に入力し、その後に「引数」を入力
 - 「コマンド名」と「引数」の間にはスペースが1つ以上必要
 - 「引数」は1つとは限らない
 - 「引数」が複数ある場合には、引数と引数の間にもスペースが1つ以上必要
- 例えば、コマンド名「ls」、引数「WWW」の場合:
「ls WWW」と入力

- ⊛ プロンプトは、「%」や「\$」と略して書かれることも

- 「% abc」とかかっている場合には、「%」の後から入力すること(「%」は入力しない!)

Copyright (C) Jyuku Shingane, Tokyo Woman's Christian University 2015. All rights reserved.

コマンド入力の基本(3)

- ⊛ ターミナルは寡黙

- 入力したコマンドが、成功して終わったとき:
- 何も言わずにプロンプトを表示する
- 入力したコマンドが、失敗したとき:
- エラーメッセージを表示してプロンプトを表示する
- 利用者に何か聞きたいとき
- メッセージを表示してプロンプトを表示しない
- コマンドの実行に時間がかかるとき:
- プロンプトを表示しない
- ※自分でプロンプトを書いたりしないこと
(プロンプトを自分で書いても、何も起こらない)

Copyright (C) Jyuku Shingane, Tokyo Woman's Christian University 2015. All rights reserved.

プログラミング

Copyright (C) Jyuku Shingane, Tokyo Woman's Christian University 2015. All rights reserved.

外国人に手紙を書く場合どうする?(p. 16)

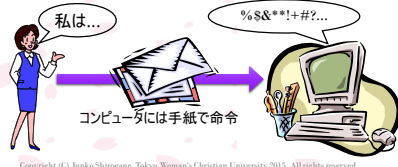
- ⊛ 相手がわかる言葉で手紙を書く
- ⊛ 相手が理解できる言葉を感じるのは大変!!
- ⊛ コンピュータには、手紙(命令書)で命令
- ⊛ コンピュータが理解できる言葉で手紙(命令書)を書く



Copyright (C) Jyuku Shingane, Tokyo Woman's Christian University 2015. All rights reserved.

外国人に手紙を書く場合どうする?(p. 16)

- ✳️ 相手がわかる言葉で手紙を書く
 - ✳️ 相手が理解できる言葉覚えるのは大変!!
- ✳️ コンピュータには、手紙(命令書)で命令
 - ✳️ コンピュータが理解できる言葉で手紙(命令書)を書く



コンピュータが理解できる言葉は?(p. 16)

- ✳️ コンピュータが理解できる言葉: **機械語**
 - ✳️ コンピュータは、**0と1の羅列**しか理解できない
 - コンピュータの中にたくさんのスイッチがあり、どのスイッチがONでどのスイッチがOFFになっているかの組み合わせ
 - = 命令書は、機械語(0と1だけで書かれたもの)になっている必要

- ✳️ 人間が理解するのは大変
- ✳️ 命令書を人間が理解できる言葉で書き、それを訳したものをコンピュータに渡す



手紙を訳すには?(p. 16)

- ✳️ 手紙を翻訳する



- ✳️ 手紙を通訳する



コンピュータへの命令書も同じ!

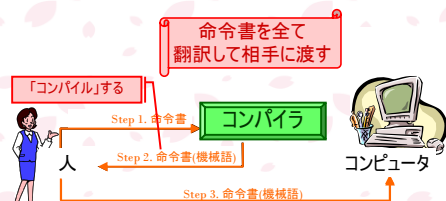
命令書を翻訳(p. 19)

- ✳️ **コンパイラ**: 命令書を最初から最後まで機械語に翻訳するためのソフトウェア



命令書を翻訳(p. 19)

- ✳️ **コンパイラ**: 命令書を最初から最後まで機械語に翻訳するためのソフトウェア



命令書を通訳(p. 20)

- ✳️ **インタプリタ**: 命令書を最初から1行ずつ読んで機械語に通訳するためのソフトウェア



命令書を通訳(p. 20)

- ⊛ **インタプリタ**: 命令書を最初から1行ずつ読んで機械語に通訳するためのソフトウェア



Copyright (C) Junko Shimomura, Tokyo Woman's Christian University 2015. All rights reserved.

命令書って何語で書くの??(p. 17)

- ⊛ コンパイラ & インタプリタ

- ⊛ 人間が話す言葉は理解できない

~~日本語、英語、ドイツ語、
中国語、韓国語、フランス語...~~

⊛ 同じ文言でも複数通りの解釈が存在(あいまい)

プログラミング言語

Copyright (C) Junko Shimomura, Tokyo Woman's Christian University 2015. All rights reserved.

プログラミング言語とは?(p. 17)

- ⊛ コンピュータに命令を伝えるための言語
- ⊛ 誰がいつ解釈しても意味が同じ

私、ハンバーグ! ⊛ 私はハンバーグを作る?(「料理中」)
私はハンバーグを食べる?(「レストラン」)
コンパイラ・インタプリタは状況判断ができない

- ⊛ 文法規則を厳密に定義

白い花模様の服 ⊛ 花模様が白い?(「花模様」にかかる)
服が白い?(「服」にかかる)
「白い」がどちらにかかるか厳密に定義する必要

Copyright (C) Junko Shimomura, Tokyo Woman's Christian University 2015. All rights reserved.

用語(p. 17)

- ⊛ 手紙(命令書)
= ソースコード(プログラム)
- ⊛ ソースコードを作成すること
= プログラミング
- ⊛ 実行可能プログラム
= コンピュータが直接実行可能な命令書
⊛ 機械語に翻訳された命令書
- ⊛ プログラム
⊛ 「ソースコード」の意味と、「実行可能プログラム」の意味と、どちらもで利用

Copyright (C) Junko Shimomura, Tokyo Woman's Christian University 2015. All rights reserved.

コンパイラとインタプリタの利点・欠点(p. 22)

- ⊛ コンパイラ = 命令書を一度に全て翻訳

- ⊛ 利点:

- ⊛ コンピュータが命令書を実行する速度が速い
- ⊛ 一度翻訳すれば何度でも実行できる

- ⊛ 欠点:

- ⊛ 命令書に文法的な間違いがあると、実行できない

- ⊛ インタプリタ = 命令書を一行ずつ通訳

- ⊛ 利点:

- ⊛ 命令書に間違いがあっても途中で実行できる

- ⊛ 欠点:

- ⊛ コンピュータが命令書を実行する速度が遅い
- ⊛ 実行のたびにインタプリタが通訳する

Copyright (C) Junko Shimomura, Tokyo Woman's Christian University 2015. All rights reserved.

Javaって何?(p. 26)

- ⊛ プログラミング言語の1つ

- ⊛ プログラミング言語の種類はたくさんあり

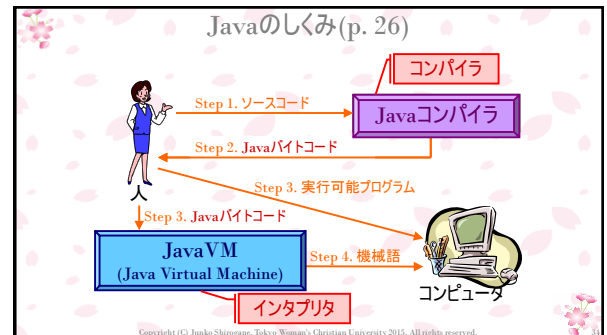
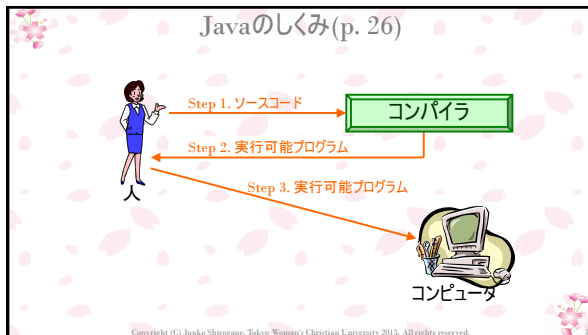
- ⊛ それぞれのプログラミング言語で得手不得手あり

- ⊛ **コンピュータやOSに依存せず** 実行可能

実行可能プログラムは通常、OSが異なると実行できない

※OS: オペレーティングシステム(基本ソフト、WindowsやMacOSなど)

Copyright (C) Junko Shimomura, Tokyo Woman's Christian University 2015. All rights reserved.



Javaプログラムの実行方法

- ⊛ Step 0: ターミナルの**カレントフォルダ**を、Javaファイルの保存場所に合わせる
 - ⊛ この作業は、コマンドプロンプトを起動したときに1度だけ行う
- ⊛ Step1: Jeditなどでソースコードを作成する
 - ⊛ ファイル名は、必ず拡張子を「**java**」とすること

Copyright (C) Jinko Shimono, Tokyo Woman's Christian University 2015. All rights reserved.

Javaプログラムの実行方法

- ⊛ Step2: ソースコードをコンパイルする
(コマンド名: **javac**, 引数: ソースコードのファイル名)


```
% javac ファイル名.java
```

➡ 「ファイル名.class」というファイルが作成される
- ⊛ Step3: JavaバイトコードをJavaVMで実行する
(コマンド名: **java**, 引数: 拡張子なしのファイル名)


```
% java ファイル名
```

拡張子は不要

Copyright (C) Jinko Shimono, Tokyo Woman's Christian University 2015. All rights reserved.

Step 0: カレントフォルダ

- ⊛ カレントフォルダ: ターミナルでの、現在の作業フォルダ
- ⊛ Javaプログラムのコンパイルをするには...
 - ⊛ コマンドプロンプトのカレントフォルダを、Javaプログラムを保存してあるフォルダに設定する必要
- ⊛ カレントフォルダを設定するには...
 - ⊛ Javaプログラムを保存したフォルダの「**パス**」を考える必要

Copyright (C) Jinko Shimono, Tokyo Woman's Christian University 2015. All rights reserved.

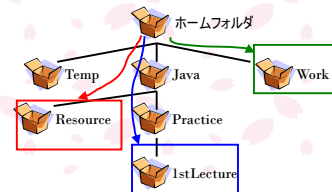
Step 0: パス(1)

- パス: ファイルやフォルダのありかを表す文字の並び
- どのようにフォルダをたどれば、目的のファイルやフォルダにたどり着くかを表すもの
- 「絶対パス」と「相対パス」に分類
- 「A/B」*で、「A」というフォルダの中に「B」というフォルダまたはファイルが入っている、という意味
 - Ex. 「Java/Practice/1stLecture」で、「Java」フォルダの中の「Practice」フォルダの中の「1stLecture」という意味
 - *Windowsでは「A\B」と表す
- 絶対パス: 最上位のフォルダから目的のファイルやフォルダへのパス
- 相対パス: 最上位以外のフォルダからのパス

Copyright (C) Junko Shimomura, Tokyo Woman's Christian University 2015. All rights reserved.

Step 0: 相対パス

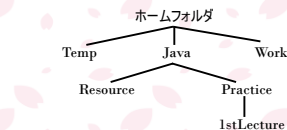
- ここでは、ホームフォルダからのパスを考えてみる



Copyright (C) Junko Shimomura, Tokyo Woman's Christian University 2015. All rights reserved.

Step 0: ホームからの相対パスの考え方(1)

- ホームフォルダから、目的のフォルダ・ファイルへの経路を「→」を使って書く
 - 経路: 「コンピュータ」から、どのようにフォルダをダブルクリックして開いていけば、目的のファイルやフォルダが見つかるか



Ex1. 「1stLecture」までの経路: **Java → Practice → 1stLecture**
 Ex2. 「Resource」までの経路: **Java → Resource**
 Ex3. 「Work」までの経路: **Work**

Copyright (C) Junko Shimomura, Tokyo Woman's Christian University 2015. All rights reserved.

Step 0: ホームからの相対パスの考え方(2)

- 「→」を「/」で置き換える

絶対パス

Ex1. 「1stLecture」までの経路:
 Java → Practice → 1stLecture
 ➡ **Java/Practice/1stLecture**

Ex2. 「Resource」までの経路:
 Java → Resource
 ➡ **Java/Resource**

Ex3. 「Work」までの経路: Work
 ➡ **Work**

Copyright (C) Junko Shimomura, Tokyo Woman's Christian University 2015. All rights reserved.

Step 0: カレントフォルダの設定

- カレントフォルダの変更のコマンド:
 - コマンドの入力
 - % cd **ホームフォルダからの相対パス**
 - Ex1. ホームフォルダの中で、「Desktop」→「Java」→「chap」に保存してある場合 (相対パス: Desktop/Java/chap):
 - % cd **Desktop/Java/chap**
 - Ex2. ホームフォルダの中で、「Download」→「chap」→「chap01」に保存してある場合 (相対パス: Download/chap/chap01):
 - % cd **Download/chap/chap01**

Copyright (C) Junko Shimomura, Tokyo Woman's Christian University 2015. All rights reserved.

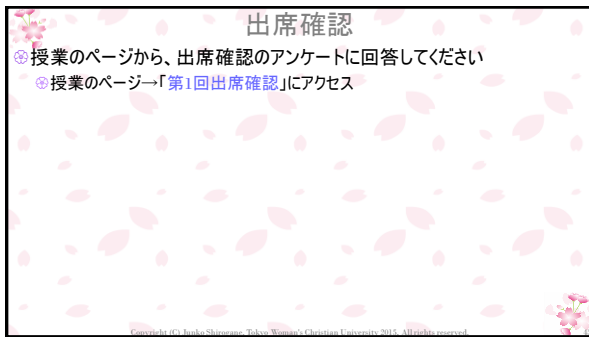
ちょっとやってみよう

- p. 53の「練習してみよう」の例題2-4をやってみよう
 - UTF-8の方をダウンロードすること
 - ダウンロードしたファイルをダブルクリックし、中のファイルやフォルダを取り出す
 - 「chap-2」というフォルダの中に入っているファイルが、例題2-4のプログラムなので、コンパイルと実行をする

例題01のコンパイルと実行

```
% javac Average.java
% java Average
```

Copyright (C) Junko Shimomura, Tokyo Woman's Christian University 2015. All rights reserved.



出席確認

- ⑧ 授業のページから、出席確認のアンケートに回答してください
- ⑧ 授業のページ→「[第1回出席確認](#)」にアクセス

Copyright (C) Tokyo Shoinen Christian University 2015. All rights reserved.