コンピュータ・サイエンス2

第14回
プログラミング・アルゴリズム(実習)(続き)

人間科学科コミュニケーション専攻
白銀 純子

第14回の内容

- プログラミング・アルゴリズムの実習(続き)

設問1

「逐次探索」とは何かを説明しなさい。

解答:
データを前から順に1つずつ比較して探していく方法

前回の質問の回答

Question!

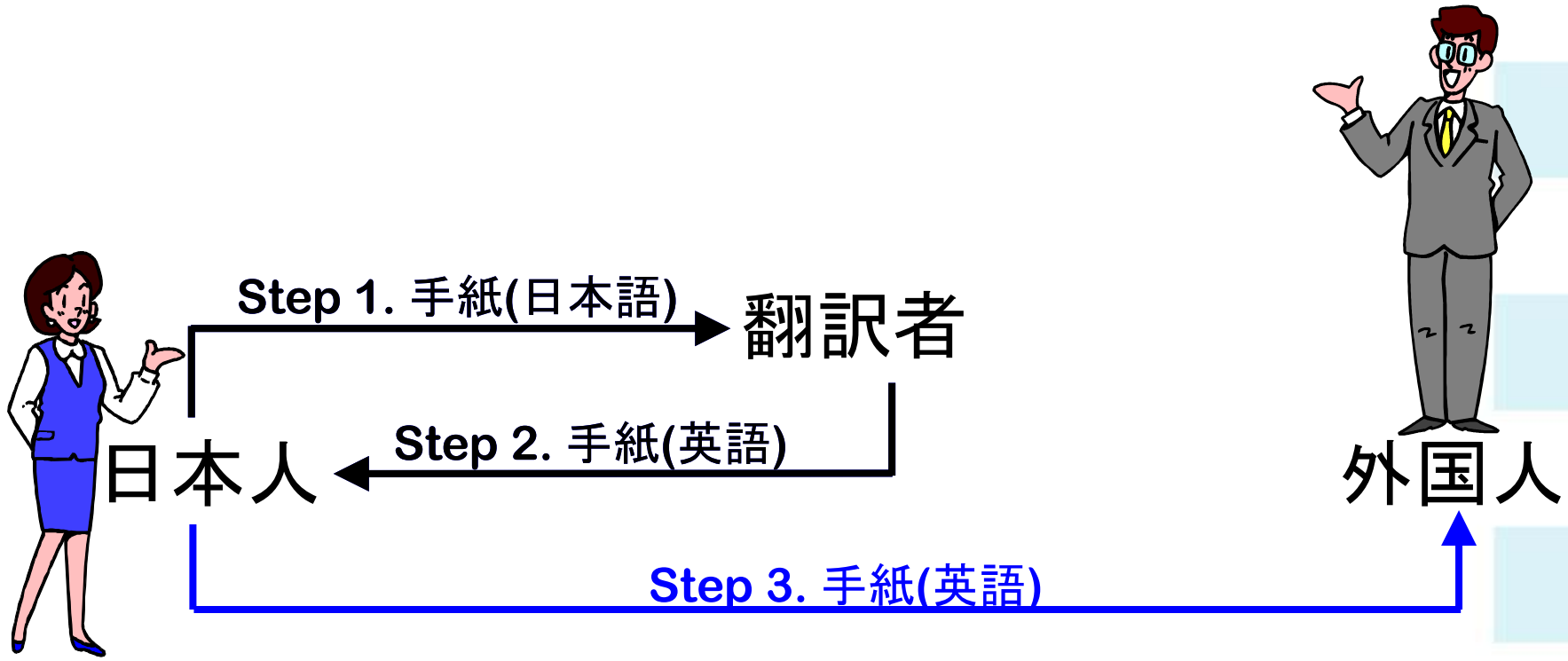
プログラミング言語の復習

変換方式

- プログラミング言語で書かれた命令書: 機械語に変換しなければ、コンピュータは実行不可能
- 変換方式は大きく分けて2種類
  - コンパイラ型: 翻訳
  - インタプリタ型: 通訳

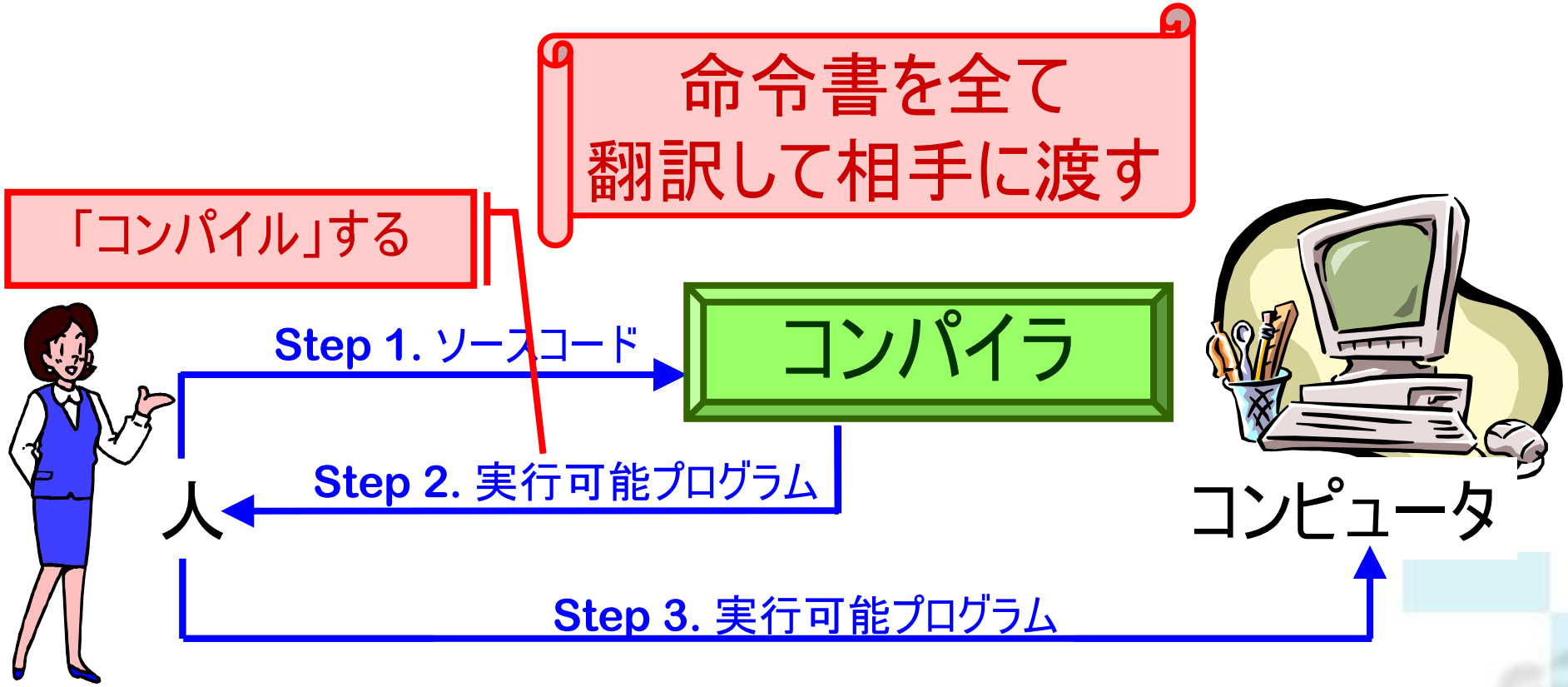
コンパイラ[概要](p. 78)

- コンパイラ: 命令書を機械語に翻訳し、コンピュータで実行可能にするためのソフトウェア

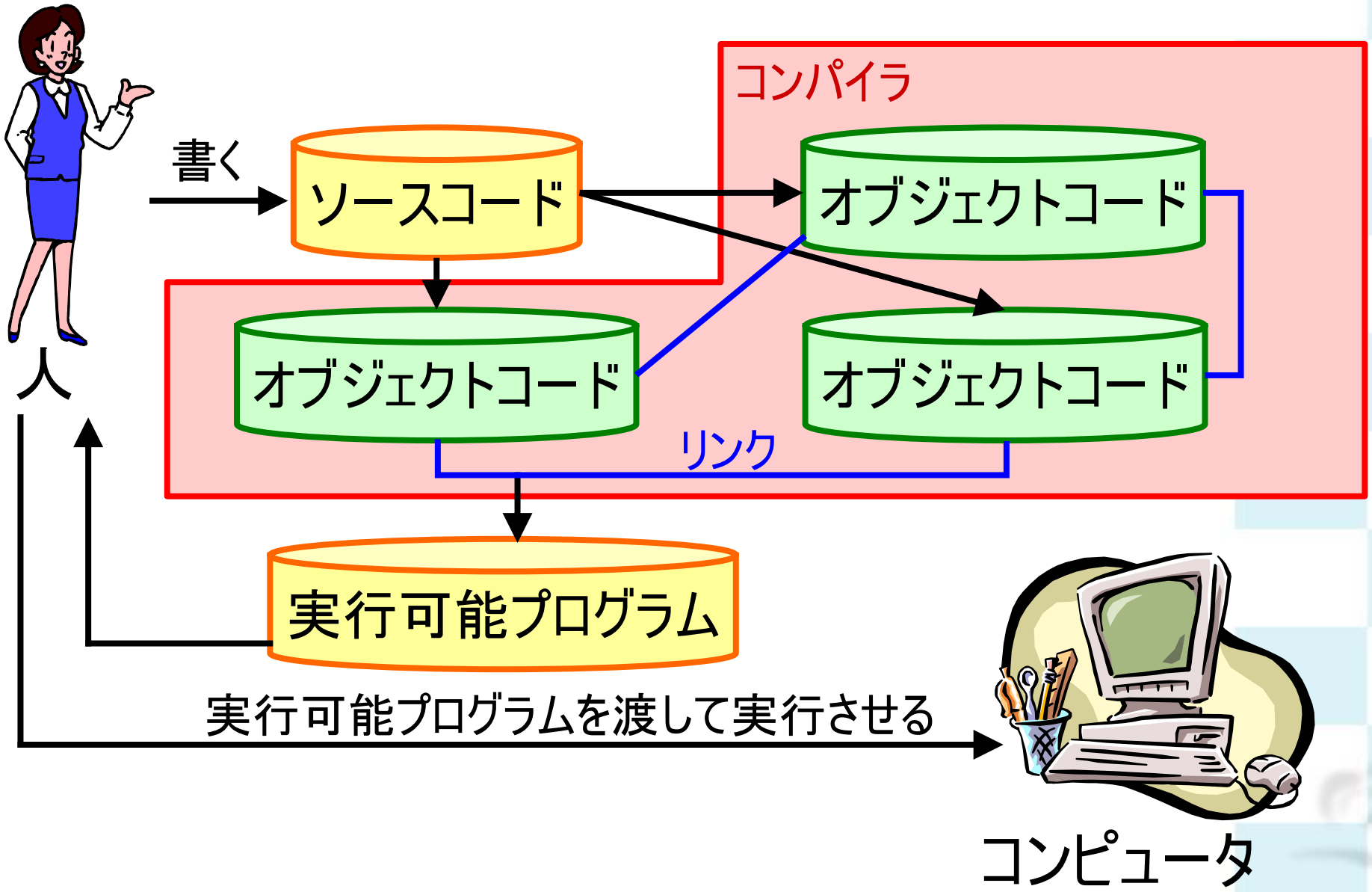


コンパイラ[概要](p. 78)

- コンパイラ: 命令書を機械語に翻訳し、コンピュータで実行可能にするためのソフトウェア



コンパイラ[詳しく](p. 78)

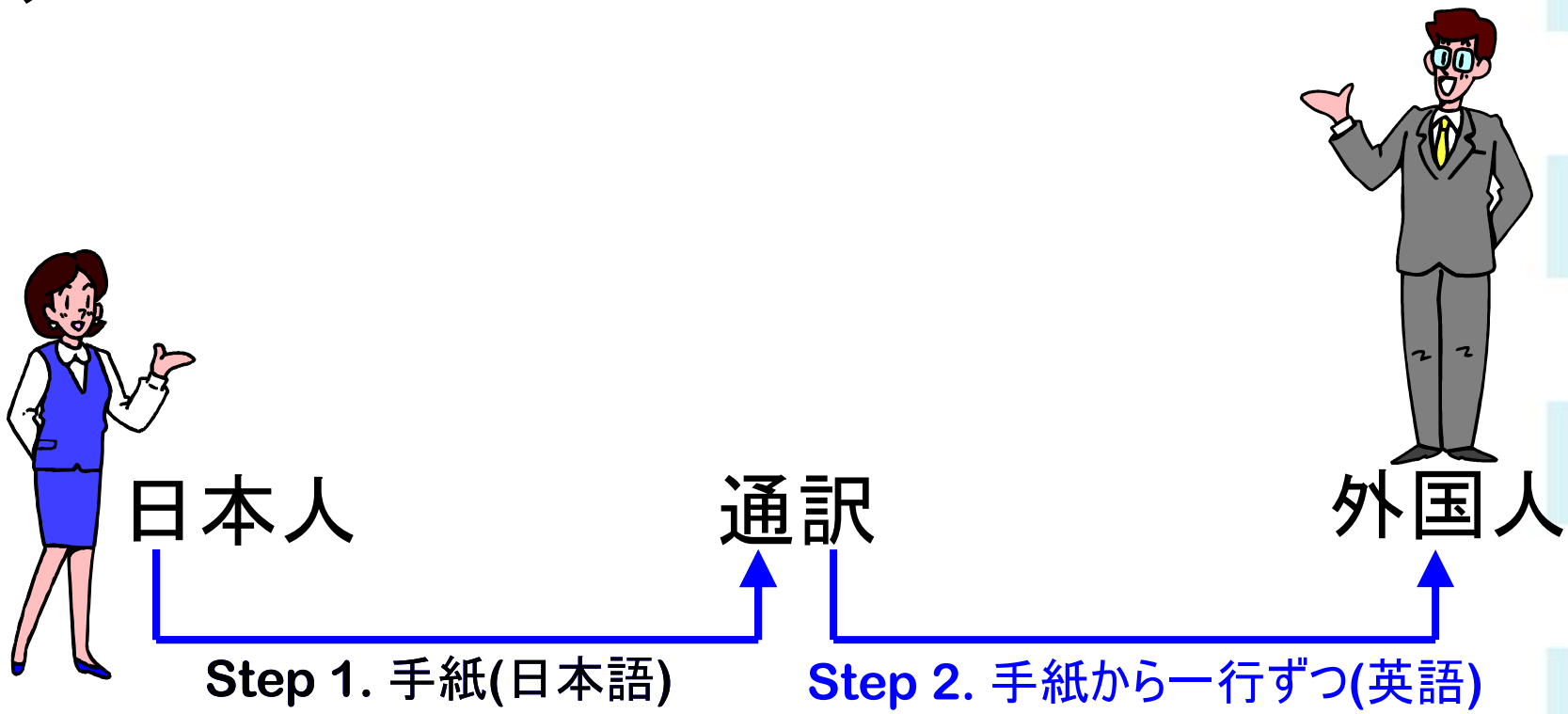


用語[1](p. 78)

- ソースコード: プログラミング言語で記述した命令書
- オブジェクトコード: ソースコードを翻訳したもの
- 実行可能プログラム: オブジェクトコードを連携させて、動作可能な形にしたもの
- プログラム: ソースコードと実行可能プログラムの双方の意味
  - どちらの意味で使われるかは、その時々で異なる

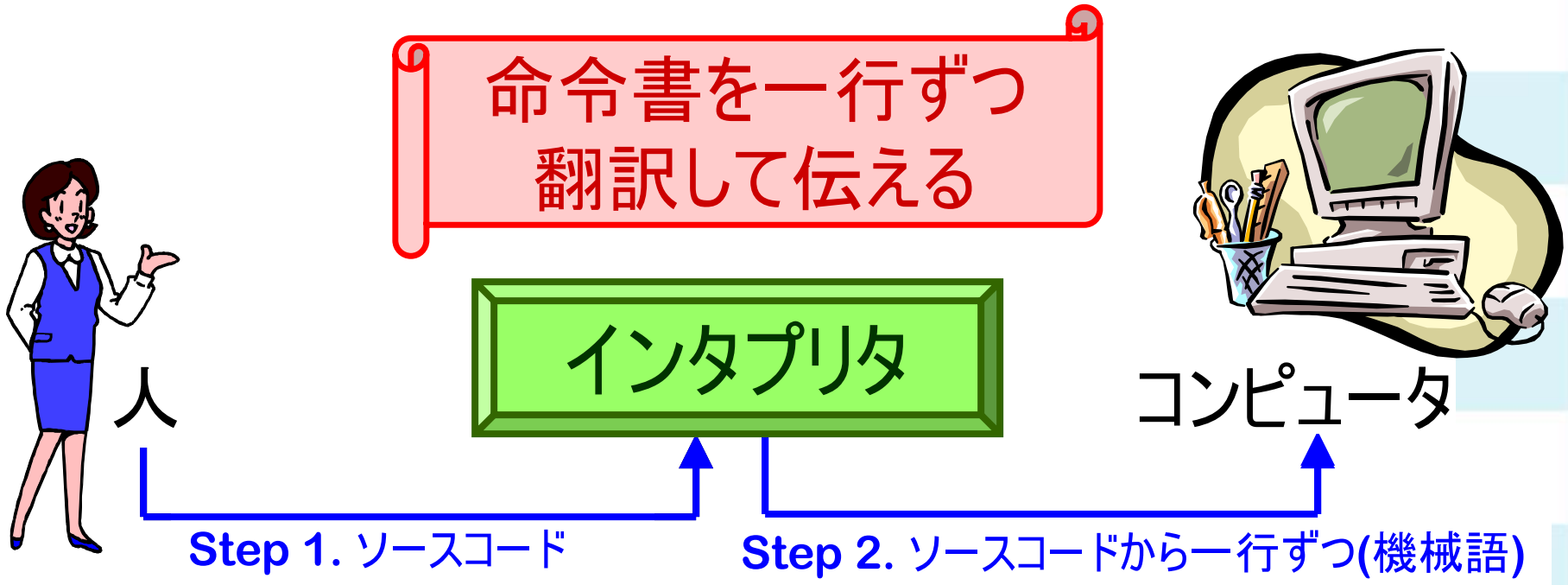
インタプリタ

- 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



インタプリタ

- 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



プログラミング実習(続き)

実習内容

- プログラミング実習
  - 命令書(プログラム)とはどのようなものか?
  - どうやって翻訳・通訳するか?
  - どうやって実行するか?
- アルゴリズム実習
  - アルゴリズムによって、そんなに処理時間に違いがあるか?

準備

- 授業のページから4つのファイルをダウンロード
  - **BubbleSort.c**
    - バブルソートをするC言語のプログラム
  - **BubbleSort.html**
    - バブルソートをするJavaScriptのプログラム
  - **MergeSort.c**
    - 併合ソートをするC言語のプログラム
  - **MergeSort.html**
    - 併合ソートをするJavaScriptのプログラム

※リンクをクリックするのではなく、右クリック→「ファイルをダウンロード」でダウンロードすること

C言語とJavaScript

- C言語
  - プログラミング言語の一種
  - 記述された命令書を機械語に翻訳した命令書を作成する形式の言語(コンパイラ型の言語)
- JavaScript
  - プログラミング言語の一種
  - 記述された命令書を機械語に通訳する形式の言語(インタプリタ型の言語)
  - Webページでよく利用

BubbleSort.c[前半1]

```
#include<stdio.h>
#include<time.h>
int main() {
  int i, j, temp, num = 30000;
  int random[num];
  for (i = 0; i < num; i++) {
    random[i] = rand();
  }
  FILE *randFile = fopen("BubbleRandomNum.txt", "w");
  for (i = 0; i < num; i++) {
    fprintf(randFile, "%d¥n", random[i]);
  }
  fclose(randFile);

  int start, end, procTime;
  start = clock();
```

並べ替えをする数の個数

BubbleSort.c[前半2]

```
#include<stdio.h>
#include<time.h>
int main() {
  int i, j, temp, num = 30000;
  int random[num];
  for (i = 0; i < num; i++) {
    random[i] = rand();
  }
  FILE *randFile = fopen("BubbleRandomNum.txt", "w");
  for (i = 0; i < num; i++) {
    fprintf(randFile, "%d¥n", random[i]);
  }
  fclose(randFile);

  int start, end, procTime;
  start = clock();
```

並べ替えをする数を
自動的に作成

BubbleSort.c[前半3]

```
#include<stdio.h>
#include<time.h>
int main() {
  int i, j, tem
  int rando
  for (i = 0; i
    random[i] = rand();
  }
  FILE *randFile = fopen("BubbleRandomNum.txt", "w");
  for (i = 0; i < num; i++) {
    fprintf(randFile, "%d¥n", random[i]);
  }
  fclose(randFile);

  int start, end, procTime;
  start = clock();
```

自動的に作った数を
「BubbleRandomNum.txt」ファイルに書き出

BubbleSort.c[後半1]

```
for (i = 0; i < num; i++) {
  for (j = 0; j < num - i - 1; j++) {
    if (random[j] > random[j + 1]) {
      temp = random[j + 1];
      random[j + 1] = random[j];
      random[j] = temp;
    }
  }
}
 end = clock();
 procTime = end - start;
 printf("Time(Bubble Sort of %d Numbers): %lf second¥n", num,
        (double) procTim

 FILE *bubbleFile = fopen("BubbleSortNum.txt", "w");
 for (i = 0; i < num; i++) {
  fprintf(bubbleFile, "%d¥n", random[i]);
 }
 fclose(bubbleFile);
}
```

バブルソートをする処理部分

BubbleSort.c[後半2]

```
for (i = 0; i < num; i++) {
  for (j = 0; j < num - i - 1; j++) {
    if (random[j] > random[j + 1]) {
      temp = random[j + 1];
      random[j + 1] = random[j];
      random[j] = temp;
    }
  }
}
end = clock();
procTime = en
printf("Time(B
        (double) procTime / CLOCKS_PER_SEC);

FILE *bubbleFile = fopen("BubbleSortNum.txt", "w");
for (i = 0; i < num; i++) {
  fprintf(bubbleFile, "%d¥n", random[i]);
}
fclose(bubbleFile);
}
```

数を並べ替えた結果を
「BubbleSortNum.txt」ファイルに書き出し

BubbleSort.html[前編1]

```
<html>
<head>
<title>バブルソート</tit
<script language="J
NUM = 5000;
random = new Array();
for (i=0; i<=NUM; i++) {
   random[i] = Math.floor(NUM * Math.random());
}
function bubbleSort() {
  for (i = 0; i < NUM; i++) {
    for (j = 0; j < NUM - i - 1; j++) {
      if (random[j] > random[j + 1]) {
        temp = random[j + 1];
        random[j + 1] = random[j];
        random[j] = temp;
      }
    }
  }
}
```

並べ替えをする
数の個数

BubbleSort.html[前編2]

```
<html>
<head>
<title>バブルソート</title>
<script language="JavaScript"><!--
NUM = 5000;
random = new Array();
for (i=0; i<=NUM; i++) {
   random[i] = Math.floor(NUM * Math.random());
}
function bubbleSort() {
  for (i = 0; i < NUM; i++) {
    for (j = 0; j < NUM - i - 1; j++) {
      if (random[j] > random[j + 1]) {
        temp = random[j + 1];
        random[j + 1] = random[j];
        random[j] = temp;
      }
    }
  }
}
```

並べ替えをする数を
自動的に作成

BubbleSort.html[前編3]

```
<html>
<head>
<title>バブルソート</title>
<script language="JavaScript"><!--
NUM = 5000;
random = new Array();
for (i=0; i<=NUM; i++) {
   random[i] = Math.floor(NU
}
function bubbleSort() {
 for (i = 0; i < NUM; i++) {
  for (j = 0; j < NUM - i - 1; j++) {
   if (random[j] > random[j + 1]) {
    temp = random[j + 1];
    random[j + 1] = random[j];
    random[j] = temp;
   }
  }
 }
}
```

バブルソートをする処理部分

BubbleSort.html[中編]

```
function writeNumber() {
 for (i=0; i<NUM; i++) document.write(random[i]," ");
 document.write("<br>");
}
// --></script>
</head>
<body>
<h1>バブルソート</h1>
<h2>並び替え前</h2>
<script langauge="JavaScript"><!--
writeNumber();
// --></script>
```

並べ替えた数をブラウザに
表示する部分

BubbleSort.html[後編]

```
<h2>並び替え後</h2>
<script langauge="JavaScript"><!--
Start = new Date();
bubbleSort();
Stop = new Date();
startTime = Start.getTime();
stopTime = Stop.getTime();
resultTime = stopTime - startTime;
writeNumber();
document.write("<br>かかった時間: " + resultTime + "msec<br>");
// --></script>
</body>
</html>
```

MergeSort.c[前編1]

```
#include<stdio.h>

int num = 30000;
void mergeSort(int random[], int start, int end) {
  int middle, i, j, k;
  int temp[num];
  if (start >= end) {
    return;
  }
  middle = (start + end) / 2;
  mergeSort(random, start, middle);
  mergeSort(random, middle + 1, end);

  for (i = start; i <= middle; i++) {
    temp[i] = random[i];
  }
  for (i = middle + 1, j = end; i <= end; i++, j--) {
    temp[i] = random[j];
  }
  i = start;
  j = end;
```

並べ替えをする
数の個数

MergeSort.c[前編2]

```
#include<stdio.h>

int num = 30000;
void mergeSort(int random[], int start, int end) {
  int middle, i, j, k;
  int temp[num];
  if (start >= end) {
    return;
  }
  middle = (start + end) / 2;
  mergeSort(random, start, middle);
  mergeSort(random, middle + 1, end);

  for (i = start; i <= middle; i++) {
    temp[i] = random[i];
  }
  for (i = middle + 1, j = end; i <= end; i++, j--) {
    temp[i] = random[j];
  }
  i = start;
  j = end;
```

併合ソートの処理部分(前半)

MergeSort.c[中編1]

```
  for (k = start; k < end; k++) {
    if (temp[i] <= temp[j]) {
      random[k] = temp[i++];
    } else {
      random[k] = temp[j--];
    }
  }
}
int main() {
  int i, j, temp;
  int random[num];
  for (i = 0; i < num; i++) {
    random[i] = rand();
  }
  FILE *randFile = fopen("MergeRandomNum.txt", "w");

  for (i = 0; i < num; i++) {
    fprintf(randFile, "%d¥n", random[i]);
  }
  fclose(randFile);
```

併合ソートの処理部分(後半)

MergeSort.c[中編2]

```
  for (k = start; k < end; k++) {
    if (temp[i] <= temp[j]) {
      random[k] = temp[i++];
    } else {
      random[k] = temp[j--];
    }
  }
}
int main() {
  int i, j, temp;
  int random[num];
  for (i = 0; i < num; i++) {
    random[i] = rand();
  }
  FILE *randFile = fopen("MergeRandomNum.txt", "w");

  for (i = 0; i < num; i++) {
    fprintf(randFile, "%d¥n", random[i]);
  }
  fclose(randFile);
```

並べ替えをする数を
自動的に作成

MergeSort.c[中編3]

```
  for (k = start; k < end; k++) {
    if (temp[i] <= temp[j]) {
      random[k] = temp[i++];
    } else {
      random[k] = temp[j--];
    }
  }
}
int main() {
  int i, j, t;
  int rand
  for (i = 0
    random[i] = rand();
  }
  FILE *randFile = fopen("MergeRandomNum.txt", "w");

  for (i = 0; i < num; i++) {
    fprintf(randFile, "%d¥n", random[i]);
  }
  fclose(randFile);
```

自動的に作った数を
「MergeRandomNum.txt」ファイルに書き出し

MergeSort.c[後編]

```
  int start, end, procTime;
  start = clock();
  mergeSort(random, 0, num - 1);
  end = clock();

  procTime = end - start;
  printf("Time(Merge Sort of %d Numbers): %f second¥n", num,
         (double) procTime / CLOCKS_PER_SEC);
  FILE *mergeFile = fopen("MergeSortNum.txt", "w");
  for (i = 0; i < num; i++) {
    fprintf(mergeFile, "%d¥n", random[i]);
  }
  fclose(mergeFile);
}
```

数を並べ替えた結果を
「MergeSortNum.txt」ファイルに書き出し

MergeSort.html[前編1]

```
<html>
<head>
<title>併合ソート</title>
<script language="J
NUM = 5000;
random = new Array();
for (i = 0; i <= NUM; i++) {
    random[i] = Math.floor(NUM * Math.random());
}
temp = new Array();
function mergeSort(start,end) {
    if (start >= end) return;
        var middle = Math.floor((start + end) / 2);
        mergeSort(start, middle);
        mergeSort(middle + 1, end);
        var p = 0;
        for (i = start; i <= middle; i++) temp[p++] = random[i];
            var i = middle + 1;
            var j = 0;
            var k = start;
```

並べ替えをする
数の個数

MergeSort.html[前編2]

```
<html>
<head>
<title>併合ソート</title>
<script language="JavaScript"><!--
NUM = 5000;
random = new Array();
for (i = 0; i <= NUM; i++) {
    random[i] = Math.floor(NUM * Math.random());
}
temp = new Array();
function mergeSort(start,end) {
    if (start >= end) return;
        var middle = Math.floor((start + end) / 2);
        mergeSort(start, middle);
        mergeSort(middle + 1, end);
        var p = 0;
        for (i = start; i <= middle; i++) temp[p++] = random[i];
            var i = middle + 1;
            var j = 0;
            var k = start;
```

並べ替えをする数を
自動的に作成

MergeSort.html[前編3]

```
<html>
<head>
<title>併合ソート</title>
<script language="JavaScript"><!--
NUM = 5000;
random = new Array();
for (i = 0; i <= NUM; i++) {
    random[i] = Math.floor(NUM * Math.random());
}
temp = new Array();
function mergeSort(start,end) {
    if (start >= end) return;
    var middle = Math.floor((start + end) / 2);
    mergeSort(start, middle);
    mergeSort(middle + 1, end);
    var p = 0;
    for (i = start; i <= middle; i++) temp[p++] = random[i];
        var i = middle + 1;
        var j = 0;
        var k = start;
```

併合ソートの処理部分(前半)

MergeSort.html[中編1]

```
            while ((i <= end) && (j < p)) {
                if (temp[j] <= random[i])  {
                    random[k++] = temp[j++];
                } else {
                    random[k++] = random[i++];
                }
            }
            while (j < p) {
                random[k++] = temp[j++];
            }
}
function writeNumber() {
    for (i = 0; i < random.length; i++) {
        document.write(random[i]," ");
    }
    document.write("<br>");
}
// --></script>
</head>
```

併合ソートの処理部分(後半)

MergeSort.html[中編2]

```
            while ((i <= end) && (j < p)) {
                if (temp[j] <= random[i])  {
                    random[k++] = temp[j++];
                } else {
                    random[k++] = random[i++];
                }
            }
            while (j < p) {
                random[k++] = temp[j++];
            }
}
function writeNumber() {
    for (i = 0; i < random.length; i++) {
        document.write(random[i]," ");
    }
    document.write("<br>");
}
// --></script>
</head>
```

並べ替えた数をブラウザに
表示する部分

MergeSort.html[後編]

```
<body>
<h1>併合ソート</h1>
<h2>並び替え前</h2>
<script langauge="JavaScript"><!--
writeNumber();
// --></script>

<h2>並び替え後</h2>
<script langauge="JavaScript"><!--
Start = new Date();
mergeSort(0,random.length-1);
Stop = new Date();
startTime = Start.getTime();
stopTime = Stop.getTime();
resultTime = stopTime - startTime;
writeNumber();
document.write("<br>かかった時間: " + resultTime + "msec<br>");
// --></script>
</body>
</html>
```

ファイルの役割

- **BubbleSort.c**
  - 「**BubbleRandomNum.txt**」に、並び替え前の数を保存
  - 「**BubbleSortNum.txt**」に、並び替え後の数を保存
- **BubbleSort.html**
  - 並び替え前と後の数をブラウザに表示
- **MergeSort.c**
  - 「**MergeRandomNum.txt**」に、並び替え前の数を保存
  - 「**MergeSortNum.txt**」に、並び替え後の数を保存
- **MergeSort.html**
  - 並び替え前と後の数をブラウザに表示

BubbleSort.c, MergeSort.c[1]

- コンパイルして機械語のファイルを作成
  1. Finder→「アプリケーション」→「ユーティリティ」→「ターミナル」をダブルクリック
  2. それぞれのファイルをコンパイル
     - BubbleSort.cの場合:
       gcc -o BubbleSort BubbleSort.c
       と入力し、「Return」キーを押す
     - MergeSort.cの場合:
       gcc -o MergeSort MergeSort.c
       と入力し、「Return」キーを押す

- gcc: C言語のプログラムを機械語に翻訳するためのコンパイラ
  - ✓「gcc -o ファイル1 ファイル2」で、「ファイル2のプログラムを機械語に翻訳し、ファイルに保存する」という命令

BubbleSort.c, MergeSort.c[2]

3. Finderで、「BubbleSort」ファイルと「MergeSort」ファイルができていることを確認
   - BubbleSort: BubbleSort.cを機械語に翻訳したファイル
   - MergeSort: MergeSort.cを機械語に翻訳したファイル

BubbleSort.c, MergeSort.c[2]

- プログラミング言語が機械語に翻訳されているかを確認
  - **BubbleSort.cをJeditで開いてみる**
  - **BubbleSortをJeditで開いてみる**
  - **MergeSort.cをJeditで開いてみる**
  - **MergeSortをJeditで開いてみる**

- **Jedit:** プログラムなどのテキストのみのファイルを開くためのアプリケーション
  - ✓ ファイルの開き方: **Jedit**を起動して、メニューバーの「ファイル」→「開く」で表示されたウィンドウで、開きたいファイルを選択

BubbleSort.c, MergeSort.c[3]

- プログラムを実行
  - BubbleSort.cの場合: ターミナルで「./BubbleSort」と入力し、「Return」キーを押す
  - MergeSort.cの場合: ターミナルで「./MergeSort」と入力し、「Return」キーを押す

- BubbleRandomNum.txtとBubbleSortNum.txtを開き、数が並び替えされているかどうかを確認
- MergeRandomNum.txtとMergeSortNum.txtを開き、数が並び替えされているかどうかを確認
- 「Time: かかった時間 second」とターミナル上に表示されるので、かかった時間を比較

※かかった時間の単位は秒

BubbleSort.html, MergeSort.html

BubbleSort.htmlとMergeSort.htmlをWebブラウザで表示

- ブラウザで表示された数が、「並び替え前」と「並び替え後」で並び替えられているかどうかを確認
- 「かかった時間msec」(ブラウザの一番下に表示)ので、BubbleSort.htmlとMergeSort.html かかった時間を比較
- **BubbleSort.cとBubbleSort.htmlでかかった時間を比較 (※BubbleSort.cは数が30000個、BubbleSort.htmlは数が5000個)**
- **MergeSort.cとMergeSort.htmlでかかった時間を比較 (※MergeSort.cは数が30000個、MergeSort.htmlは数が5000個)**

Question!

期末試験

- 期末試験: 1月29日(火) 1限 24101教室
  - 時間: 60分
  - 持ち込み: すべて不可
  - 内容: 後期の講義内容すべて
    - 用語の意味の選択・説明
    - 各種概念に関しての説明
    - 回路図・真理値表
    - etc.