社団法人 情報処理学会　研究報告
IPSJ SIG Technical Report

2003－CE－71　(3)
2003／10／17

Nigari - Java 言語へも移行しやすい初学者向けプログラミング言語

長　慎也[†]　甲斐宗徳[††]　川合　晶[†]
日野孝昭[†]　前島真一[†]　筧　捷彦[†]

大学のプログラミングの授業では，Java などの既存のプログラミング言語を題材とすることが多い．それらは習得が難しく，プログラミング嫌いを生む原因となる．初学者には，興味を惹く例題がすぐに書け，それでいて既存言語へ容易に移行できるプログラミング言語と環境を用いるのが適切である．そこで，オブジェクト指向のプログラミングにすぐ入れて，なお Java への移行も容易な言語 Nigari とそのプログラミング環境を実際に開発して，授業に用いてみた．本発表では早稲田大学コンピュータ・ネットワーク工学科での，その試行結果を報告する．

Nigari - a programming language for novice students - easy to learn OOP fundamentals leading to Java

Shinya CHO,[†] Munenori KAI,[††] Akira KAWAI,[†]
Takaaki HINO,[†] Shin'ichi MAESHIMA[†]
and Katsuhiko KAKEHI [†]

Programming lessons in universities start in a language widely used in practice, such as Java, of which richness in syntax and semantics tends to be rather an obstacle for students to learn and enjoy programming. We developed a programming language Nigari and its supporting system to present students an environment where they learn OOP fundamentals by writing attractive examples to be visualized on the screen from the beginning. Nigari is designed to lead to writing programs in Java. We report on our class experiment of applying Nigari and its system at the CS department in Waseda University.

1. Java を用いたプログラミングの授業の現状

情報科学の授業は，まずプログラミングを教えることから始まるが，そこで使われる言語は，Java 言語 (以下，Java) などの既存の言語を使用することが多い．これは世の中の実情に対応した教育を行うという目的からであるが，このような言語はプログラムに関する知識をもっていない初学者には敷居が高すぎる，という問題がある．

Java も，次のような点で，初学者にとって扱いづらいものとなっている．

- **多数の「おまじない」の存在**
  どんな短いプログラムでも `class`，`import`，`public`，`static` などの予約語を伴う宣言を書かなければならない．しかし，それらの予約語が必要となる理由を説明するには多くの概念と知識が必要となるため，「とにかく書くと覚えよ.」という以外はない．こうして，プログラムが「おまじない」と化してしまう．
- **コンパイルエラーの頻発**
  Java は，変数宣言の不備や初期化のし忘れなどで多くのエラーを発生させる上に，そのメッセージも初学者には難解なものになる．このため学習者は，プログラムの実行がなかなか行えず，挫折感を味わうことになる．
- **コンソールアプリケーションばかりの例題**
  グラフィックスを用いた Java のアプリケーションは，その作成に AWT などの多くの知識が必要であり，初学者には作らせるのが難しい．このため，

† 早稲田大学 理工学研究科
Graduate School of Science and Engineering, Waseda Uniersity
†† 成蹊大学工学部 経営・情報工学科
Department of Industrial Engineering and Information Sciences, Seikei Uniersity

作らせるプログラムが単純な計算などを行うコンソールアプリケーションに限定されてしまい，学習者の興味を惹くのが難しい．

- **オブジェクト指向を体得できない例題**
  Java の初学者向けの授業で実際に作らせるプログラムは，一切インスタンスを作らず，クラス 1 個だけを定義し，すべて static な変数やメソッドを用いたプログラムにものから始まることが多い[1]．加えて，以降に作らせるプログラムもオブジェクト指向を必要としないものに終始してしまいがちである．オブジェクト指向については単なる説明を重ねるだけで，その概念を体得できるような学習ができていない．

こうした状況を改善する試みとして，オブジェクト指向のプログラミング入門用に，Java を簡素化した言語仕様を持つプログラミング言語 Nigari とその環境 Nigari System[2] を開発した．これを Java の入門として使い，その後 Java に移行する方法を考え，早稲田大学のコンピュータ・ネットワーク工学科 1 年生に対するプログラミングの授業で実施してみた．

2. 関 連 研 究

簡素化した言語仕様をもつプログラム言語を作成しプログラミングの学習者に適用した研究は，これまでにも行われてきている[3][4][5][6]．

しかし，ドリトル[3] や NB2[5] などの言語の仕様は，プログラムの読みやすさ，書きやすさを優先したために，Java などの既存の言語とはまったく異なる仕様となっている．このため，学習者は，Java 言語を学習する際に，それ以前に学習した言語との違いに戸惑いを覚えることになる．実際，LOGO のような初心者向けの言語を習得させても，C 言語の理解の助けには必ずしもならなかった，という報告[4] がある．

また若葉[6] など，Java 言語に近い仕様をもった，教育目的の言語もあるが，どちらかというと C 言語のような手続き型言語の方に近く，作成できるアプリケーションもコンソールアプリケーションにとどまっている．

3. Nigari の概要

プログラム言語 Nigari とそのプログラミング環境である Nigari System は，次のような特徴をもっている．

- **「おまじない」の記述が不要な仕組み**
  Nigari は，Java と比べてクラスやメソッドの宣言が簡単で，いわゆる「おまじない」のような予約語を極力書かなくてよいようになっている．
- **プログラムの可視化**
  Nigari System には，ユーザ (学習者) が，特別なグラフィックス命令を書かなくても自動的にプログラムの動きを可視化するような機能が組み込まれている．
  ユーザは，プログラムの流れを直感的に理解できるし，興味惹かれるアプリケーションを簡単に作ることができる．
- **オブジェクト指向の理解の支援**
  Nigari System でのプログラミングは，グラフィックス画面に現れる画像を「オブジェクト」として設定し，そのオブジェクトの振る舞いを記述する，という流れで行っていくので，学習者がオブジェクト指向を強く意識できるようになっている．

4. プログラミング環境 Nigari System

まず，言語 Nigari のプログラミング環境である Nigari System について述べる．

4.1 設計時と実行時

Nigari System には「設計時」と「実行時」の 2 つの状態がある．設計時とは，プログラムを実行する前の状態で，ユーザは，プログラムの編集やオブジェクトの配置を行う．一方，実行時は，設計時に作成したプログラムを実行している状態である．

これらの状態の切り替えはユーザが任意のタイミングで行える．

4.2 ユーザインターフェース

図 1 は，Nigari System に現れるウィンドウ群を例示している．

- **メインウィンドウ**
  オブジェクトを表示するためのウィンドウで，設計時，実行時によって異なる用途がある．
  設計時には，オブジェクトの配置を行い，実行開始時の画面のレイアウトを編集するのに用いる．配置されている各オブジェクトをダブルクリックすると，エディタのウィンドウが開いてそのオブジェクトの実行時の動作を表すプログラムを編集できるようになる．
  実行時には，メインウィンドウ上の各オブジェクトが，設計時に書かれたプログラムに応じて並行に処理を行い，その結果を表示する．
  各オブジェクトは `x`，`y`，`p` というオブジェクト変数 (5.3) を持つ．実行時，設計時いずれの場合も，それらの変数の値に基づいて，オブジェクト自身がメインウィンドウ上に表示される．ここで，`x`

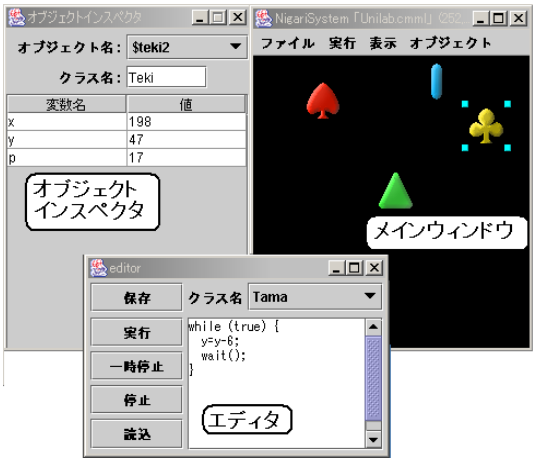


図 1 Nigari のスクリーンショット

はオブジェクトの x 座標，y は y 座標，p は絵柄の番号である．0〜39 の番号で 40 種類の絵柄が用意されている．

- **エディタ**
  オブジェクトの動作記述 (正確にはそのオブジェクトのクラスの記述) であるプログラムを編集するのに使う．
- **オブジェクトインスペクタ**
  特定のオブジェクトが持つ変数の値を閲覧・変更するのに使う．どのオブジェクトを対象とするかはメインウィンドウ上でオブジェクトをクリックして選択する．設計時にも実行時にも使えるが，特に実行時には，選択されたオブジェクトの変数の値の変化がリアルタイムに表示される．

5. Nigari 言語仕様

5.1 プログラミングスタイルと言語仕様

Nigari では，各オブジェクトの個別の動作をプログラムとして記述する．各オブジェクトは，実行時にそれぞれそのプログラムを並行に実行する．

この特性にあわせて，プログラムを記述するソースファイルには，オブジェクトの動作だけを書けばよく，動作と直接関係ないものは極力書かずに済むようにしてある．

5.2 ソースファイルの構造

5.2.1 クラスの宣言

オブジェクトの動作を表すものをクラスと呼ぶ．クラスの宣言については，次のような規則を設けた．

- **1 個のソースファイルには，ただ 1 個のクラスの内容を記述する．**
- **記述されるクラスの名前はファイル名と同じである．**

この規則によれば，ファイル名からクラス名が一意に決定できるので，クラス名をファイル本体に書く必要がない．また 1 つのファイルに複数のクラスを書くこともないので，クラスの境界を示すための特別な区切りもいらない．このため，クラスの宣言であることを示す特別な構文を必要としない．

5.2.2 メ イ ン 文

もっとも単純なソースファイルは，文を並べることで構成される．この文の並びをメイン文と呼ぶ．メイン文は，オブジェクトが実行時に行う動作を表す．なお，ソースファイルには，文以外にメソッドの定義も記述できる (5.4)．逆にメソッドの定義が必要がなければ，ソースファイルには文だけを書けばよいことになる．

5.3 変　　数

Java と異なり，どの変数にもあらゆるデータ型の値を代入することができる．

各オブジェクトに固有の変数を，オブジェクト変数という．オブジェクト変数は，そのオブジェクトからだけ直接に参照が可能である．

どのオブジェクトからも直接に参照できる共通な変数を，グローバル変数という．グローバル変数の名前は先頭に`$`がついたものに限る．

オブジェクト変数もグローバル変数も，変数宣言なしで使う．

5.4 メソッドの定義とローカル変数

メイン文に対するサブルーチンとして，メソッドを定義することができる．メソッドの定義は，ソースファイルの任意の場所に書くことができる．

メソッドへ渡された引数の値を保持するための変数 (仮引数) は，メソッドにローカルな変数として扱われる．また，メソッド内では，ローカル変数を明示的に宣言して使うことも可能である．

5.5 オブジェクト指向の実現

Nigari には，オブジェクト指向プログラミングを可能にするための，次のような機能が用意してある．

5.5.1 間 接 参 照

あるオブジェクトから他のオブジェクトの変数を参照したり，他のオブジェクトのメソッドを呼び出すことができる．間接参照は，Java と同様に演算子 “.” を用いて行う．

5.5.2 オブジェクトの動的生成

オブジェクトは，設計時に (間接的に生成して) 配置するほかに，実行時にプログラムで生成することもできる．生成には，Java と同様に `new` 演算子を用いる．

実行時に生成されたオブジェクトも，他のオブジェクトと一緒になって，並行にそのプログラムを実行しその結果を表示する．

5.6 組み込みメソッド

マウス入力・キーボード入力・グラフィックス描画を行うメソッドや，標準的な数学関数などが用意されている．その他にも次のようなメソッドがある．

- `wait`
  文字通りの処理を待機させるという働きの他に，他のオブジェクトに処理を譲るという働きを併せ持っている．詳しい働きは 6 で述べる．
- `die`
  オブジェクトを画面から消去し，オブジェクトに割り当てられたスレッドを破棄する．

6. 実行系 (Nigari VM)

Nigari のプログラムは Nigari VM によって実行される．Nigari VM は，実行開始にあたって，設計時に配置された各オブジェクトにスレッドを 1 個ずつ割り当てる．また，`new` で生成されたオブジェクトにもスレッドを 1 個割り当てる．スレッドは，割り当てられたオブジェクトのクラスに記述されたメイン文をそれぞれ並行に実行する．`wait` メソッドの呼出しが起きるか，ある一定数の命令を実行するかしたときに，他のスレッドに処理を譲る．スレッドは，メイン文の終わりに達するか，`die` メソッドの呼出しが起きるかしたときに，消滅する．

7. プログラム例

7.1 変数の値によるアニメーション

図 2 に，Nigari のプログラムの一例を示す．このプログラムを実行したオブジェクトは，オブジェクト変数 `x` の値を 5 ずつ増やし，組み込みメソッドである `wait` メソッドを呼ぶ，という動作を繰り返す．このオブジェクトは，Nigari System のメインウィンドウ上では，右に 5 ドット移動し，少し (約 10 ミリ秒) 待つといった動作の繰返しとして観察される．これは，4.2 のメインウィンドウの項目で説明したように，変数 `x`，`y`，`p` の値に応じてそのオブジェクトが自動的にメインウィンドウ上に表示されるためである．

7.2 マウス入力と if 文

図 3 に示したプログラムは，オブジェクトが，自分自身と，マウスカーソルの x 座標の位置関係によって，左右に移動する (マウスカーソルのある方向に寄ってくる) プログラムである．組み込みメソッドの `getMouseX` を用いて，マウスの位置と自分の変数 `x` の値を大小比較し，比較結果に応じて動作を変化させている．

```
while(x<300) {
  x=x+5;
  wait(10);
}
```

図 2 Nigari のサンプルプログラム (1)

```
while(true){
  if(x<getMouseX()) x=x+1;
  if(x>getMouseX()) x=x-1;
  wait(10);
}
```

図 3 Nigari のサンプルプログラム (2)

```
while (y<180) {
  if (x>$player.x) x=x-1;
  if (x<$player.x) x=x+1;
  y=y+1;
  wait(10);
}
```

図 4 Nigari のサンプルプログラム (3)

7.3 他のオブジェクトの参照

図 4 に示したプログラムは，このプログラムで動作するオブジェクトとは別のオブジェクト `$player` を追跡するプログラムである．設計時に作成されたオブジェクトは，ここで示した `$player` のように，グローバル変数を用いて参照できる．どの変数名を使うかは，オブジェクトを作成するときにユーザが設定する．

また，`$player.x` は，オブジェクト `$player` が持つ変数 `x` を間接参照している．

7.4 メソッドの定義

図 5 に示したプログラムで動作しているオブジェクトは，別のオブジェクト `$player` との衝突を検知すると，自分自身を消去する．ここでは，`diff` というメソッドが定義されている，これは 2 つの引数の値の差を非負整数で返すメソッドである．これを用いて，自分と `$player` の x,y 座標を比較し，x,y ともに差が 20 未満であれば `die` メソッドにより消去を行っている．

7.5 オブジェクトの動的生成

図 6 に示したプログラムで動作しているオブジェクトは，マウスボタンが押されるたびに，新しく `Tama` クラスのオブジェクトを生成する．生成したオブジェクトを変数 t に代入し，間接参照を用いて，新しいオブジェクトの位置を，自分と同じ位置に設定している．

```
while(true) {
  x=x+1;
  if (diff(x,$player.x)<20 &&
      diff(y,$player.y)<20) {
    die();
  }
  wait();
}
function diff(a,b) {
  if (a>b) return a-b;
  return b-a;
}
```

図 5 Nigari のサンプルプログラム (4)

```
while (true) {
  if(getKey(1)==1){
    t=new Tama();
    t.x=x;t.y=y;
  }
  wait(10);
}
```

図 6 Nigari のサンプルプログラム (5)

8. 実　　装

Nigari System は，プラットフォームによらず使えることを目指して，Java で実装した．実際，Linux (Vine 2.5) でも Windows(2000/Me/XP) でも，JDK がインストールしてあれば動作する．さらに，初心者でも簡単にインストールできるように，インストーラ付きで配布している．

授業利用に対しては，Nigari 専用のファイルサーバを用意し，学生が Nigari で作成したプログラムを自動的にアップロードする仕組みを設けて，学習履歴が追跡できるように配慮した．

9. 実　　験

本システムを，実際の大学の授業で使用した結果を報告する．

9.1 授業の概要

- **授業名:** プログラミング A
- **対象学科・学年:** 早稲田大学理工学部，コンピュータネットワーク工学科 (CS 学科)1 年
- **目標:** Java 言語の基礎の習得
- **期間，回数:** 2003 年 4 月 14〜同年 7 月 7 日，12 回
- **一回あたりの授業時間:** 2 時限 (約 3 時間)

4/14	講義概要
4/21	PC の使い方とプログラミング
4/28	Nigari のインストールと試用
5/12	変数 / while 文
5/19	while 文 (つづき) / if 文
5/26	if 文 (つづき) / マウス入力
6/02	複数のオブジェクト/マルチスレッド
6/09	メソッド / オブジェクトの実行時生成 (6/16 から Java を使用)
6/16	Java の概要 / 変数の宣言 / while 文
6/23	制御構造/メソッド/文字列
6/30	配列 / コマンドライン引数 / 並べ換え
7/07	並べ換え (つづき) / 文字入力
7/14	(補講) クラス試験

図 7 授業内容 (クラス 1)

- **人数:** 約 240 人
- **教科書:** 「Java 言語プログラミングレッスン (上)」[7)]
- **試験:** クラス試験，共通試験 1 回ずつ．クラス試験はクラスによって内容が異なる．共通試験は，Web を用いて行った．
- **授業形態:** 各自ノートパソコン持参，授業 1 回ごとに，説明を受け，演習を行う
- **クラス構成:** 3 クラスに分かれて授業．授業用資料や課題は担当教員により異なる．クラス分けは各学生の出席番号を 3 で割った余りで決めた．

9.2 実験の実施方法

3 つのクラスのうち，クラス 1 は Nigari System を利用し，クラス 2，3 については利用しなかった．1 クラスの授業内容を図 7 に示す．

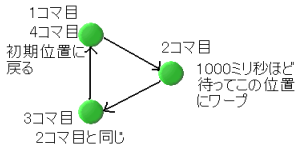
クラス 1 は，6 月 9 日の授業までは Nigari System を利用し，6 月 16 日以降に Java の説明と実習を行った．Nigari を使った授業の実習において出題した演習問題の一部を図 8 に示す．

9.3 アンケート

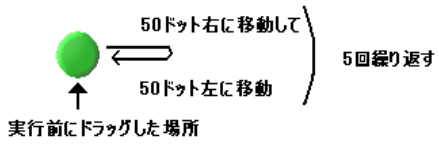
次のような調査をアンケートにて行った．アンケートはすべて Web にて回答する方式をとった．

- **能力調査**
  コンピュータの用途，使用頻度，プログラミング経験の有無などを，自己申告で回答してもらった．クラス 1 は第 2 回 (4 月 21 日)，そのほかのクラスは 5 月中旬頃に回答した．これは，パソコンの操作方法を習う時期がクラスによって異なり，Web を使ってアンケートに答えられるまでの時間に差が出たためである．

5/12(変数) 次の図のように，三角形の頂点を右回りで移動して最初の場所に戻る 4 コマのアニメーションを作りましょう．



5/19(while 文) 次の図のように，左右に 5 往復するオブジェクトを作りましょう．



6/2(複数のオブジェクト) 2 つのオブジェクトが万有引力の法則に従って運動する様子をシミュレートしましょう．

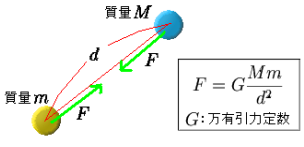


図 8 演習問題 (抜粋)

- **毎回の授業評価 (4 月 28-6 月 30 日)**
  クラス 1 では，毎回の授業の出席点呼の代わりに，「授業の分量」「授業の難易度」「授業の楽しさ」「演習問題の達成度」の 4 項目を回答してもらった．
- **総合評価 (7 月 7 日)**
  授業全体を通して，理解度や感想を問うアンケートを授業の最終日 (補講を除く) に実施した．クラス 1 については，Java と Nigari の違いや，Java の学習における Nigari の効果などについて問う設問も用意した．

10. 実 験 結 果

クラス 1 のアンケートの集計結果を 10.1〜10.4 に示す．クラス 2，3 との比較結果を 10.5 に示す．

10.1 授業の難易度の変化

図 9 に，アンケート結果に基づく，授業の難易度の，授業ごとの変化を示す．

5 月 12 日の時点では，変数の基礎的な概念の習得が内容であり，ほとんどの学生が問題なく理解していると考えられる．それ以降，`if` 文，`while` 文の概念を習得するにつれ，難易度が上がっていく．これは，`if` 文，`while` 文などの個々の概念の難しさを示すというよりは，プログラムが複雑になり仕上げるのが難しくなったことを示している．

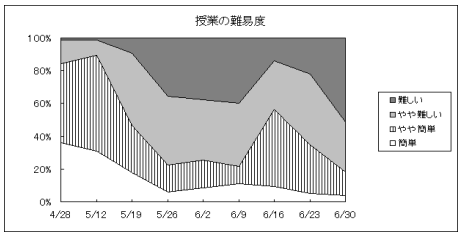


図 9 アンケート結果：授業の難易度の変化 (クラス 1)

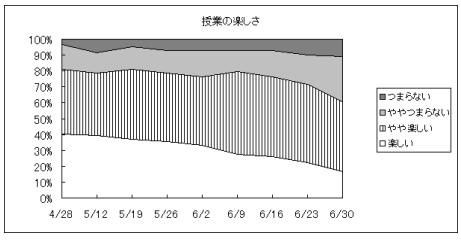


図 10 アンケート結果：授業の楽しさの変化 (クラス 1)

6 月 16 日は最初の Java での授業であるが，難易度が低下している．これは Nigari で習得した変数や while 文の概念をそのまま Java に応用することができたためだと考えられる．

10.2 授業の楽しさの変化

図 10 に，授業の楽しさの，授業ごとの変化を示す．

授業の難易度が上がるにつれ，若干ながら楽しさは減少するが，難易度の上昇に比べて緩やかな減少で，「難しいけれどおもしろい」と感じている学生が多いと考えられる．Java の授業に入ると，楽しさの減少が目立つようになる．

10.3 Nigari を用いた授業の利点と欠点

7 月 7 日実施の総合評価のアンケートに，授業に関する感想を自由に記入する欄を設置した．この欄に書かれた意見のうち，Nigari に関する意見が 27 件含まれた．その詳細を図 11 に示す．Nigari の利点として，「プログラムが視覚的に表示されてわかりやすい」「初心者にとってとっつきやすい」といった意見が多く見られた．また「オブジェクト指向，マルチスレッドなどの概念を理解する手助けとなった」という意見もあった．

一方，Nigari を学習する時間によって，本来の Java の学習ができる時間が減ったという意見が見られた．この意見はプログラミングの経験者から特に多く寄せ

授業に対する意見総数	73 件
Nigari に関する肯定的意見	19 件
Nigari に関する否定的意見	9 件
肯定的意見の詳細:	(件)
敷居が低くとっつきやすい	7
プログラムが視覚的	7
オブジェクト指向の理解の助けになる	2
その他	5
否定的意見の詳細:	(件)
早く Java に移ってほしかった	6
Nigari ではカバーできない点が多い	1
Java とのギャップを感じた	1
簡単すぎる	1

図 11 アンケート結果：自由意見 (クラス 1)

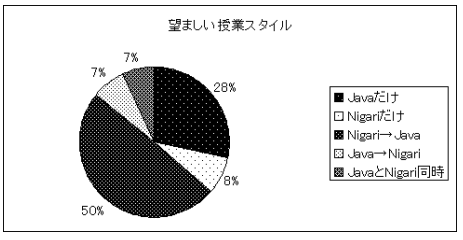


図 13 アンケート結果：望ましい授業スタイル

クラス	平均点
1	85.8
2	75.4
3	81.1

図 14 共通試験結果

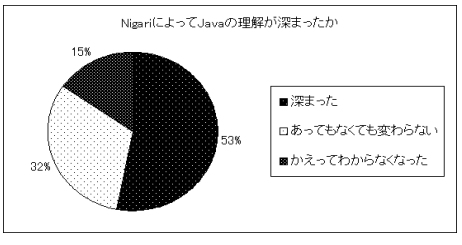


図 12 アンケート結果：Java によって Nigari の理解が深まったか

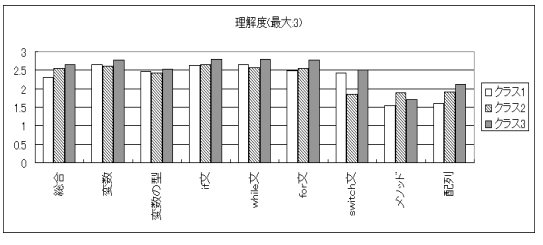


図 15 アンケート結果：クラス別，分野別理解度

られた．また，配列や文字入力など，Nigari では扱わなかった仕組みを Java で学習しなければならないので，Nigari を用いることに疑問を感じる学生もいた．

10.4 Java の学習に Nigari を用いることの是非

図 12 に,「Nigari を使ったことで Java の理解が深まったか」という質問に対する回答の割合を示す．これについては，半数強の学生が「深まった」と答えた．一方，15%の学生が「かえってわからなくなった」と答えている．

図 13 に,「望ましい授業スタイルはどれか」という質問に対する回答の割合を示す．約半数の学生が，今回採用した Nigari を導入に用いて，その後 Java に移行するというスタイルを支持したが，3 割の学生は，Nigari を使わずに Java を最初から学習するのがよいと答えた．

このように，大方の学生からは，Nigari を用いることに対して賛同を得られたが，Nigari は用いないほうがよいという意見も少なくなかった．その理由として，Java 特有のメソッドの宣言方法など，Nigari とは異なる仕様の部分に違和感を覚え，混乱してしまう，と述べたものがみられた．

10.5 クラス 2,3 との比較

10.5.1 共 通 試 験

共通試験の結果を図 14 に示す．

クラス 1 の成績が最も高い．ただし，共通試験を Web で実施した際に機器上のトラブルが起きたことを考慮して眺める必要がある．トラブルはクラス 2 で多発し，クラス 3 で若干問題が発生したものの，クラス 1 ではほとんど発生なかった．

10.5.2 最終アンケートの結果

授業で学習した項目についてどれほど理解ができたか (自己申告) を図 15 に示す．

クラス 1 では，メソッド，配列の部分が特に落ち込んでいるが，その要因として次のようなことが考えられる．

- Java でのメソッドについては，Nigari で扱った例が少なかったこと，Java での宣言形式が Nigari のそれと比べて複雑であることなどから，理解しにくくなった．
- 配列については，Nigari では学習せず，Java になって初めて学習した．さらに，学習時間が少なかった．

11. 考　　察

実験結果から，Nigari にはつぎのような効果があると言える．

- **簡単な言語仕様を持つ Nigari を用いて，プログラミング学習の敷居を低くすることができる．**
- **Nigari System によるプログラムの視覚化によって，学習者の興味を惹くことができる．**
- **Nigari を用いて，Java の基本的な仕組み，特に変数や制御構造，オブジェクト指向などの概念を習得でき，それがそのまま Java に応用できる．**

一方，現在の Nigari には次のような問題点があると言える．

- **Java を詳しく学習したい学生にとっては，Java を学習する時間が減ってしまう．**
  この問題は，学習者の能力がすでに高い場合に顕著になる．そこで，能力別にクラス分けを行い，例えば，プログラミング経験がない学生だけに Nigari を使わせるといった改善案が考えられる．
- **クラスやメソッドの宣言など，Java と Nigari で仕様が違っている部分について，混乱を来たす可能性がある．**
  学生からの意見を考慮すると，Nigari の言語仕様をもう一段 Java に近づける必要性があると考えられる．例えば，クラス宣言の構文やメイン文を廃して必ずメソッドを宣言しなければならなくするなどの言語仕様改訂の是非を，現在検討中である．
- **Java の入門として教えるべき内容のすべてを網羅できているわけではない．特に配列や入出力についての学習を補助する仕組みがない．**
  Nigari でのプログラムの可視化機能は，変数 x，y，p を用いた画像表示にとどまっている．この仕組みだけでは，配列のようなデータ構造を視覚的に表示することはできない．
  この問題の改善案として，例えば図 16 で示すように，メインウィンドウの中に変数の値や配列の内容を直接表示したり，オブジェクト間の参照関係を矢印で図示したりするなど，可視化機能を拡張する作業を現在進めている．

12. ま　と　め

Java 言語への導入として，Java に似ていて，かつ簡素な言語を用いて Java の基礎を学習するための言語 Nigari とその環境 Nigari System を提案した．

実験では，最初に Nigari を用いて，その後 Java へ移行するという授業を実施し，この方法によって学習者への負担を減らし，学習者の興味を持続させることが可能であり，しかも Nigari で学習したことが Java にも適用できることを示した．

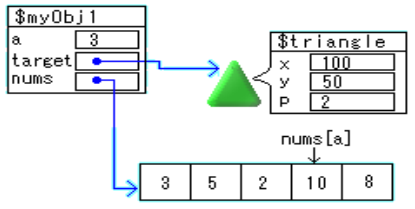


図 16　Nigari の可視化機能の拡張

しかしながら，学生の意見などを考慮すると，Nigari の言語としての簡素性が，その後の Java の学習に良い影響を与えているとばかりは言えない面がある．一方，Nigari の可視化の仕組みは多くの学生から歓迎されたと考えてよい．そこで，Java のプログラムそのものを可視化する仕組みを開発して Nigari を使った場合との学習効果を比較するなど，Nigari の特性をさらに詳しく分析していく予定である．

参　考　文　献

1) Westfall, R.: Technical opinion: Hello, world considered harmful, *Communications of the ACM*, Vol. 44, No. 10, pp. 129–130 (2001).
2) 長慎也, 川合晶, 日野孝昭, 前島真一: Nigari System (2003). taurus.kake.info.waseda.ac.jp/Nigari/.
3) 兼宗進, 御手洗理英, 中谷多哉子, 福井眞吾, 久野靖: 学校教育用オブジェクト指向言語「ドリトル」の設計と実装, 情報処理学会トランザクション 「プログラミング」, Vol. 42, No. SIG11, pp. 78–90 (2001).
4) 和田勉: LOGO を用いた「プログラミングの世界」への導入教育の経験, 情報処理学会研究報告「コンピュータと教育」, Vol. 92, No. 77, pp. 9–18 (1992).
5) 橋本裕, 早川栄一, 並木美太郎, 高橋延匡: プログラミング学習を支援する言語処理系「NB2」の設計, 情報処理学会研究報告「コンピュータと教育」, Vol. 47, No. 5, pp. 33–40 (1998).
6) 吉良智樹, 並木美太郎, 岩崎英哉: 初心者入門用言語「若葉」の言語仕様と処理系の実装, 情報処理学会トランザクション 「プログラミング」, Vol. 40, No. SIG10, pp. 28–38 (1999).
7) 結城浩: Java 言語プログラミングレッスン (上), ソフトバンク パブリッシング (1999).