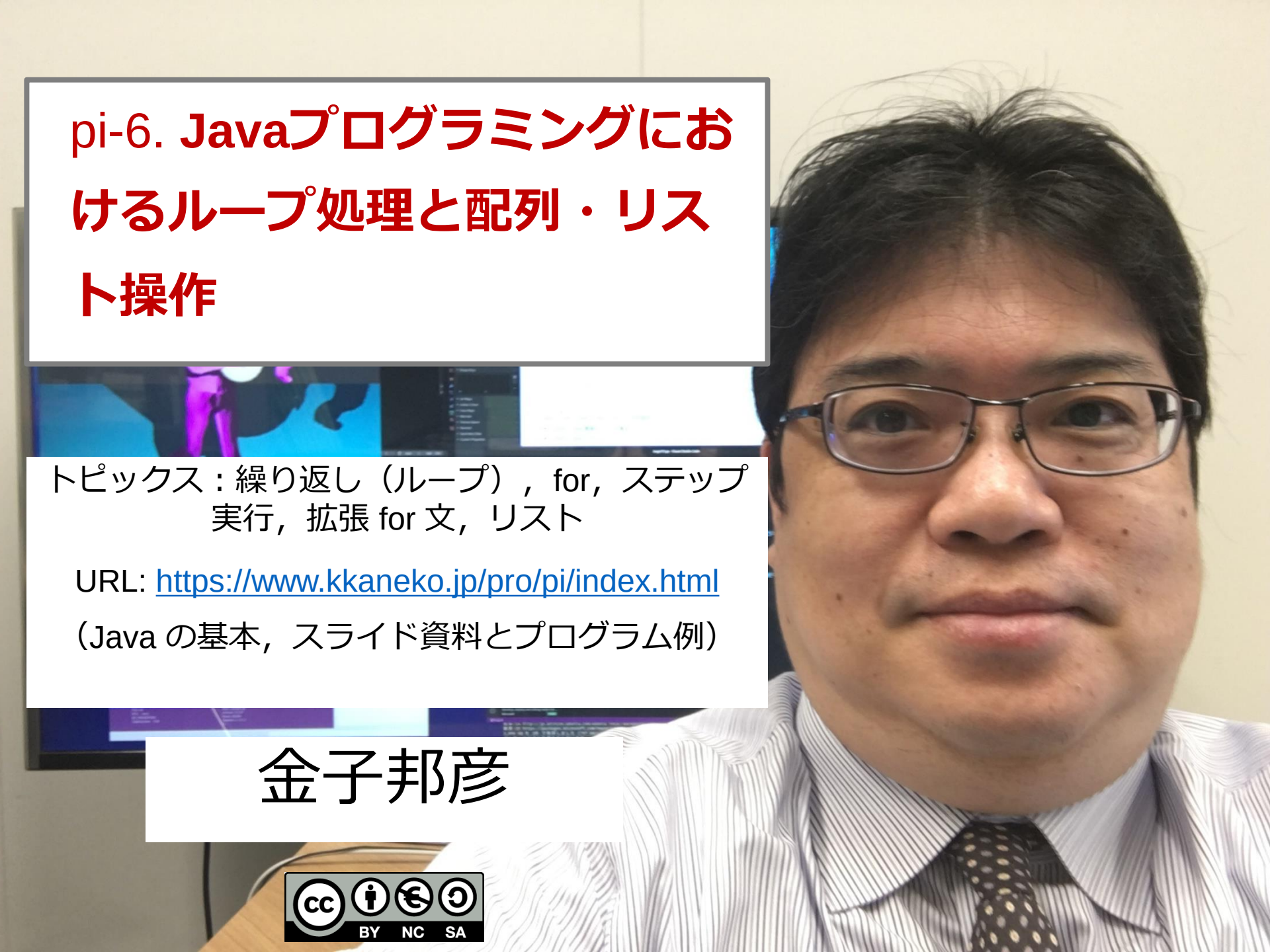
pi-6. Javaプログラミングにおけるループ処理と配列・リスト操作



トピックス：繰り返し（ループ），for，ステップ実行，拡張 for 文，リスト

URL: https://www.kkaneko.jp/pro/pi/index.html

（Java の基本，スライド資料とプログラム例）

金子邦彦

CC BY NC SA

物体の落下距離： **9.8 × (時間)² ÷ 2**

時間は 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10

⇒ 同じ式の計算を 11 回繰り返し

```java
public class YourClassNameHere {
    public static void main(String[] args) {
        double x[] = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10};
        int i;
        for(i = 0; i <= 10; i++) {
            System.out.println((9.8 / 2) * x[i] * x[i]);
        }
    }
}
```

Java プログラム

実行結果

Print output (drag lower right corner to resize)

```
0.0
4.9
19.6
44.1
78.4
122.5
176.4
240.10000000000002
313.6
396.90000000000003
490.0
```

全体まとめ

- for による**繰り返し（ループ）**
  同じ処理や操作を繰り返す

```java
public class YourClassNameHere {
    public static void main(String[] args) {
        double x[] = {10, 20, 30, 40};
        int i;
        for(i = 0; i <= 3; i++) {
            System.out.println(x[i]);
        }
    }
}
```

Java プログラム

```
Print output (drag lower rig
10.0
20.0
30.0
40.0
```

実行結果

- **拡張 for 文**により，**リスト**などのコレクションについて，**繰り返し処理**ができる

アウトライン

番号	項目
	復習
6-1	配列と繰り返し（ループ）
6-2	リストと繰り返し（ループ），拡張 for 文
6-3	マップと繰り返し（ループ），拡張 for 文

各自、資料を読み返したり、課題に取り組んだりも行う

この授業では、**Java** を用いて基礎を学び、マスターする

オブジェクトとメソッド

hero.moveDown()

hero　　オブジェクト
moveDown()　メソッド
間を「.」で**区切って**いる

- **メソッド: オブジェクト**に属する操作や処理.
- **メソッド**呼び出しでは，**引数**を指定することがある．**引数**（ひきすう）は，**メソッド**に渡す値のこと

hero.attack("fence", 36, 26)

Java プログラムの書き方

プログラムの例

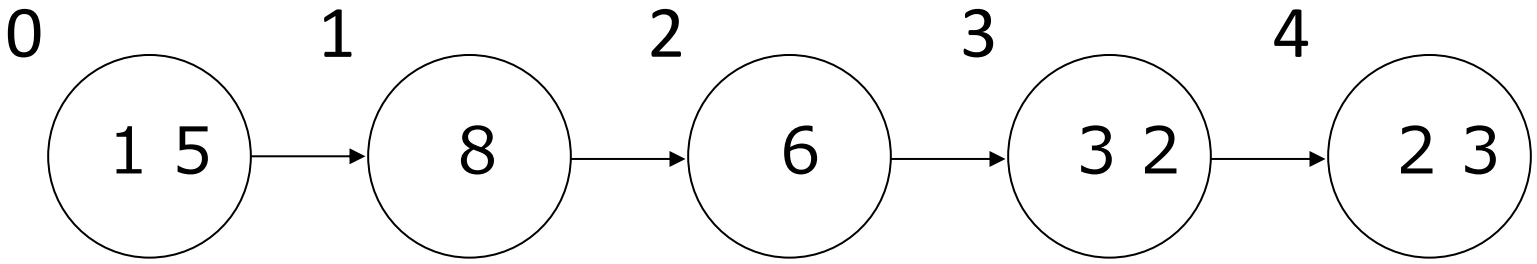
```
x = 100
a = x + 200
enermy1 = hero.findNearestEnemy()
hero.attack(enemy1)
```

- **代入**：**オブジェクト名** ＋ 「**=**」
  ＋ 式または値またはメソッド呼び出し
- **メソッドアクセス**：**オブジェクト名** ＋ 「**.**」
  ＋ **メソッド名** ＋「**()**」 （引数を付けることも）

その他，属性アクセス，関数呼び出し，制御，「*」，「+」などの演算子，コマンド，定義など

リスト

- **リスト**は，**同じ型の要素の並び**
- **リスト**の要素には**順序がある**．０から始まる番号（**添字**）が付いている
- 要素の**削除**，**挿入**により**サイズが増減**する



マップ (Map)

キー	値（バリュー）
1	Red
2	Yellow
3	Blue

- **マップ**は，**キー**と**値（バリュー）**の**ペア**の集まり
- 同じ値の**キー**は**2回以上登場しない**

Java Tutor の起動

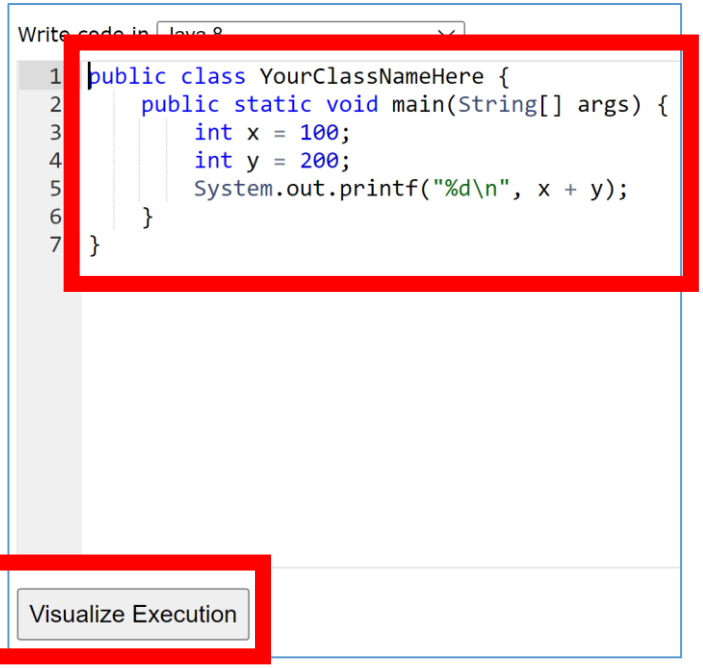
① **ウェブブラウザ**を起動する

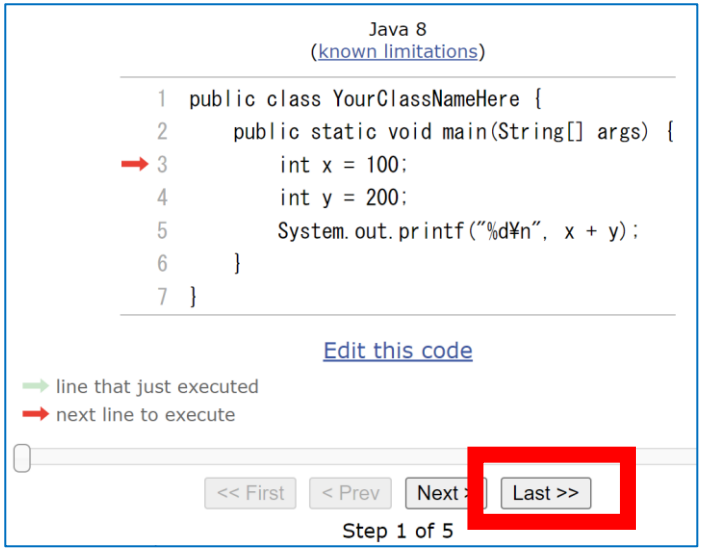
② **Java Tutor** を使いたいので，次の URL を開く

http://www.Javatutor.com/

③ 「**Java**」をクリック　⇒ **編集画面**が開く

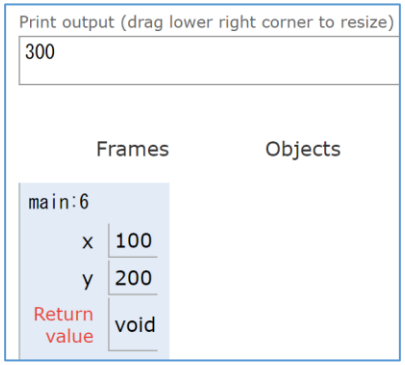
Java Tutor でのプログラム実行手順

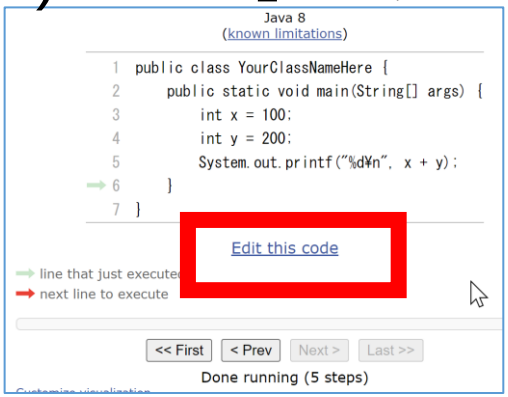




(1)「**Visualize Execution**」をクリックして**実行画面**に切り替える

(2)「**Last**」をクリック.





(3) **<u>実行結果を確認</u>**する.

(4)「**Edit this code**」をクリックして**編集画面**に戻る

Java Tutor 使用上の注意点①

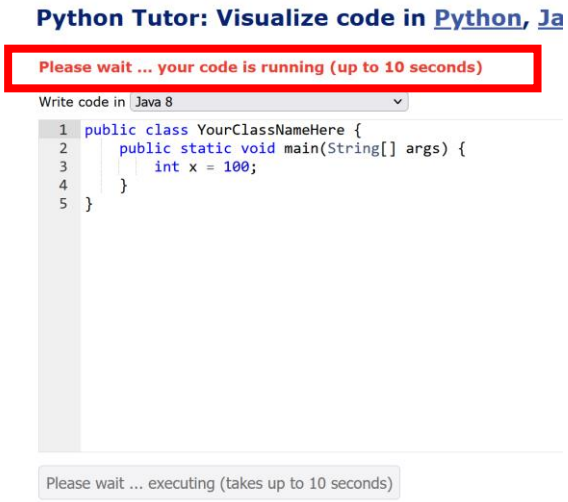
- **実行画面で，次のような赤の表示が出ることがある → 無視してよい**

  過去の文法ミスに関する確認表示

  邪魔なときは「**Close**」

Java Tutor 使用上の注意点②

「**please wait ... executing**」のとき，**１０秒ほど待つ**.

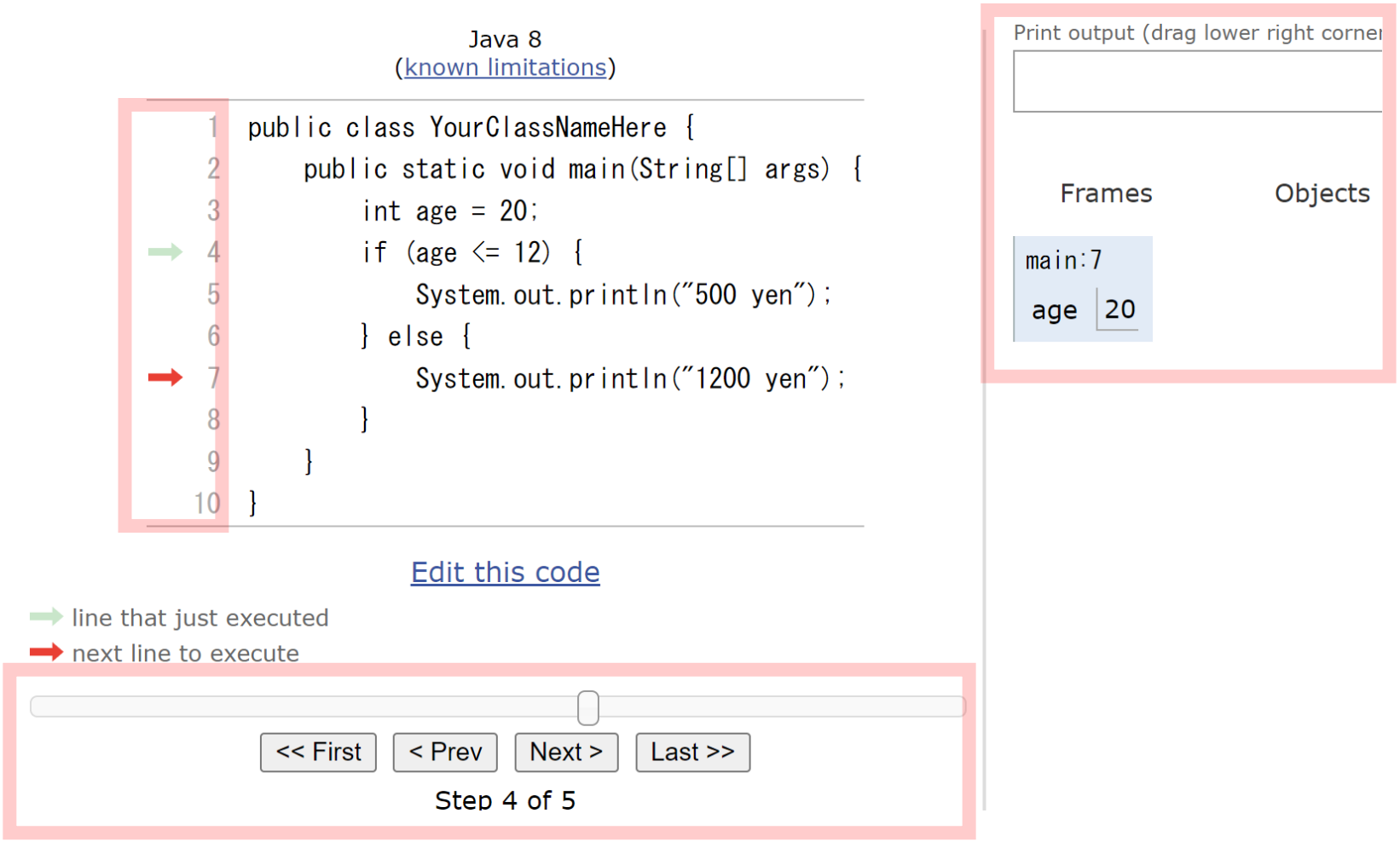


→ **混雑しているとき**は，**「Server Busy・・・」というメッセージが出る**ことがある.

混雑している．**少し（数秒から数十秒）待つ**と自動で表示が変わる（変わらない場合には，操作をもう一度行ってみる）

Java Tutor でのステップ実行

ステップ実行により，**プログラム実行の流れ**を確認できる



6-1. 配列と繰り返し（ループ）

配列

データの並びで，**０から始まる番号（添字）**が付いている

array

0	1	2	3	4	5	6	7	8	9	10	11	12
0	31	28	31	30	31	30	31	31	30	31	30	31

演習

資料：17 ～ 18

【トピックス】

- **配列**

① Java Tutor のエディタで次のプログラムを入れる

月の日数についてのデータを作る。**サイズ 13 の配列**.

※ うるう年のことは考えないことにする

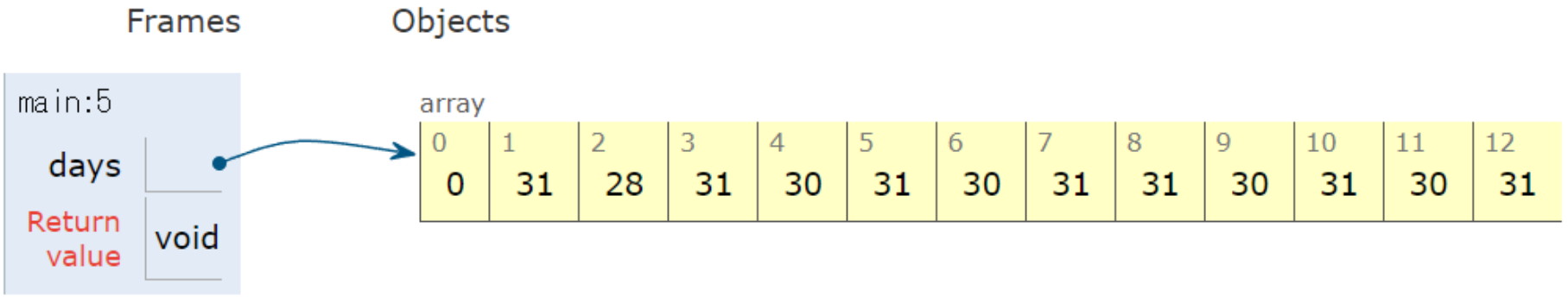
9月について表示させてみて確認も行う.

```
1  public class YourClassNameHere {
2      public static void main(String[] args) {
3          int days[] = {0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};
4          System.out.println(days[9]);
5      }
6  }
```

② 実行し，結果を確認する

Print output (drag lower right corner to resize)

```
30
```



結果の
「30」が表示
されるので確認

「**Visual Execution**」をクリック．そして「**Last**」をクリック．結果を確認．
「**Edit this code**」をクリックすると，エディタの画面に戻る

繰り返し（ループ）

繰り返し（ループ）では，**同じ処理や操作を繰り返す**

配列の中身の表示（要素は４個ある）

array

0	1	2	3
10.0	20.0	30.0	40.0

⇒ 表示を 4 回繰り返し

```
1  public class YourClassNameHere {
2      public static void main(String[] args) {
3          double x[] = {10, 20, 30, 40};
4          int i;
5          for(i = 0; i <= 3; i++) {
6              System.out.println(x[i]);
7          }
8      }
9  }
```

Java プログラム

Print output (drag lower rig

```
10.0
20.0
30.0
40.0
```

実行結果

繰り返しのプログラム例

```
public class YourClassNameHere {
    public static void main(String[] args) {
        double x[] = {10, 20, 30, 40};
        int i;
        for(i = 0; i <= 3; i++) {
            System.out.println(x[i]);
        }
    }
}
```

配列の組み立て

繰り返し

「for(i = 0; i <= 3; i++)」のあとに「{」

字下げを行うことで,
プログラムを読みやすくしている

演習

資料：22 ～ 27

【トピックス】

- **配列**
- **繰り返し（ループ）**
- **for**

① Java Tutor のエディタで次のプログラムを入れる

配列の中身の表示（要素は４個ある）

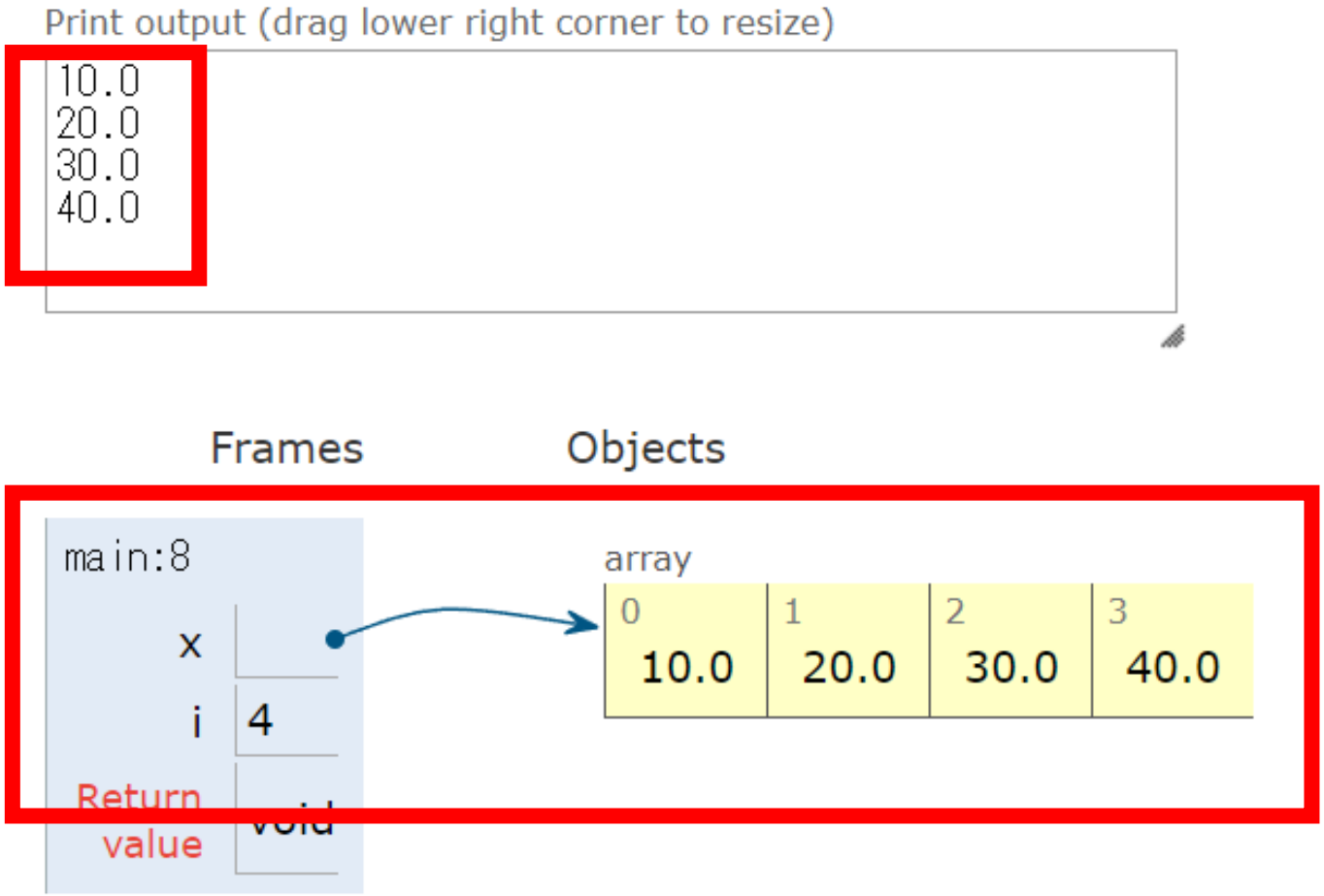
array

0	1	2	3
10.0	20.0	30.0	40.0

⇒ 表示を 4 回繰り返し

```
1  public class YourClassNameHere {
2      public static void main(String[] args) {
3          double x[] = {10, 20, 30, 40};
4          int i;
5          for(i = 0; i <= 3; i++) {
6              System.out.println(x[i]);
7          }
8      }
9  }
```

② 実行し，結果を確認する



結果を確認

「**Visual Execution**」をクリック．そして「**Last**」をクリック．結果を確認．
「**Edit this code**」をクリックすると，エディタの画面に戻る

③ Java Tutor のエディタで次のプログラムを入れる

物体の落下．0〜10秒まで．1秒ごと

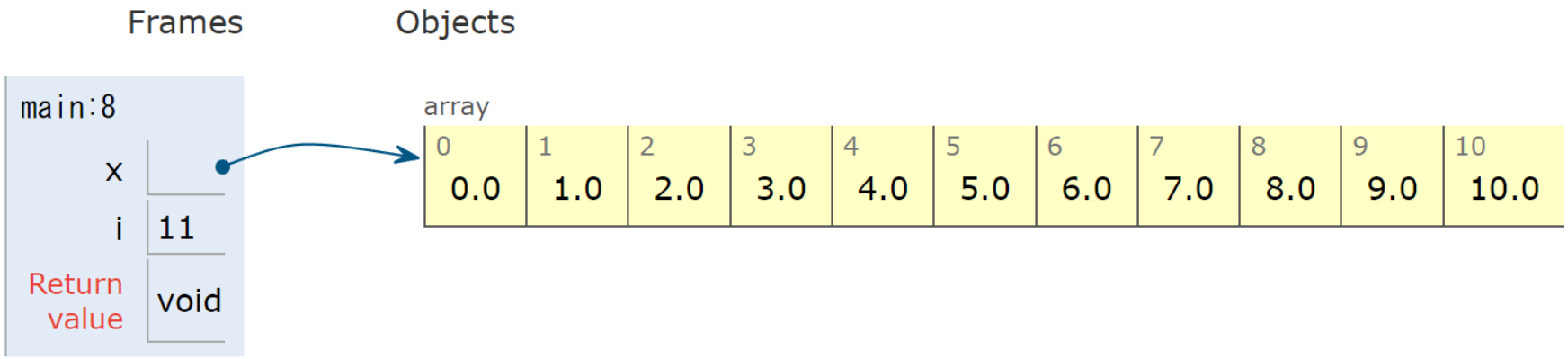
```
1  public class YourClassNameHere {
2      public static void main(String[] args) {
3          double x[] = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10};
4          int i;
5          for(i = 0; i <= 10; i++) {
6              System.out.println((9.8 / 2) * x[i] * x[i]);
7          }
8      }
9  }
```

④ 実行し，結果を確認する

Print output (drag lower right corner to resize)

```
0.0
4.9
19.6
44.1
78.4
122.5
176.4
240.10000000000002
313.6
396.90000000000003
490.0
```

ここをドラッグすると，表示枠が広がる



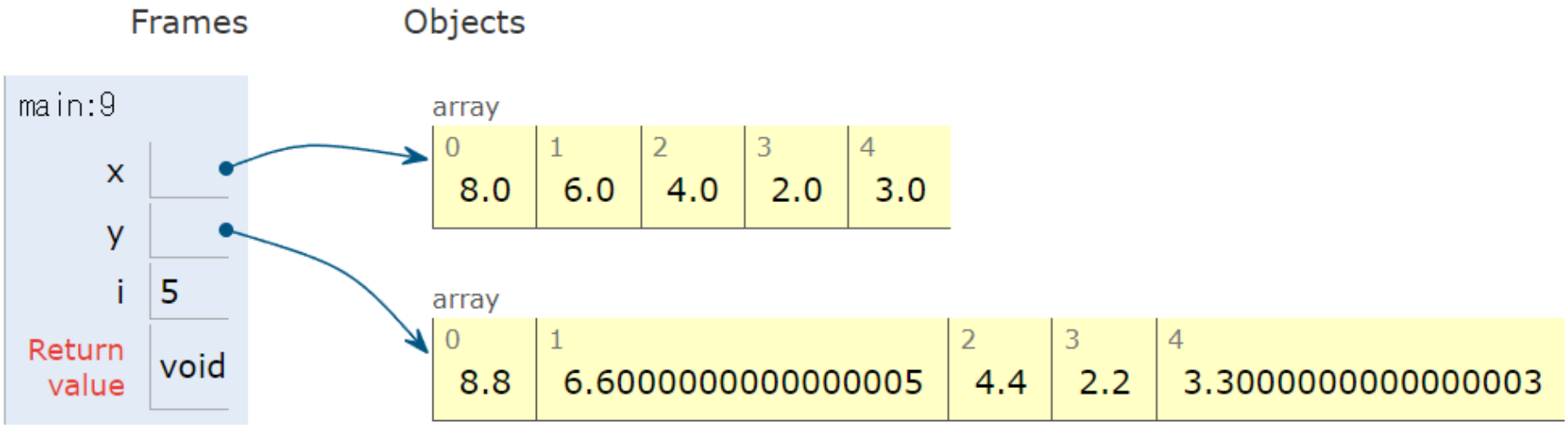
「**Visual Execution**」をクリック．そして「**Last**」をクリック．結果を確認．
「**Edit this code**」をクリックすると，エディタの画面に戻る

⑤ Java Tutor のエディタで次のプログラムを入れる

配列 x のすべての要素の 1.1 倍を y に格納

```
1  public class YourClassNameHere {
2      public static void main(String[] args) {
3          double x[] = {8, 6, 4, 2, 3};
4          double y[] = {0, 0, 0, 0, 0};
5          int i;
6          for(i = 0; i <= 4; i++) {
7              y[i] = x[i] * 1.1;
8          }
9      }
10 }
```

⑥ 実行し，結果を確認する



このプログラムは引き続き使用する．消さずに残しておくこと．

「**Visual Execution**」をクリック．そして「**Last**」をクリック．結果を確認．
「**Edit this code**」をクリックすると，エディタの画面に戻る

演習

資料：29 ～ 30

【トピックス】

- **配列**
- **繰り返し（ループ）**
- **for**
- **ステップ実行**

①「**Step 1 of 21**」と表示されているので，
全部で，ステップ数は 21 あることが分かる
（ステップ数と，プログラムの行数は**違うもの**）

Java

```
 1  public class Main {
 2      public static void main(String[] args)  {
 3          double x[] = {8, 6, 4, 2, 3};
 4          double y[] = {0, 0, 0, 0, 0};
 5          int i;
 6          for(i=0; i<=4; i++){
 7            y[i] = x[i] * 1.1;
 8          }
 9      }
10  }
```

Edit this code

it has just executed
ie to execute

of code to set a breakpoint; use the Back and Forward buttons to jump there.

<< First | < Back | Step 1 of 21 | Forward > | Last >>

② **ステップ実行**したいので，「**Next**」をクリックしながら，**矢印の動き**を確認

※「**Next**」ボタンを何度か押し，それ以上進めなくなったら終了

Java

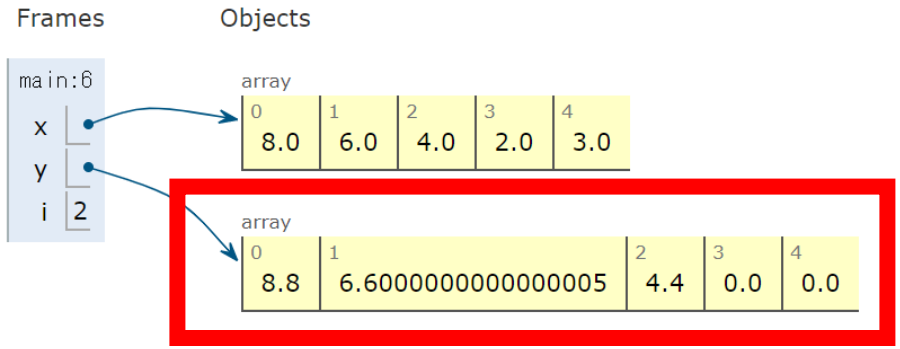
```
1  public class Main {
2      public static void main(String[] args)  {
3          double x[] = {8, 6, 4, 2, 3};
4          double y[] = {0, 0, 0, 0, 0};
5          int i;
6          for(i=0; i<=4; i++){
7            y[i] = x[i] * 1.1;
8          }
9      }
10 }
```

Edit this code

at has just executed
ine to execute

of code to set a breakpoint; use the Back and Forward buttons to jump there.

<< First | < Back | Step 13 of 21 | Forward > | Last >>



実行が進むと，
y の中身が更新される

見どころ
6行目，7行目が
繰り返される

6-2. リストと繰り返し（ループ），拡張 For 文

Java の for 文の種類

- **for 文: 変数の<u>値を変化</u>させるもの**

```
for(i=0; i<=4; i++){
  y[i] = x[i] * 1.1;
}
```

i の値： 0 → 1 → 2 → 3 → 4

- **拡張 for 文: コレクション**（リストやマップなど）の**<u>要素をたどるため</u>**のもの.

```
for(String s : m) {
  System.out.printf("%s\n", s);
}
```

m は、文字列を要素とするコレクションオブジェクト

s の値： “15” → “8” → “6” → “32” → “23”

演習

資料：**34 ～ 35**

【トピックス】

- **リスト**
- **ArrayList**
- **拡張 for 文**

① Java Tutor のエディタで次のプログラムを入れる

拡張 for 文を使ってみる

```
1  import java.util.ArrayList;
2
3  public class YourClassNameHere {
4      public static void main(String[] args) {
5          ArrayList<String> m = new ArrayList<String>();
6          m.add("15");
7          m.add("8");
8          m.add("6");
9          m.add("32");
10         m.add("23");
11         for(String s: m) {
12             System.out.println(s);
13         }
14     }
15 }
```

② 実行し，結果を確認する

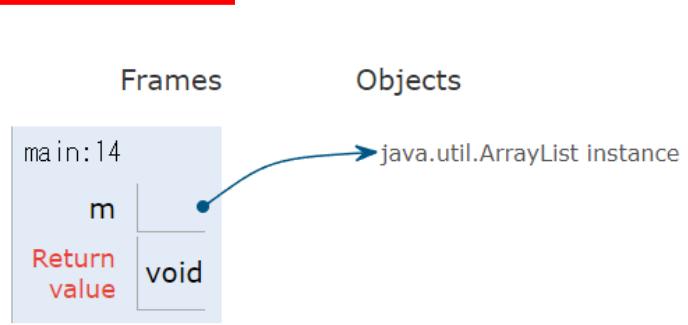
Java 8
(known limitations)

```
1  import java.util.ArrayList;
2
3  public class YourClassNameHere {
4      public static void main(String[] args) {
5          ArrayList<String> m = new ArrayList<String>();
6          m.add("15");
7          m.add("8");
8          m.add("6");
9          m.add("32");
10         m.add("23");
11         for(String s: m) {
12             System.out.println(s);
13         }
14     }
15 }
```

Edit this code

Print output (drag ower right corner to resize)

```
15
8
6
32
23
```



「**Visual Execution**」をクリック．そして「**Last**」をクリック．結果を確認．
「**Edit this code**」をクリックすると，エディタの画面に戻る

リストの生成，要素の挿入，拡張 for文

- **リストの生成**

```
ArrayList<String> m = new ArrayList<String>();
```

m: オブジェクト名　　Stringは，
要素の**クラス**名

- **要素の挿入**

```
m.add("15");
```

m: オブジェクト名，add: メソッド，"15": 挿入したい要素

- **拡張 for 文**

```
for(String s: m) {
    System.out.println(s);
}
```

全体まとめ

- **基本データ型** ：**整数**，**浮動小数**，**文字**，**boolean** (**true/false**を扱う)
- **配列**は，**データの並び**で，**０から始まる番号（添字）**が付いている
- **条件分岐**
- **繰り返し（ループ）**

全体まとめ

- for による**繰り返し（ループ）**
  同じ処理や操作を繰り返す

```java
public class YourClassNameHere {
    public static void main(String[] args) {
        double x[] = {10, 20, 30, 40};
        int i;
        for(i = 0; i <= 3; i++) {
            System.out.println(x[i]);
        }
    }
}
```

Java プログラム

```
Print output (drag lower rig
10.0
20.0
30.0
40.0
```

実行結果

- **拡張 for 文**により，**リスト**などのコレクションについて，**繰り返し処理**ができる

関連ページ

- **Java プログラミング入門**

GDB online を使用

https://www.kkaneko.jp/pro/ji/index.html

- **Java の基本**

Java Tutor, GDB online を使用

https://www.kkaneko.jp/pro/pi/index.html

- **Java プログラム例**

https://www.kkaneko.jp/pro/java/index.html

資料中のソースコード 6-1

```
public class YourClassNameHere {
    public static void main(String[] args) {
        int days[] = {0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};
        System.out.println(days[9]);
    }
}
```

資料中のソースコード 6-1

```
public class YourClassNameHere {
    public static void main(String[] args) {
        double x[] = {10, 20, 30, 40};
        int i;
        for(i = 0; i <= 3; i++) {
            System.out.println(x[i]);
        }
    }
}
```

資料中のソースコード 6-1

```
public class YourClassNameHere {
    public static void main(String[] args) {
        double x[] = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10};
        int i;
        for(i = 0; i <= 10; i++) {
            System.out.println((9.8 / 2) * x[i] * x[i]);
        }
    }
}
```

資料中のソースコード 6-1

```
public class YourClassNameHere {
    public static void main(String[] args) {
        double x[] = {8, 6, 4, 2, 3};
        double y[] = {0, 0, 0, 0, 0};
        int i;
        for(i = 0; i <= 4; i++) {
            y[i] = x[i] * 1.1;
        }
    }
}
```

資料中のソースコード 6-2

```
import java.util.ArrayList;

public class YourClassNameHere {
    public static void main(String[] args) {
        ArrayList<String> m = new ArrayList<String>();
        m.add("15");
        m.add("8");
        m.add("6");
        m.add("32");
        m.add("23");
        for(String s: m) {
            System.out.println(s);
        }
    }
}
```