

Java基礎文法

2017/10/16(月) プログラミングII 第二回
福井大学 工学研究科 情報・メディア工学専攻
長谷川達人

本日の目次

- 前回のコメントに対する返信
- 開発環境の使い方
- 駆け足だった点の復習
- 課題演習

前回のコメントに対する返信

- 家のPCでもJava書きたい場合どうすればよいか？
 - 「Eclipse 日本語」で出てくる「Pleiades」のページに行き、Eclipse NEONのJava, FullEditionあたりをDLして使う。
- わからないことは質問に行ってもよいか？
 - 講義中の質問も講義外の質問も受け付ける。602室or下記のアドレスに連絡する。[t-hase@u-fukui.ac.jp]
- 課題にこたえられない場合は欠席となるのか？
 - ならない。書くだけ書いて出せば出席にはする。未提出の場合欠席とする。

前回のコメントに対する返信

- 変数型の違いや範囲がわからなかった.

型名	大きさ	値
boolean	1bit	true or false
char	16bit 文字	Unicode文字 ¥u0000~¥uFFFF
byte	8bit 整数	-128~127
short	16bit 整数	-32768~32767
int	32bit 整数	-2147483648~2147483647
long	64bit 整数	$-9.22 \times 10^{18} \sim 9.22 \times 10^{18}$
float	32bit 小数	
double	64bit 小数	
String	メモリの許す限り	

前回のコメントに対する返信

- もう少しゆっくり話してほしい.
 - すみません. . . ノッてくると早口になる癖があるので、わからなかった点は聞き返してもらえると助かります.
- JavaとCの違いをもう少し詳しく知りたい.
 - Javaの利点は前回資料のP15の通り（オブジェクト指向，ガベージコレクション，マルチスレッド，Java仮想マシン）.
 - Cの利点は，詳細なメモリ管理が実装できること，一般的に高速な処理が実現できることである.

前回のコメントに対する返信

- どのくらい勉強すれば試験でいい点が取れるか？
- 過去問等試験勉強のための資料はあるか？
 - 残念ながら今期初開講なのでない.
 - 試験でいい点を取るには、各回の課題や練習問題を、資料を見ずともスラスラ解けるレベルになるよう練習するとよい.
- コマンドライン引数にファイル名を含まないのはなぜか？
 - コンパイル : `javac Sample1.java`
 - 実行 : `java Sample1 aa ii uu`
 - 上記の通り、javaコマンドで見ると、ファイル名は引数に含むが、`String[] args`に送られるのは`aa`, `ii`, `uu`である.

前回のコメントに対する返信

- なぜC言語よりも書く量が増えるのか？（いい質問！）
 - オブジェクト指向で出てくる**クラス**という考え方のせいである.
 - 詳細は追々説明するが、`System.out.println("")`の場合、`System`というクラスの`out`というフィールドの`println()`というメソッドを実行するという意味がある.
- なぜStringのSだけ大文字なのか？（いい質問！）
 - 実はint等は型なのだが、Stringはクラスである.
 - 本来クラスの初期化は `String str=""`;という書き方をしないのだが、Stringだけは型のように上記の記述方法が**例外的に**許可されている.

本日の目次

- 前回のコメントに対する返信
- **開発環境の使い方**
- 駆け足だった点の復習
- 課題演習



使用する開発環境

コマンドでもコンパイルできるが、**Eclipse**という**IDE**を使う。

IDE

統合開発環境のことで、GUIベースでエディタやコンパイラ、デバッガなどを簡単に使用できるようにしたものである。

Eclipse

オープンソースの統合開発環境（非常に有能）で、Java開発はEclipseを使うことが多い。

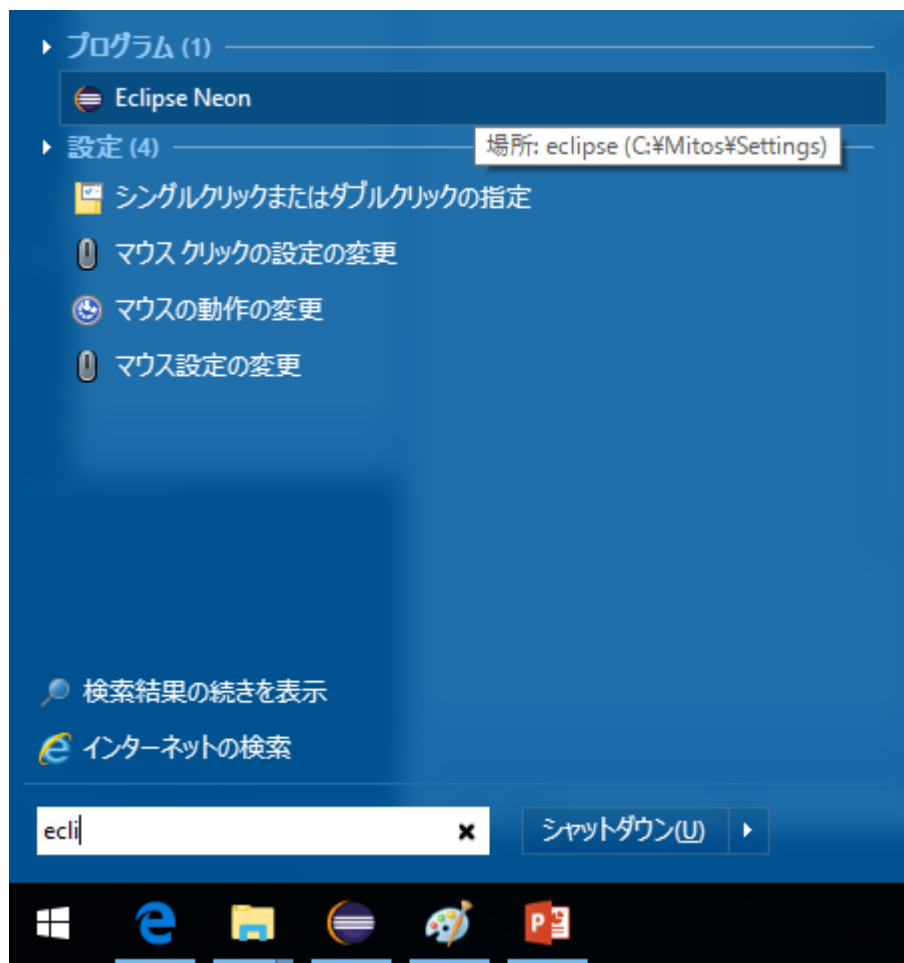
<http://mergedoc.osdn.jp/>

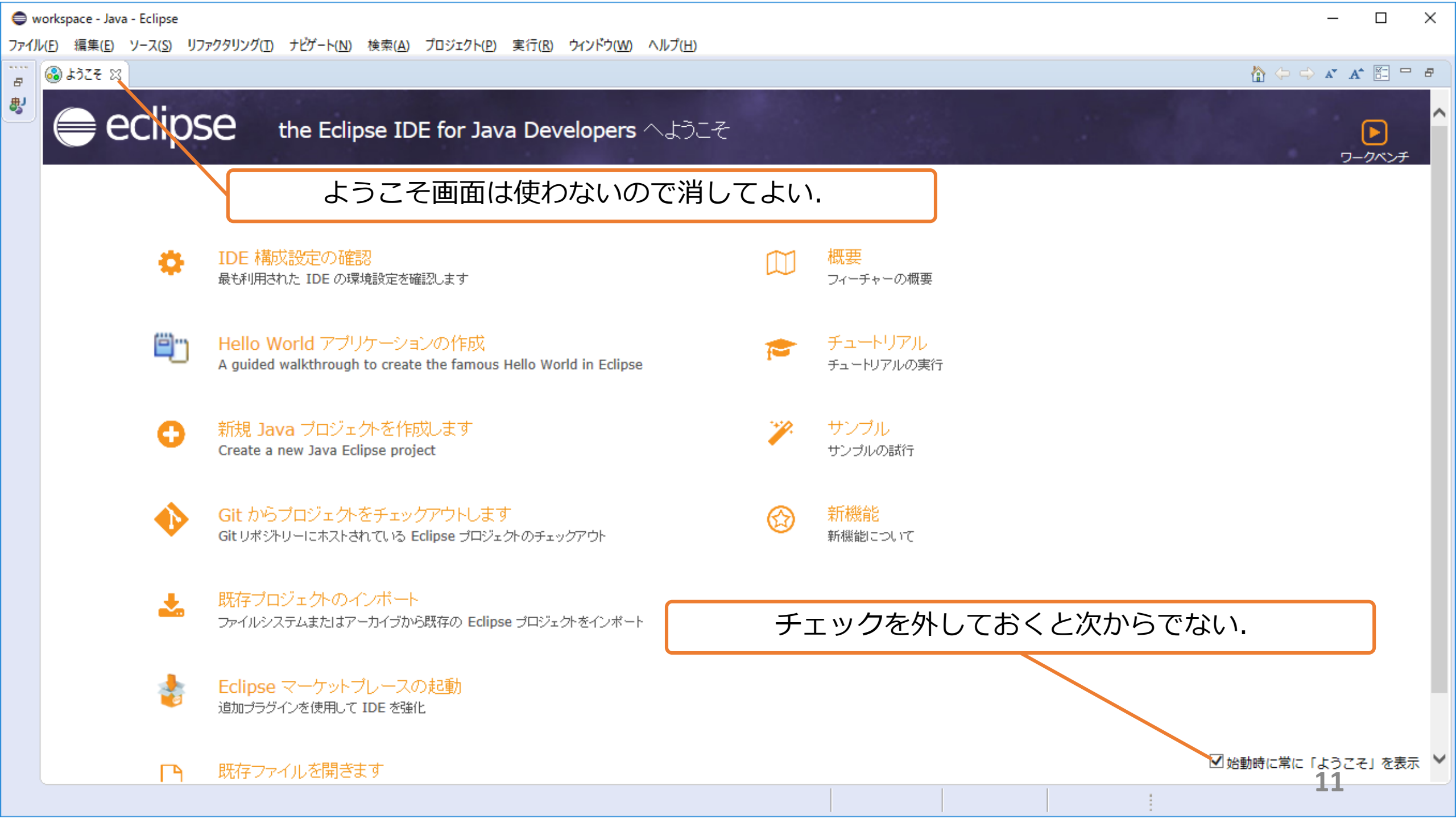
※自宅で学習したい人は上記から日本語版フリーDL可能

使用する開発環境

まずは、**Eclipse**を起動する.

> Windowsキーを押して「ecli」等を入力すると出てくるので起動する.






workspace - Java - Eclipse


ファイル(F) 編集(E) ソース(S) リファクタリング(R) ナビゲート(N) 検索(A) プロジェクト(P) 実行(R) ウィンドウ(W) ヘルプ(H)


ようこそ


the Eclipse IDE for Java Developers へようこそ


ワークベンチ


ようこそ画面は使わないので消してよい。


 **IDE 構成設定の確認**
最も利用された IDE の環境設定を確認します


 **概要**
フィーチャーの概要


 **Hello World アプリケーションの作成**
A guided walkthrough to create the famous Hello World in Eclipse


 **チュートリアル**
チュートリアルの実行


 **新規 Java プロジェクトを作成します**
Create a new Java Eclipse project


 **サンプル**
サンプルの試行

 **Git からプロジェクトをチェックアウトします**
Git リポジトリにホストされている Eclipse プロジェクトのチェックアウト

 **新機能**
新機能について

 **既存プロジェクトのインポート**
ファイルシステムまたはアーカイブから既存の Eclipse プロジェクトをインポート

 **Eclipse マーケットプレースの起動**
追加プラグインを使用して IDE を強化

 **既存ファイルを開きます**

チェックを外しておくと次からでない。

☒ 始動時に常に「ようこそ」を表示

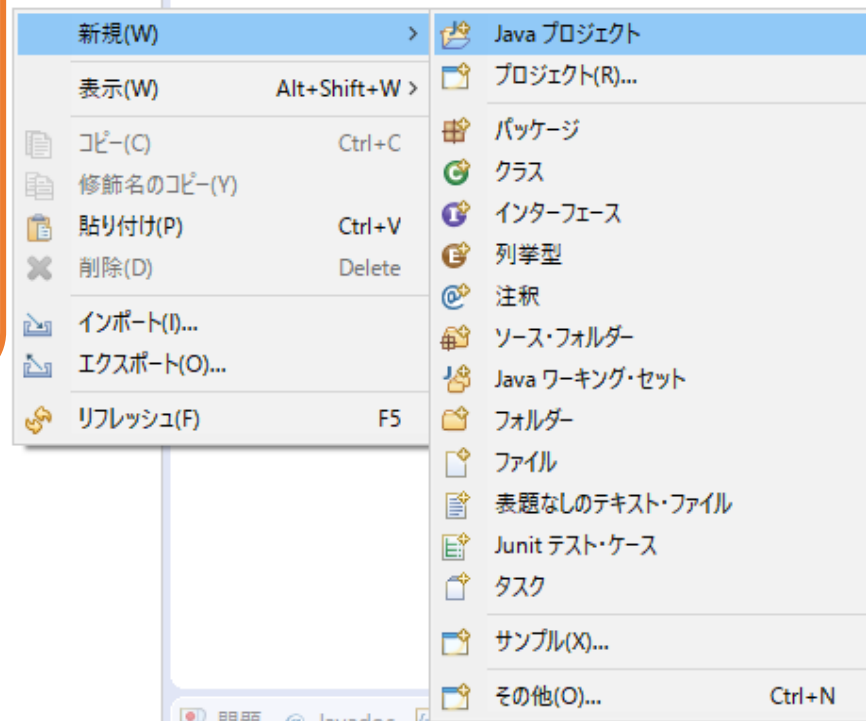
11



パッケージ・エクスプローラー

このゾーンで
1. 右クリック
2. 新規 (W)
3. Javaプロジェクト
を選択する。

早速プロジェクトを作る。



クイック・アクセス

タスク・リスト



検索

すべて アクティブに...

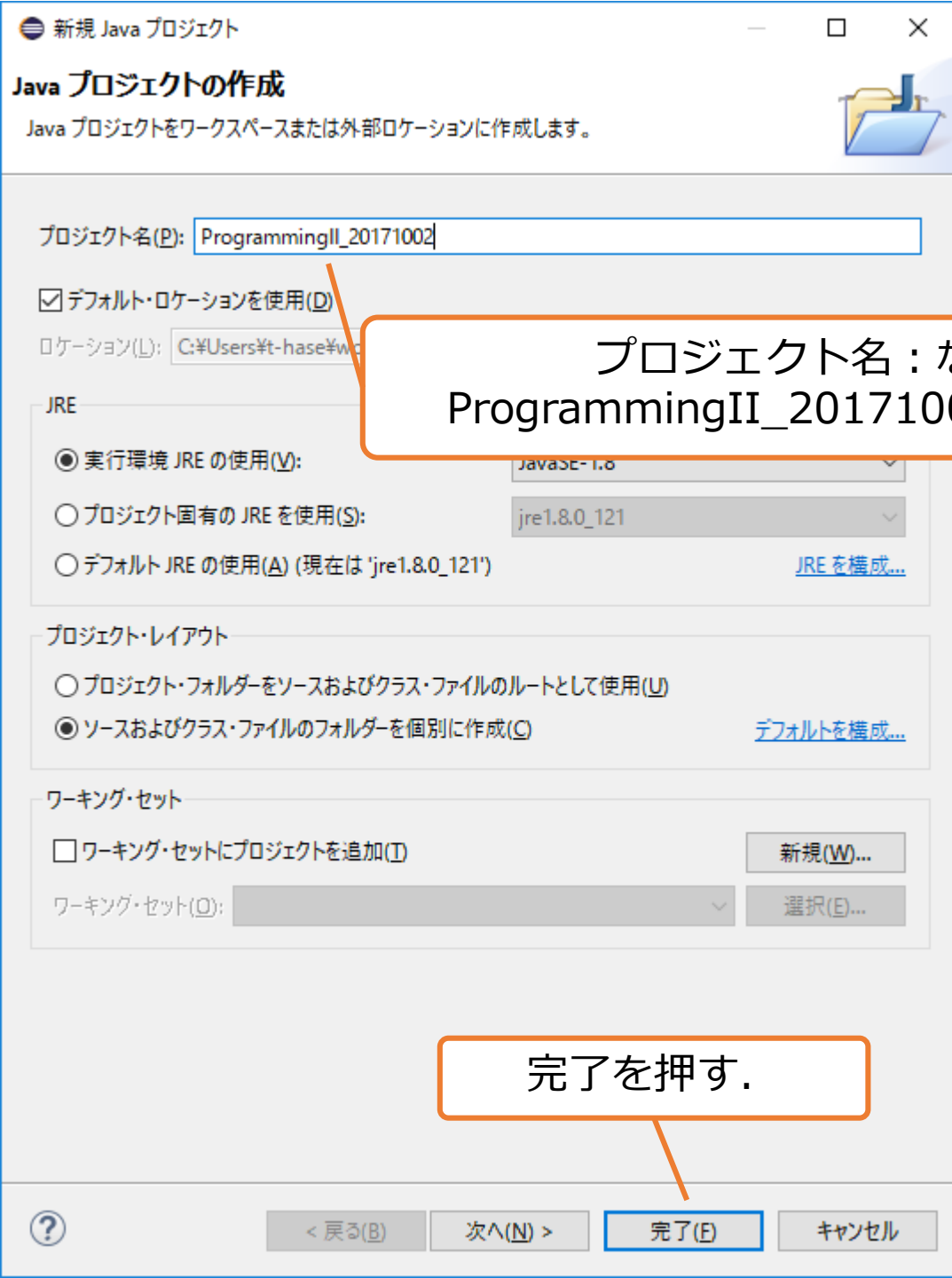
アウトライン

表示するアウトラインはありません。

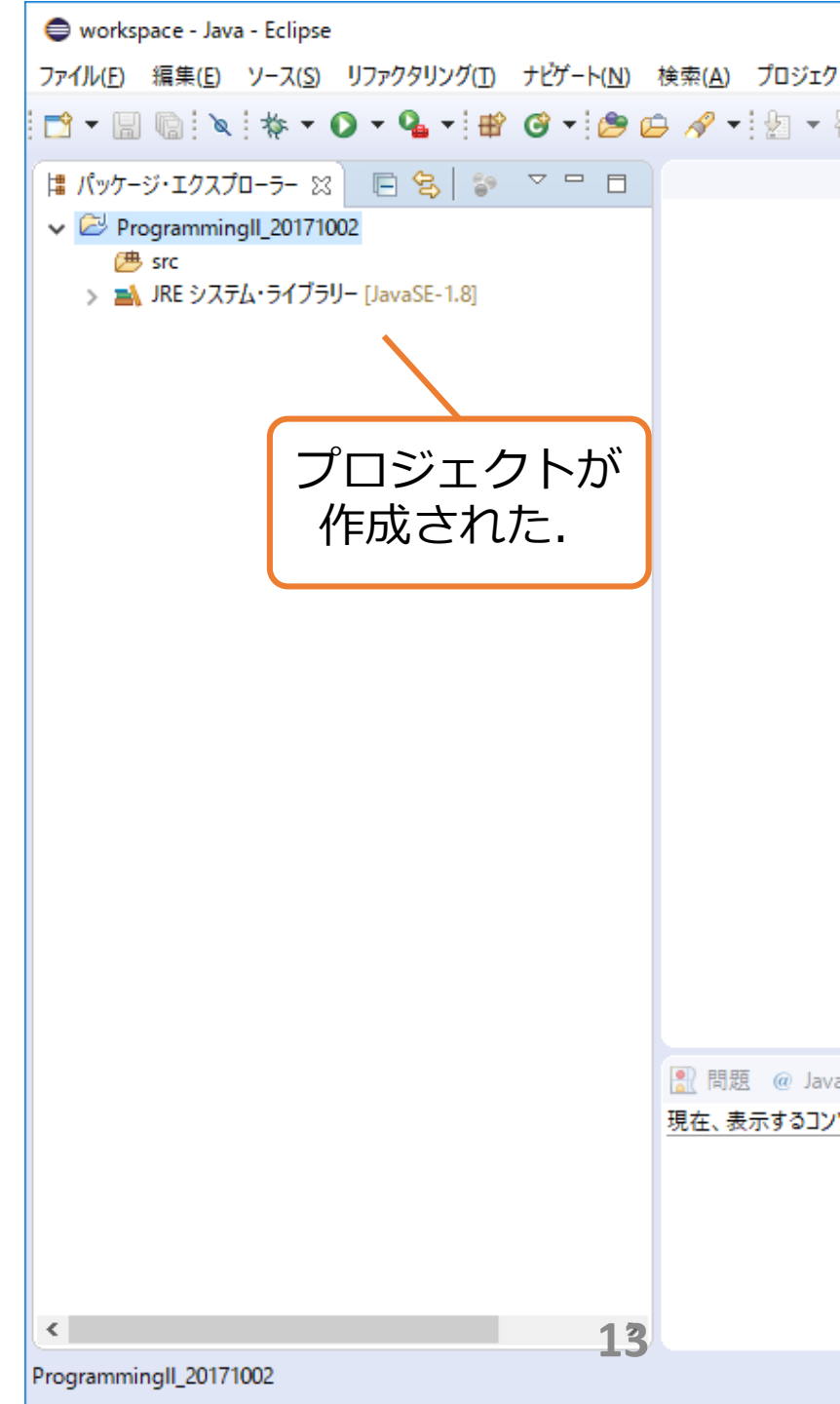
問題 @ Javadoc

現在、表示するコンソールがありません。



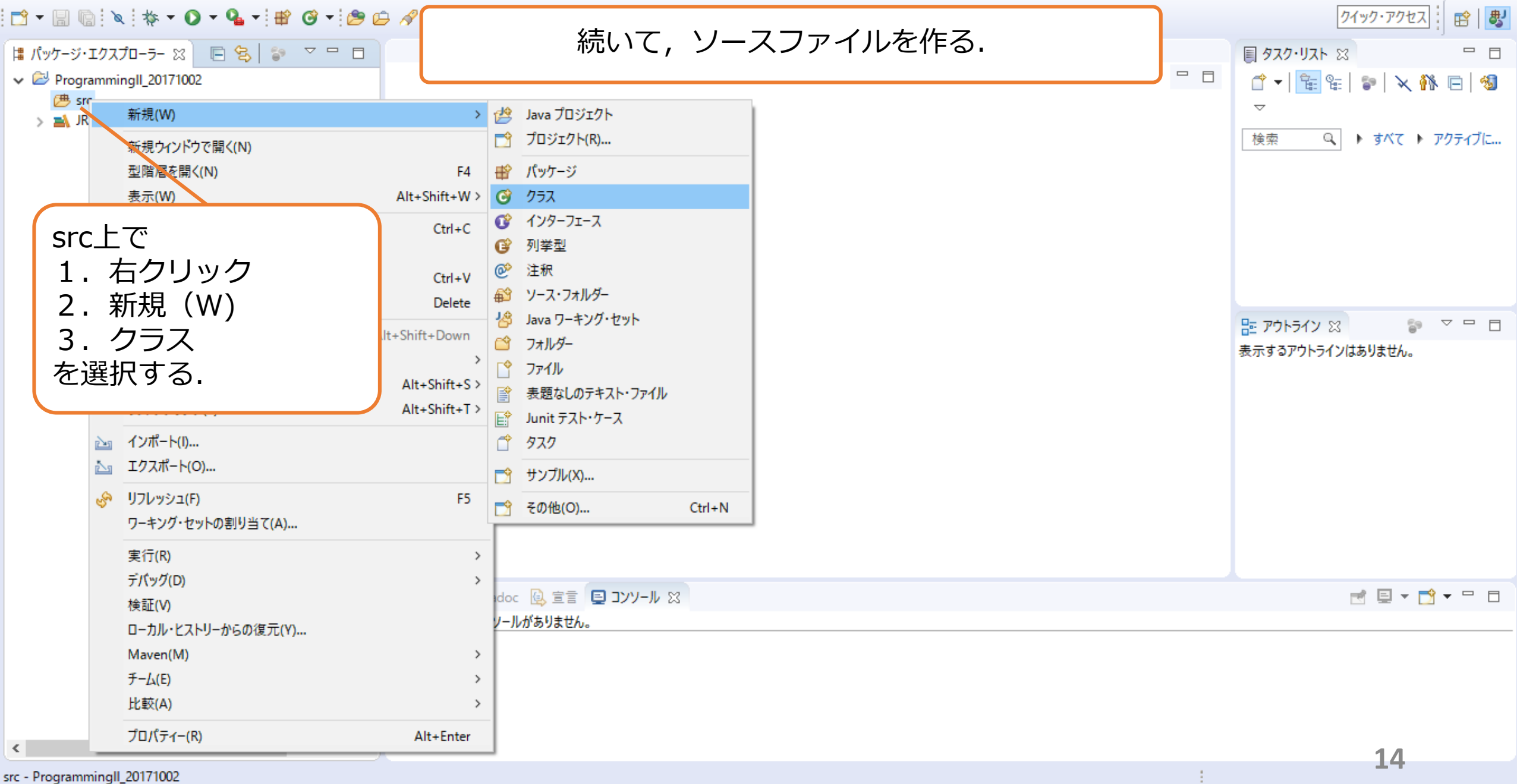


プロジェクト名：なんでもよいが
ProgrammingII_20171002としておくとよい。



続いて、ソースファイルを作る。

src上で
1. 右クリック
2. 新規 (W)
3. クラス
を選択する。



新規 Java クラス

Java クラス

⚠ デフォルト・パッケージの使用は推奨されません。

ソース・フォルダー(D): ProgrammingII_20171002/src 参照(O)...

パッケージ(K):

☐ エンクロージング型(Y):

名前(M): Renshu1_1

修飾子: ☒ public ☐ パッケージ(C) ☐ private ☐ protected
☐ abstract(T) ☐ final(L) ☐ static(C)

スーパークラス(S): java.lang.Object 参照(E)...

インターフェース(I): 追加(A)... 除去(R)

どのメソッド・スタブを作成しますか?

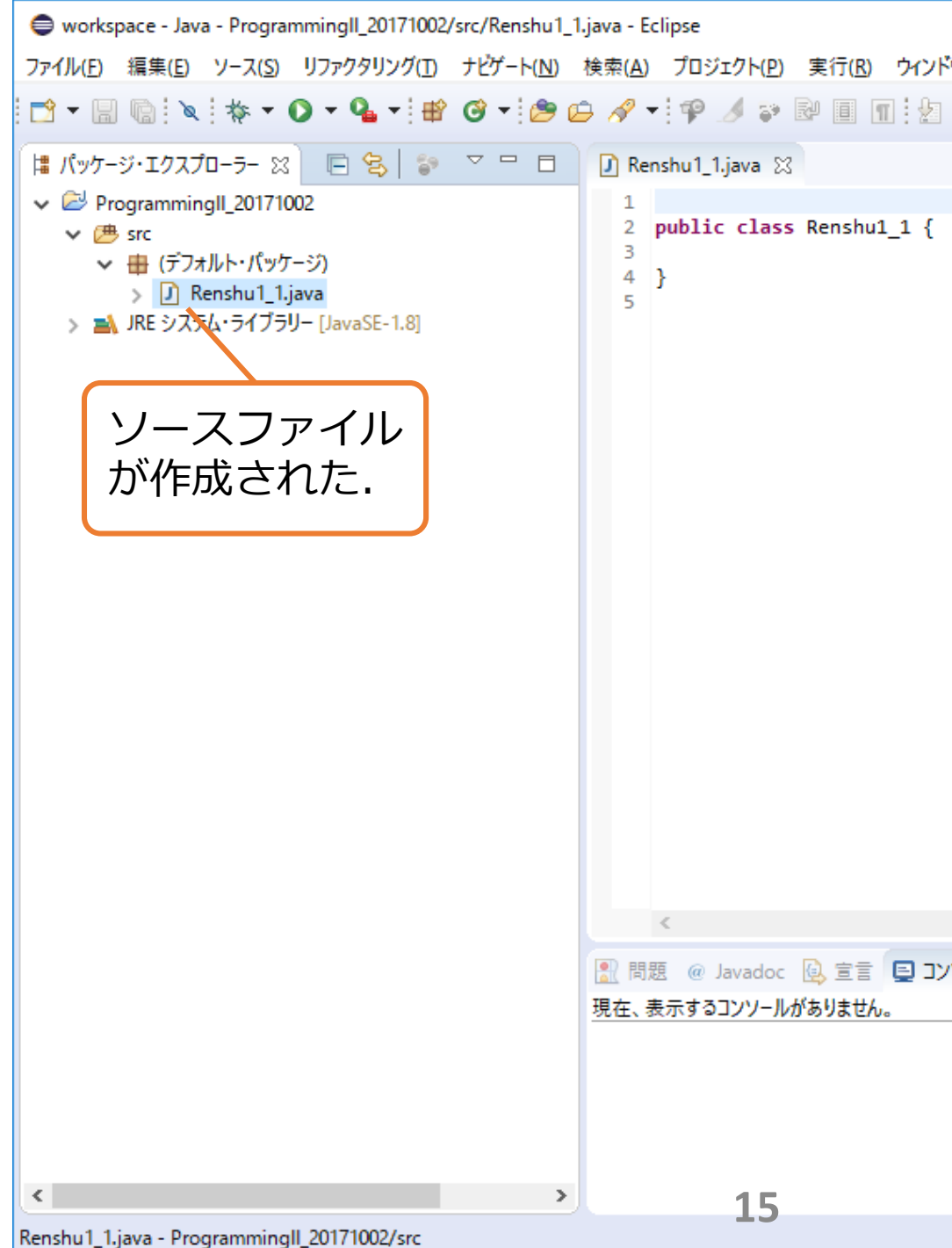
☐ public static void main(String[] args)(V)
☐ スーパークラスからのコンストラクター(U)
☒ 継承された抽象メソッド(H)

コメントを追加しますか? (テンプレートの構成およびデフォルト値については[ここ](#)を参照)
☐ コメントの生成(G)

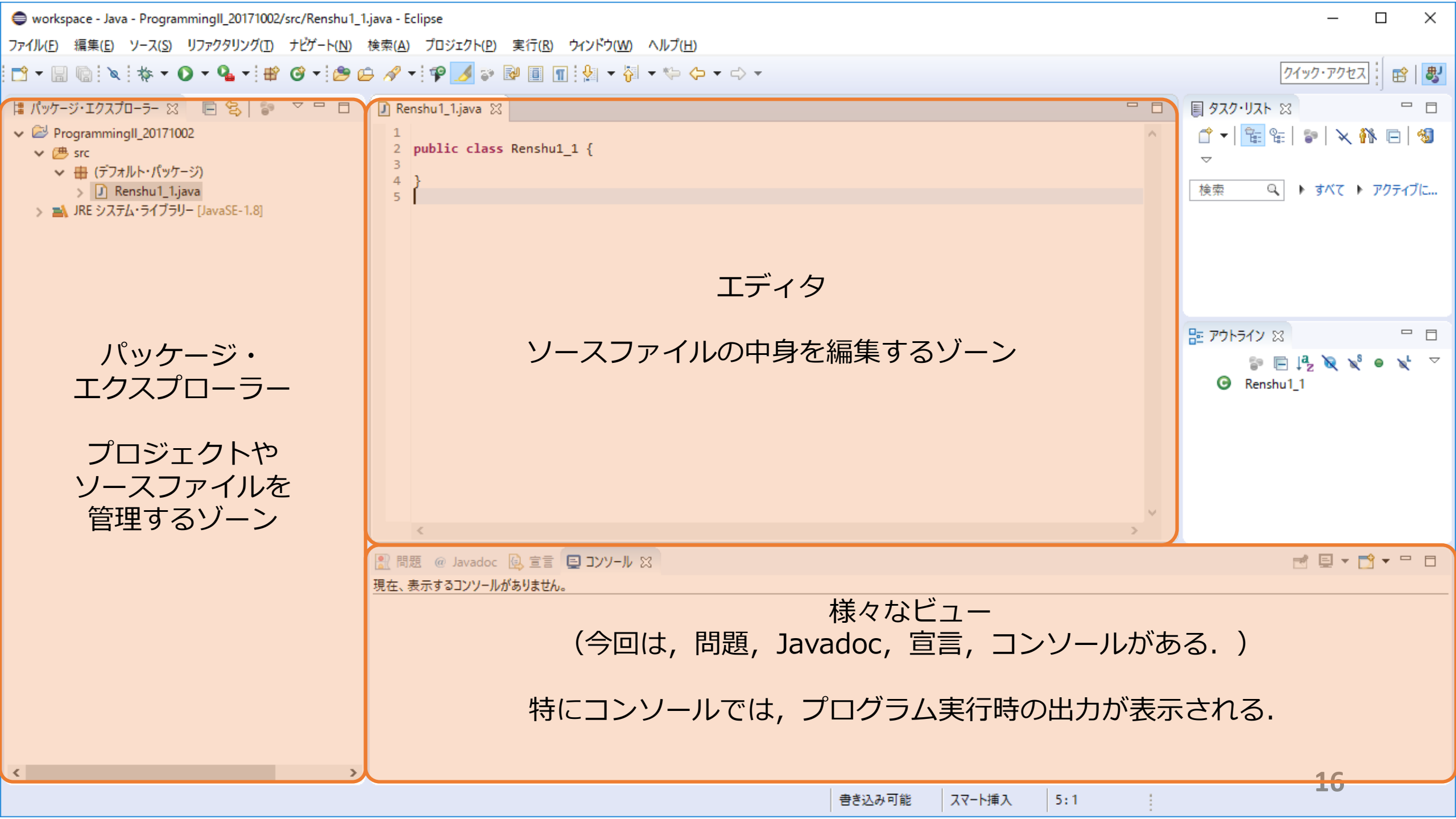
完了(E) キャンセル

ファイル名：なんでもよいが
Renshu1_1としておくとい。

完了を押す。



ソースファイル
が作成された。



パッケージ・エクスプローラー

プロジェクトやソースファイルを管理するゾーン

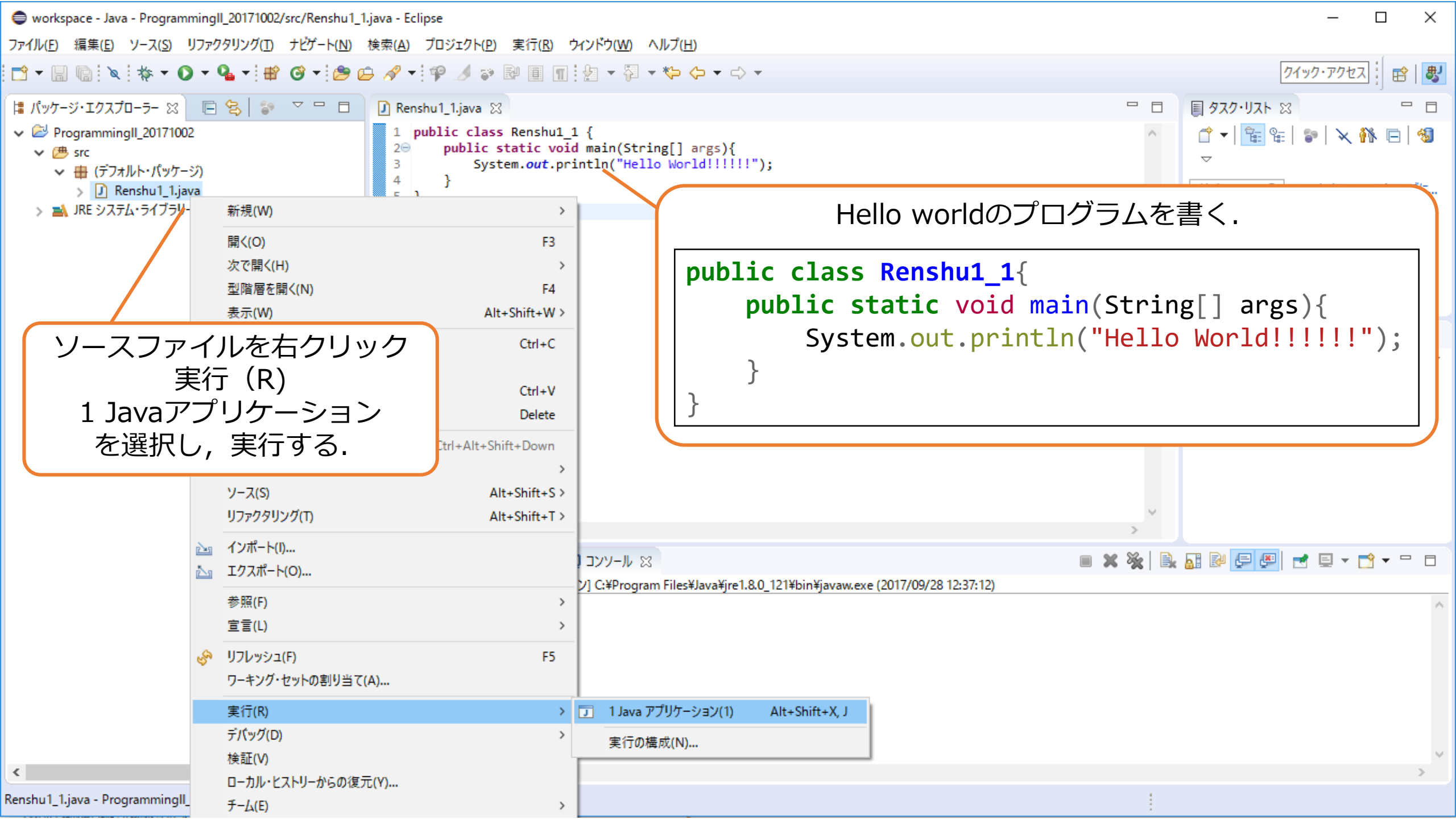
エディタ

ソースファイルの中身を編集するゾーン

様々なビュー

(今回は, 問題, Javadoc, 宣言, コンソールがある.)

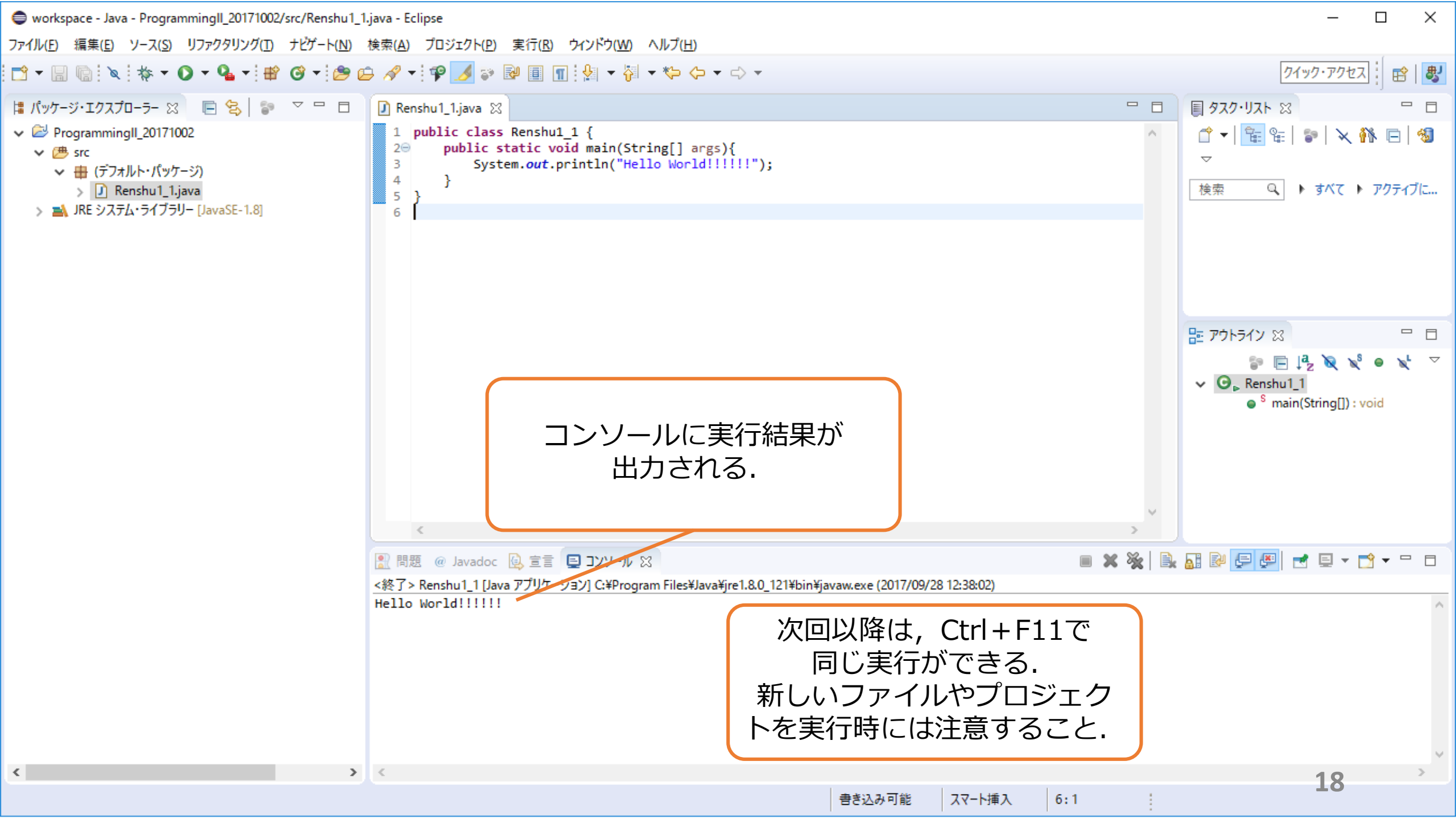
特にコンソールでは, プログラム実行時の出力が表示される。



ソースファイルを右クリック
実行 (R)
1 Javaアプリケーション
を選択し, 実行する.

Hello worldのプログラムを書く.

```
public class Renshu1_1{  
    public static void main(String[] args){  
        System.out.println("Hello World!!!!!!");  
    }  
}
```

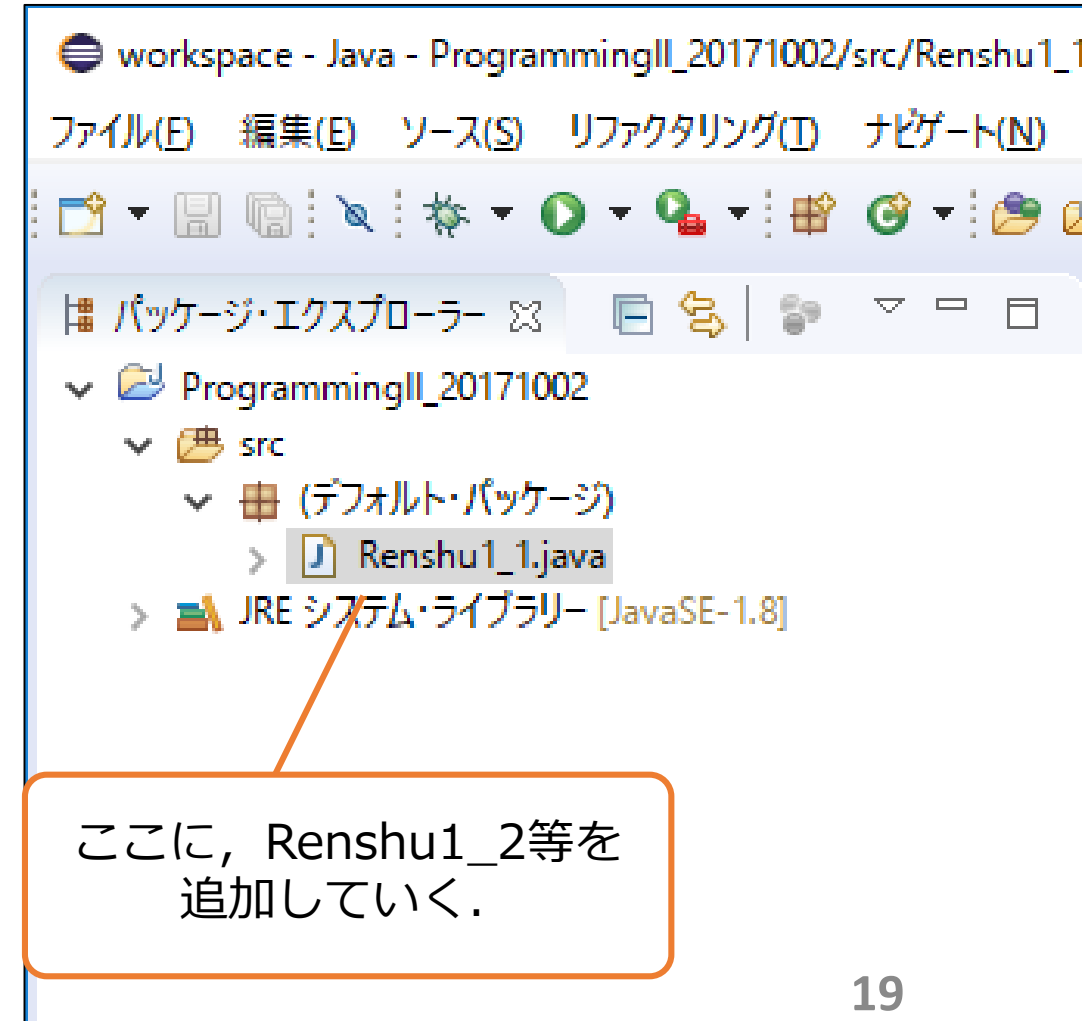


コンソールに実行結果が
出力される.

次回以降は、Ctrl+F11で
同じ実行ができる。
新しいファイルやプロジェクト
を実行時には注意すること。

今後の運用方法

- **プロジェクトファイル**は各回ごとに作成する.
 - 次はProgrammingII_20171016
 - なお, 10月9日は体育の日 (祝日)
- **ソースファイル**は同じプロジェクト内に複数作成していく.
- 元ファイルは**マイドキュメント直下**に保存される.



本日の目次

- 前回のコメントに対する返信
- 開発環境の使い方
- 駆け足だった点の復習
- 課題演習



Java基礎文法

条件分岐 2 (switch)

条件分岐 (switch文) の基礎的な書き方は次のとおりである.

```
switch(変数名){  
  case 定数 1 :  
    //変数名==定数 1 の時の処理  
    break;  
  case 定数 2 :  
    //変数名==定数 2 の時の処理  
    break;  
  default:  
    //変数名がその他の時の処理  
    break;  
}
```

```
if(変数名==定数 1){  
    //変数名==定数 1 の時の処理  
}else if(変数名==定数 2){  
    //変数名==定数 2 の時の処理  
}else{  
    //変数名がその他の時の処理  
}
```

Switchが使えるときには次の条件がある.

- ・ 1種類の変数を取り扱う.
- ・ 比較対象が定数である.
- ・ 使える条件は==のみである.

Java基礎文法

練習問題 1 : Switch

10までの乱数は次のプログラムで実装できる.

```
int rand = (int)(Math.random()*10);
```

では, この乱数が偶数か奇数かを表示するプログラムを Switch文で実装せよ. 出力はrandの値 + (偶数か奇数か)とする.

ヒント: randをそのまま扱うと, Switchのケースを10パターン書かなければならず面倒なので, 偶数奇数を判別するための2値変数 (int flag) あたりを定義し, それに対してSwitchを書くといい.

Java基礎文法

配列 (forと拡張for)

Javaでは拡張for文 (for-each文) という使い方ができる.

```
String[] array = {"a", "i", "u", "e", "o"};  
for(String str : array){  
    System.out.print(str + ",");  
}
```

型名 変数名 : 配列名

という風を書く

a,i,u,e,o,

配列の中身が, 1つずつstrに
格納されてループ内の処理が
実行される.

Java基礎文法

配列 (forと拡張for)

勿論普通のfor文でも書ける.
左右の処理は同義である.

Javaでは拡張for文 (for-each文) という使い方ができる.

```
String[] array = {"a", "i", "u"};  
for(String str : array){  
    System.out.print(str + ",");  
}
```



a,i,u,

```
String[] array = {"a", "i", "u"};  
for(int i=0;i<array.length;i++){  
    String str = array[i];  
    System.out.print(str + ",");  
}
```



a,i,u,

Java基礎文法

練習問題 2 : 拡張for

先ほどの拡張forのサンプルプログラム（下記）を実装し，出力を確認せよ.

```
String[] array = {"a", "i", "u"};
for(String str : array){
    System.out.print(str + ",");
}
```

```
String[] array = {"a", "i", "u"};
for(int i=0;i<array.length;i++){
    String str = array[i];
    System.out.print(str + ",");
}
```

Java基礎文法

配列（多次元配列）

多次元配列は次の書き方で宣言が可能である.

```
String[][] array = new String[2][3]; //2行3列の2次元配列
```

```
System.out.println("array[0][1]:" + array[0][1]);  
System.out.println("array.length:" + array.length);  
System.out.println("array[0].length:" + array[0].length);
```



```
array[0][1]:null  
array.length:2  
array[0].length:3
```

2行3列の2次元配列なのでこんな感じ

array[0][0]	array[0][1]	array[0][2]
array[1][0]	array[1][1]	array[1][2]

Java基礎文法

コマンドライン引数

メインメソッドの引数(`String[] args`)は暗記するとしてきたが、実はコマンドラインから実行した際に送られてくる引数である。

```
public class Sample1{  
    public static void main(String[] args){  
        System.out.println("args.length:" + args.length);  
        int i=0;  
        for(String str : args){  
            System.out.println("args[" + i + "]: " + str);  
            i++;  
        }  
    }  
}
```

このこと

Java基礎文法

コマンドライン引数

コマンドプロンプトから, Javaを実行する時に引数を指定すると次のような動作をする.

```
java Sample1
```

引数なしで実行

0

```
java Sample1 aa ii uu
```

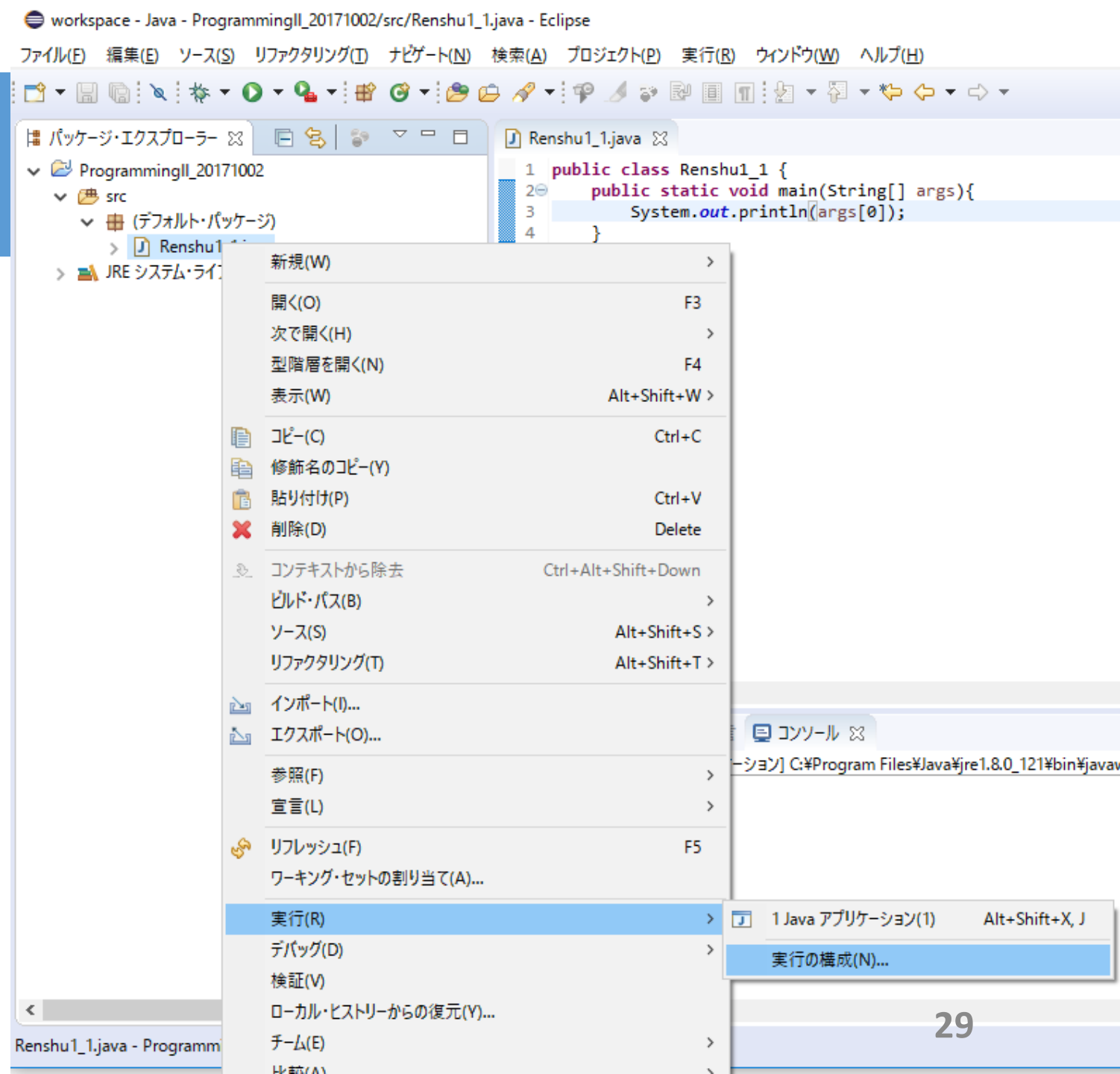
3引数(aa, ii, uu)で実行

```
3  
aa  
ii  
uu
```

Java基礎文法

コマンドライン引数

Eclipseでコマンドライン引数を用いて実行するには、
ソースファイル右クリック
実行 (R)
実行の構成 (N)
を選択し、 , , ,



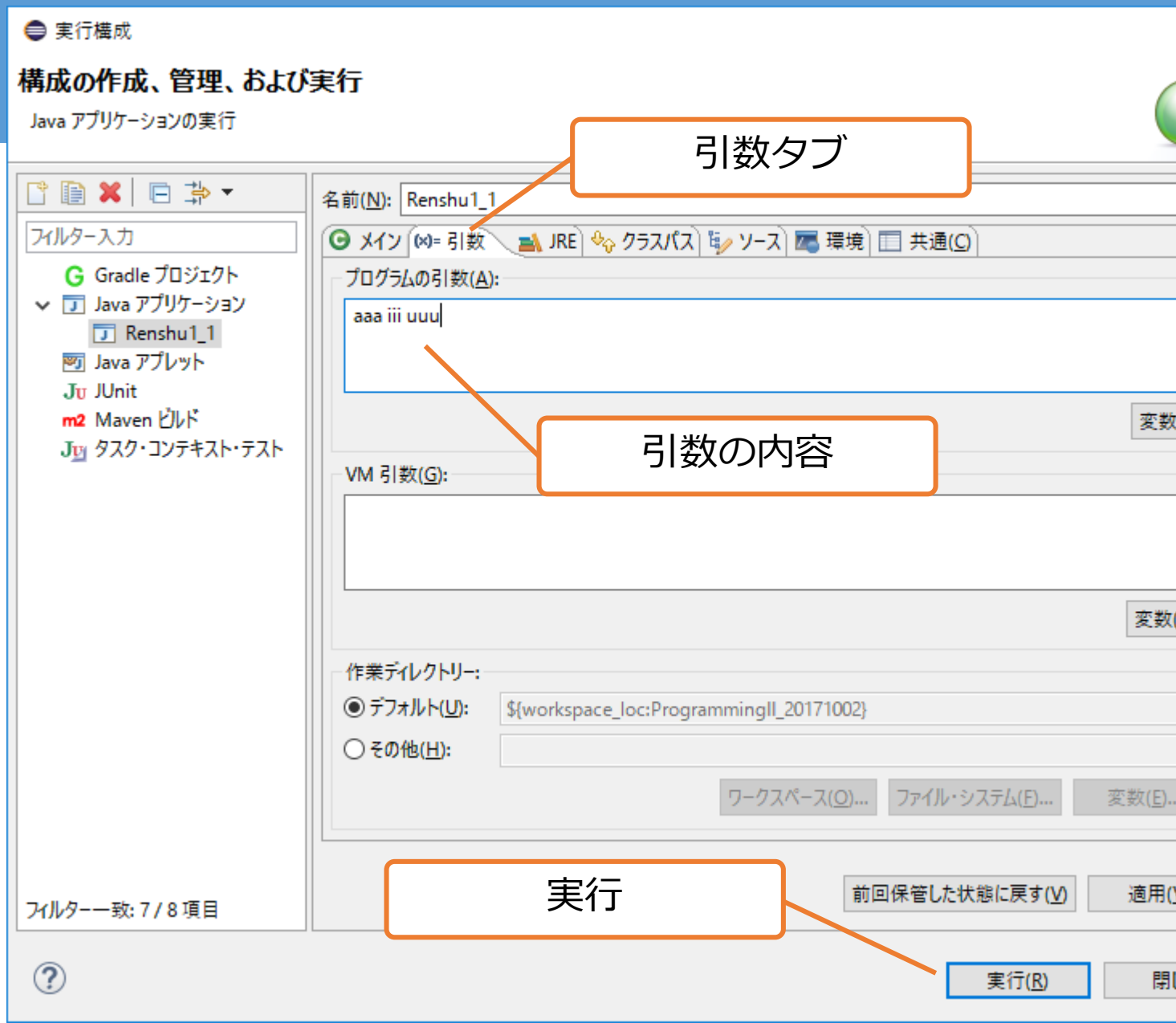
Java基礎文法

コマンドライン引数

引数タブを押下すると、
プログラムの引数 (A)を
設定することができる。

ここで実行を押下すると、
>java Renshu1_1 aaa iii uuu
と同じ動作となる。

args[0]に"aaa"が、
args[1]に"iii"が、
args[2]に"uuu"が
格納された状態でプログラム
が実行される。



Java基礎文法

練習問題 3 : コマンドライン引数

コマンドライン引数のサンプルプログラム(下記)の出力を確認せよ.

```
public class Sample1{  
    public static void main(String[] args){  
        System.out.println("args.length:" + args.length);  
        int i=0;  
        for(String str : args){  
            System.out.println("args[" + i + "]: " + str);  
            i++;  
        }  
    }  
}
```

当然実行時にはEclipseの
実行構成を設定する必要がある.

本日の目次

- 前回のコメントに対する返信
- 開発環境の使い方
- 駆け足だった点の復習
- **課題演習**

本日の提出課題

課題

後ほど出題する演習課題をEclipseで開発し,
ソースコードを提出する.
(作成したファイルの数だけファイルの送信を行う)

資料を公開しているサイトから, 「課題提出」で提出ページに行ける.

<http://hsgw-nas.fuis.u-fukui.ac.jp/lecture.html>

直リンクはこちら

http://hsgw-nas.fuis.u-fukui.ac.jp/lecture_file.html

演習課題のルール

Reidai1.java

ファイル名は指定された通りとする。
ここを間違えると提出後の自動採点で0点となる。

前提条件：

```
int i = 0;
```

前提条件がある場合は、mainメソッドに、
前提条件を記述する必要がある。
ヒントが書いてあることもある。

課題要件：

- コマンドライン引数argsの要素数を調べ、変数iに格納後、出力せよ。

入出力例：

入力：aa ii uu ee oo

出力：5

演習課題で皆さんが編集できるのは、
この青いゾーンのみである。

演習課題のルール

Reidai1.java

前提条件：

```
int i = 0;
```

```
i = args.length;
```

```
System.out.println("" + i);
```

回答例

課題要件：

- コマンドライン引数argsの要素数を調べ、変数iに格納後、表示せよ。

入出力例：

入力：aa ii uu ee oo

出力：5

演習課題のルール

Reidai1.java

```
class Reidai1{  
    public static void main(String[] args){  
        int i = 0;  
        i = args.length;  
        System.out.println("" + i);  
    }  
}
```

ファイル名が指定されているため
当然クラス名も同じとする必要がある。

mainメソッドはあえて前提条件には
明示しないが、当然記述する必要がある。

従って、先ほどの例題は、このような
プログラムを書いたうえで、青いゾーンを
実装するという意味となる。

演習課題 1

_はアンダーバーなので注意

Kadai1_1.java

前提条件：

//配列の長さ

```
int len = args.length;
```

課題要件：

- コマンドライン引数argsは文字列配列である。入力を全て半角カンマで結合して出力せよ。
- 出力の最後の1文字は半角カンマで良い。
- args.lengthが0のことではない。

入出力例：

入力：I have a pen

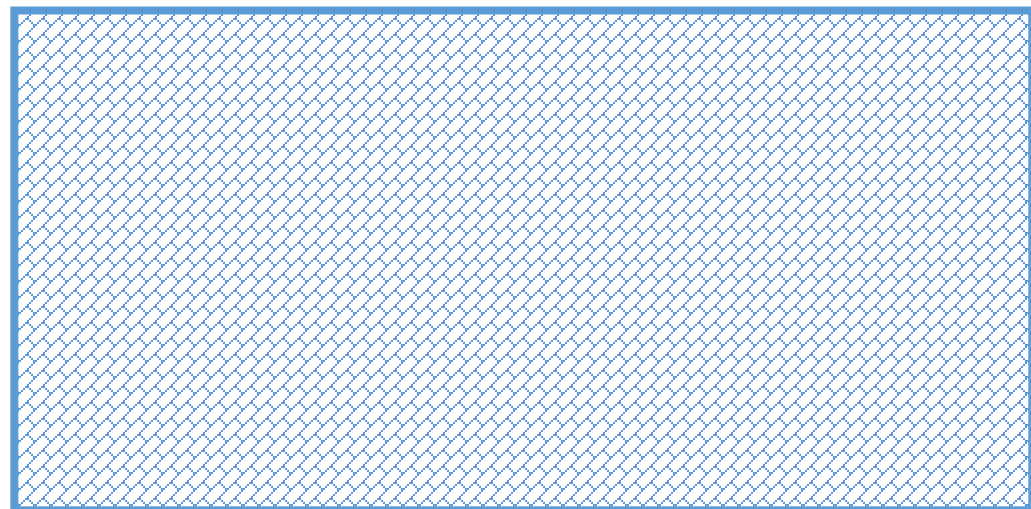
出力：I,have,a,pen,

演習課題 2

Kadai1_2.java

前提条件：

//拡張for文とswitch文を用いる.



課題要件：

- コマンドライン引数argsは文字列{"A", "B", "C", "D"}のいずれかの配列となる。入力に合わせて{"Ant", "Boy", "Cat", "Dog"}と改行区切りで出力せよ。
- args.lengthが0のことではない。

入出力例：

入力：B B D

出力：Boy

Boy

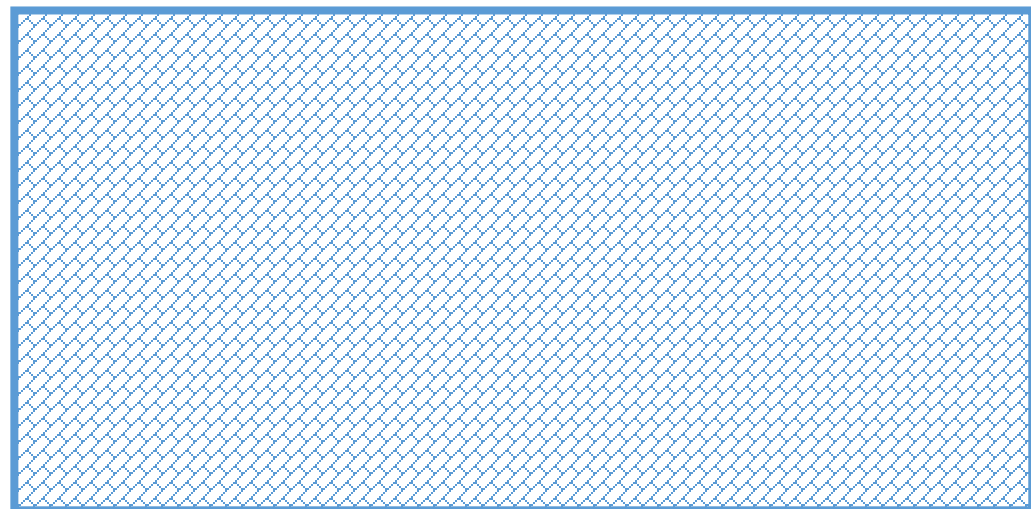
Dog

演習課題 3

Kadai1_3.java

前提条件：

//拡張for文を用いる.



課題要件：

- コマンドライン引数argsの全要素には数値が入る. 全てキャストし, 合計値を出力せよ.
- args.lengthが0のことではない.

入出力例：

入力：10 5 1 2 5

出力：23

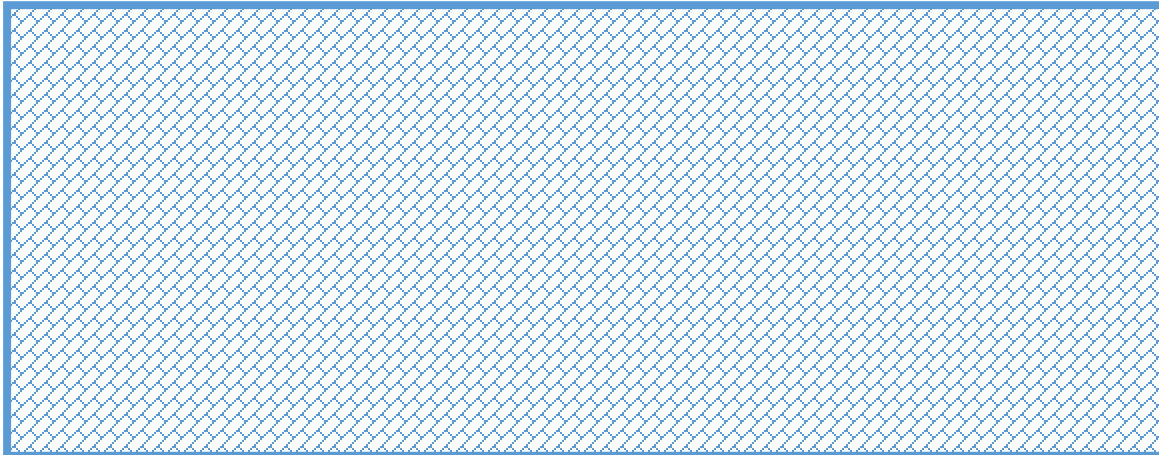
演習課題 4 (発展)

System.out.print("")は最後に改行をしない。
System.out.println("")は最後に改行を行う。

Kadai1_4.java

前提条件：

```
int M = Integer.parseInt(args[0]);  
int N = Integer.parseInt(args[1]);  
int[][] array = new int[M][N];
```



課題要件：

- コマンドライン引数argsは数値MとNが入力される。M行N列の多次元配列"array"を作成し、各要素番号m,nに応じて $(m+1)*(n+1)$ の結果を各要素に格納せよ。また半角スペース区切りと改行を用いて出力せよ。

入出力例：

入力：3 5

出力：1 2 3 4 5

2 4 6 8 10

3 6 9 12 15

次週予告

※次週以降も計算機室

全体

Javaの最も重要な特徴である**オブジェクト指向**の考え方を学ぶ。
クラス、インスタンス、メソッドについても触れる。
資料は事前に公開するので必要ならば印刷してくること。

