オブジェクト指向プログラミングを支援するリアルタイムシミュレーションシステム

寺尾 剛* 白井 治彦** 黒岩 丈介*** 小高 知宏* 小倉 久和***

Real-time Simulation System that Educates Object Oriented Programming

Go TERAO*, Haruhiko SHIRAI**, Josuke KUROIWA***
Tomohiro ODAKA* and Hisakazu OGURA***

(Received January 31, 2008)

We constructed the system that supported the computer language education. It supports the education of object-oriented programming. This system targets object-oriented programming beginners. The support system is a game base. That name is RFG (Robot Fight Game). Learner makes the agent by using the Java. It studies by making the agent. Because the system is a game, it is possible to study easily. The system can be repeated and be used, It is possible to use it for a long time. The system was evaluated by the experiment and the questionnaire. The experimental subjects are six people. As a result, the effectiveness of the system was confirmed.

Key words : Simulation System, Education System, Agent System, Object Oriented Programming

1. はじめに

近年, ソフトウェアシステムの規模の拡大に伴いオブジェクト指向の概念が生まれた. この考え方は計算機言語教育でも重要な要素になってきた. ゲームをベースにした教育支援システムやアニメーションを用いた教育システム [1],[2],[3] はこれまでにも提案されているが, オブジェクト指向の理解を中心としたシステムは確立されているとはいえない. そこで, 本研究では初めてオブジェクト指向プログラミングを行う人がオブジェクト指向の概念を理解できるゲームベースの教育支援システムの作成を行った.

様々なシステムで, 多機能化やユーザーに使いやすいシステムの開発が進んできた. それによって, システムの規模はますます拡大し, プログラムを作るのに必要な知識も増え, プログラムも複雑化してきた. その結果, オブジェクト指向という概念が生まれた. オブジェクト指向とはどう処理を行うかではなく, どのような処理をさせたいかということに注目する考えかたであり, またすべてのものをオブジェクトととらえオブジェクトによって構成しているという考え方である.

教育機関の教育方法として, 教師が学習者に課題を提出し, 学習者がその課題を行うという方法がとられている. この方法を使用することによって, 多くの人に教育を行うことが可能となっている. しかし, 個々の実力に合わせた学習が難しい, 与えられる課題をこなすだけとなりやる気がでないなどの問題点も考えられる.

そこで学習者自身が自主的な学習を行えるような, また, 個々の実力に合わせた学習ができるような教育システムの構築が急務といえる.

そこで, 本研究ではオブジェクト指向を理解させる一つの支援システムの構築を目的とした. 構築の際には学習者が自主的に学習が行え, オブジェクト指向の概念を理解し、既存のクラスを利用したプログラミングを行

*工学研究科原子力・エネルギー安全工学専攻
**技術部
***工学研究科知能システム工学専攻
*Nuclear Power and Energy Safety Engineering Course, Graduate School of Engineering
**Dept. of Technology
***Human and Artificial Intelligent Systems Course, Graduate School of Engineering

えるようなシステムの構築を行うことも目的とした.

ここではシステムにリアルタイムシミュレーションシステム（ゲーム）を取り入れることによって実現している. リアルタイムシミュレーションシステムを用いることで次のようなことが考えられる. 結果がすぐに得られ, 得られた結果より, 自分が行ったことがどのように反映されたかが, すぐにわかる. また, このシステムを使用した学習者が何度も使用でき, なおかつゲームを楽しませながら教育支援を行うことによって, 従来行われていた教育方法よりもより学習者が自主的に学習が行える環境を与えることができるようになるのではないかと考えた.

本研究ではオブジェクト指向の概念とjavaで使われている一部の機能を使用するリアルタイムシミュレーションシステムを用いたロボット対戦ゲームRFG（Robot Fight Game）を作成した. RFGによってオブジェクト指向の概念を学習者に理解させることでOOP（オブジェクト指向プログラミング）が行えるようにすることが本研究の目的である. さらに, このシステムを用い, オブジェクト指向の学習における1つの学習方法を提案することも目的のひとつである.

また, 構築したRFGによって学習者の理解にどのように反映されるか評価実験を行った.

2. シミュレーションシステムの構築

2.1 シミュレーションシステムの設計

現在さまざまな教育機関によって, 計算機言語教育が行われている. この学習においてはプログラミングの基礎である構造化という考えを中心に学習が行われている. その結果, 構造化の考え方を学習するために教材やシステムは多くの種類があり, 学習者にあった教材を選ぶことが可能である. しかし, オブジェクト指向などの概念を理解できるような教材やシステムは構造化に関する教材などにくらべ少ないと考えられる. そこでオブジェクト指向の概念を理解させる学習支援システムの構築を行った.

システムの構築にあたり現在の学習における問題点や重要な要素を考え, 設計の基本方針とした.

現在, 計算機言語教育にあたり, 教師が生徒に基礎を教え, 教師に教わった基礎を用いた課題をこなし提出を行うという過程を置いて教育が行われている. この教育方法は多くの人を教育するにおいて有効であるといえる. しかし, 教師が生徒に課題を与えるという形態をとっているため, 生徒はやらされている感が強くやる気が向上しないという問題点も考えられる. また, 課題に面白みがなく学習者のやる気を促すのが難しいという問題も挙げられる.

このようにいくつかの問題点が存在する. そこで問題点を解決する方法として, 構造化の考えをメインにした課題をこなすのではなくオブジェクト指向の特徴が顕著に現れたプログラムについて学習することによって, オブジェクト指向の教育を進めるようにする.

また, 生徒のやる気を向上させる方法として, 次のことを考えてシステムの作成を行う. 単純に課題を解かせるしていくのではなく, ゲーム性や娯楽性のある教育システムを作成する. これによって学習者が楽しみつつまた, 持続性の高い学習環境を作成することが可能であると考えた.

その他として, 学習者が行った作業によりどのような結果が反映されるかが一目でわかる必要がある. 反映された結果がすぐに確認できるということも重要な要素であると考えられる.

また, プログラムを初めて勉強する人, またはオブジェクト指向プログラミングを行ったことのない人のための教育支援を基本としてシステムを構築する.

オブジェクト指向の概念を理解を促すにはオブジェクト指向の概念に対応するプラットフォームを作成することによって解決できると考えシステムの構築を行った.

2.2 RFGの構築

本研究により作成されたシステムは, プログラムを初めて勉強する人, またはオブジェクト指向プログラミングを行ったことのない人に向けて作られた計算機言語教育を支援するシミュレーションシステムである. オブジェクト指向という概念には多くの概念がある. そこでこのシステムではオブジェクト指向の概念の一部であるクラス, インスタンス, メソッド, 継承の概念の学習支援が行えるシステムの構築を行った. 構築を行ったシステムはRFG（Robot Fight Game）である.

RFGはオブジェクト指向の概念に対応しているプラットフォームを形成している. このシステムでは学習者が一体のエージェント（ロボット）を作成することとなる. エージェントの作成においてオブジェクト指向の概念を用いることとなり, このシステムを使用することにオブジェクト指向の概念が理解できるようになっている. RFGでは学習者がエージェントを作成することとなるが, エージェントの作成方法として学習者がプログラミングを行い作成することとなる.

RFGではオブジェクト指向の概念に対応するプラットフォームを形成しているが, 学習者がエージェントやエージェントのパーツを個々のオブジェクトと認識することによってオブジェクトはオブジェクトによって構成していることを理解する. システムではエージェント

のパーツをクラス, パーツの装着, 実装をインスタンス, パーツにさせる行動や機能をメソッドとしてオブジェクト指向に対応させることによりオブジェクト指向のクラス, インスタンス, メソッドを理解させる. また, 継承の概念はシステム内に用意してあるパーツ（クラス）から新たな機能（メソッド）を付け加えることで特徴を受け継いだパーツ（クラス）を作成することに対応している.

RFG は java によって構成されている. システムは図 1 のように構成されている. インターフェイスでは学習者がエージェントの際に必要なエディタを開き, エージェントのプログラミング終了の際にはテキストファイルの拡張子を.java に自動に変更する. ここで拡張子を変更するのは学習者が作成したエージェントのクラスがそのままの形でシステムに反映されるためである. さらに変更された java ファイルをインターフェイスを通すことで java のコンパイラでコンパイルにかける. コンパイルにかけられることによって学習者が作成したエージェントのクラスを使用できるようになる. コンパイル後インターフェイスを通してプログラムが実行されることとなる. これによってエージェントを管理するシミュレータ部でシミュレーションが行われる. 学習者はシミュレーション上での結果をインターフェイスを返して確認する.

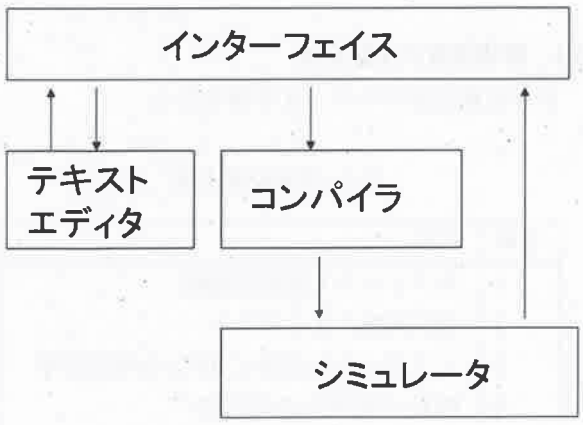

図 1: システムの構成

2.3 RFG を用いた学習

現在大学等で行われている計算機言語教育では最初に講師が教科書や教材を使い, 概念やプログラミング方法の説明を行う. その後に課題を与えて理解を促すという形態である. RFG を用いた学習においてはここで課題を与える代わりに RFG を使用することで, 理解を促すことになる. RFG は学習者がロボット (エージェント) を組み立て, そのロボットの戦術を作成し, 作成したロボットとその動作（戦術）を行うプログラムを用いてロボット同士を戦わせるシステムであり, エージェントの作成や行動ルールの作成には付録 A にある表 5 のエージェント記述言語を用いる. この記述言語は java の一部の機能だけを使用した記述言語であり, 学習者は実際の環境に近い形でオブジェクト指向プログラミングの学習を行うこととなる. エージェント同士の戦闘をさせるまでにはエージェントとその行動ルールを何度も作成することになり, 理解させることができる.

RFG を用いた学習の流れを表 1 に示す. 手順 1 では大学等で行われる学習同様にオブジェクト指向の概念やプログラミング方法の説明を行う. 手順 2 ではオブジェクト指向の概念が RFG にどのように対応しているかを説明する. 手順 3 では RFG の使用方法を説明する. 手順 4 では RFG による学習を行う.

表 1: RFG を用いた場合の学習の流れ

手順	内容
1	オブジェクト指向の説明
2	オブジェクト指向と RFG の対応説明
3	RFG の使用方法の説明
4	RFG による学習

ここでは手順 4 で行われた RFG の学習の流れを詳しく説明する.

手順 4 の RFG 使用による学習の流れを表 2 に示す.

表 2: 学習の流れ

手順	内容
4-1	エージェントの設計
4-2	エージェントの作成
4-3	エージェントの行動ルールの作成
4-4	コンパイル
4-5	エージェントの動作確認

手順 4-1 では学習者はエージェントの設計を行い, エージェントを何のパーツによって構成するかを考える.

手順 4-2 ではエージェント記述言語を用いて, プログラミングによってエージェントの作成を行う. エージェントの作成の際には図 2 の A の部分にエージェントのパーツを記述言語を用いて表す. 例としてプログラムの Missile weaponUnit1 = new Missile(); の文に注目する. これは表 5 の番号 3 のルールを使っている. <クラス名>は番号 4 のルールから< Weapon クラス>を選

```
public class RoboL {
    Missile weaponUnit1 = new Missile();
    Sword weaponUnit2 = new Sword();
    WalkUnit moverUnit = new WalkUnit();
    ShortSensor sensorUnit = new ShortSensor();
    MiddleSensor sensorUnit2 = new MiddleSensor();

    public void go (){
      weaponUnit1.ready();
      weaponUnit2.ready();
      sensorUnit.ready();
      sensorUnit2.ready();
      moverUnit.ready();
      moverUnit.setspeed(100);

      moverUnit.setdirection(sensorUnit.getdirection());
```

A

B

図 2: 学習者が作成するプログラム例

んでいる. さらに< Weapon クラス>では番号 5 のルールからは Missile を選ぶ. 続いて<オブジェクト名>のルールでは学習者が名を自由につければよいのでここでは weaponUnit1 とつける.

このように記述ルールに従うと Missile weaponUnit1 = new Missile(); のような文が構成される.

この文によって weaponUnit1 という名の Missile をエージェントが持つこととなる. 同様な操作を何度か繰り返しエージェントを作成する.

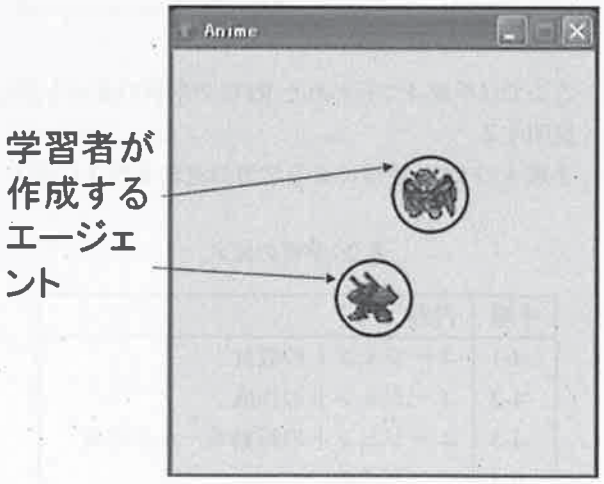

図 3: シミュレーション画面

手順 4-3 のエージェントの行動ルールの作成では作成されたエージェントの行動ルールを作成する. 行動ルールの記述場所は図 2 の B の部分である. 行動ルールは番号 10 の記述ルールによって記述される. 行動ルールの作成もエージェントの作成と同様に記述ルールに従って記述することとなる. このように何度も記述ルールに従ってプログラミングを行うことによって学習者は学習していくこととなる.

また, 手順 4-4 では作成されたエージェントとその行動ルールをコンパイルすることになる. ここで実際にエージェントと行動ルールの作成方法に間違いがなかったかどうかを確認することとなる.

最後に手順 4-5 ではコンパイルされたエージェントの動作を確認し, 学習者が考えたエージェントが作成され, 学習者が予期した行動をとるかどうかの確認を行う. 図 3 は RFG を実際に運用した画像である. このような画像からエージェントが予期した行動かどうかを学習者が判断することとなる.

3. RFG の評価実験

本システムはオブジェクト指向の学習を行ったことのない人を対象としている教育支援システムである. それらの人に対してシステムが有効であるかを検証するために評価実験を行った.

この実験の被験者はオブジェクト指向の学習をしたことのない大学 3~4 年生を対象とし行った. 被験者の人数は 6 名である. 被験者は C 言語などのオブジェクト指向ではない計算機言語教育を受けたことのある学生で, 初等のプログラミング技術を有している. この実験では 6 名の被験者を 2 つのチームに分けて実験を行った. 各チーム内でエージェントを一体作成するものとする. また, 実験の期間は 1 週間である.

3.1 評価実験の実験手順

評価実験は表 3 のような手順で行う.

表 3: 評価実験手順

手順	内容
1	オブジェクト指向の説明
2	選択問題によるテスト
3	オブジェクト指向と RFG の対応説明
4	RFG の使用方法の説明
5	RFG による学習
6	テストとアンケート

手順 1 では被験者はオブジェクト指向の学習を受けたことがないので, オブジェクト指向の概念がどのようなものかを知ってもらうために行った. 説明の際には通常の教育機関で行う講義形式で説明を行った. 手順 2 では 2 択の選択問題を被験者に解いてもらい先ほどの説明で理解度を調査した. 選択問題は文章で書かれている問題を「はい」か「いいえ」で答えるもので表 4 に示すように全 14 項目である. 選択問題は被験者すべての

学生に対して行った．手順3ではオブジェクト指向の概念がRFGではどのように対応しているかの説明を行った．手順4ではRFGの使用方法を説明した．手順5ではRFGを使用した．試用期間の最後には各チームで作成したエージェント同士を対戦させた．手順6では2で行った2択の選択問題を再び解き，理解度（正答率）が向上しRFGがオブジェクト指向の学習に有効であるか検証した．また，被験者にはアンケートを行い，RFGのゲームとして楽しめたか，システムは使いやすかったかなどいくつかのシステムに関する感想を調査した．

表4: 理解度評価問題

問題の内容
オブジェクト指向はデータやその集合を現実世界のモノになぞらえた考え方である
オブジェクト指向とは計算機言語特有の考え方である
オブジェクト指向とはデータを中心としてそのデータに対する操作として手続きを書いていくのではなく，手続きを中心に考えていく考え方である
オブジェクトはいくつかのオブジェクトで構成されていてもよい
オブジェクトは1つのオブジェクトからしか構成されない
インスタンスによってクラスからオブジェクトが生成される
インスタンスによってオブジェクトからクラスが生成される
1つのクラスからは1つのオブジェクトしか生成できない
クラスとは設計図みたいなものである
クラスとオブジェクトは同意義である
メソッドとは物体全体に対する命令である
メソッドとはあるオブジェクトに対する命令である
継承とはそのオブジェクト特有の機能だけを引き継ぐものである
あるオブジェクトを継承した場合継承したオブジェクトは継承したオブジェクトに類似する

3.2 評価結果

システム運用前に行った選択問題に対する正答率を図4に示す．また，システム運用後に行った選択問題に対する正答率を図5に示す．質問項目は14問あり，システム運用前では68％の正答率に対してシステム運用後の回答では82％と正答率の向上につながった．また，システム運用前と運用後の正答率では継承やメソッドに対する問題の正答率が大きく向上していた．これはRFGのシステム内でエージェントに動作させる際に他の多くの機会でメソッドを使用することになるためであると考えられる．また，継承に関してはもとのパーツから新たなパーツを作る際にもとのパーツにあるメソッドが継承してもそのまま使えるということをプログラミングをすることによって理解したためと考えられる．

システムの運用前と運用後の選択問題のテストでは運用後の方がほとんどの項目で正答率が向上した．しかし，「1つのクラスからは1つのオブジェクトしか生成できない」の問いに対しては向上が見られなかった．これはRFGのシステムの特性上エージェントのプログラミングを行う際に1つクラスから2つ以上のオブジェクトを作成することがないためと考えられる．

システムの特性上すべての概念を理解するのにはいたらなかったが，システム運用前とシステム運用後では問いに対する正答率の向上が見られた．これらのことからRFGを用いることでオブジェクト指向の概念を学習支援するのにある一定の有効性があることがわかった．

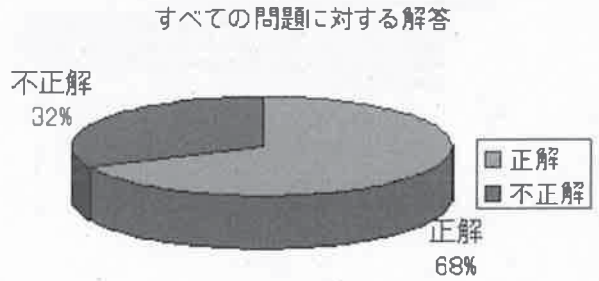

図4: システム運用前に行った選択問題に対する正答率

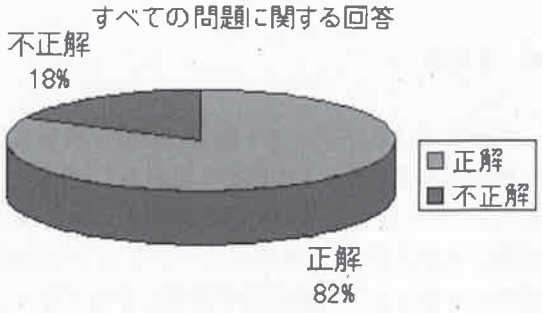

図5: システム運用後に行った選択問題に対する正答率

図6は学習者が実際に作成プログラムである. プログラムでは記述ルールに従いエージェントとその行動ルールが作成されていることが確認できる. また, ver3 boos = new ver3(); の文はエージェントの作成で ver3 をエージェントのパーツとしていることがわかる. そして, 番号3の記述ルールを使用し, クラス名は ver3 であるがシステムでは Weapon クラスの Laser, Bomb, Missile, Shield, Sword もしくは Sensor クラスの ShortSensor, MiddleSensor, LongSensor か Mover クラスの Booster, WalkUnit の各クラスしか用意していない. このことから学習者はシステムであらかじめ用意してあったクラスを継承し, 学習者独自のクラスを作成していることが確認できる. このように学習者はシステムの機能を利用しオブジェクト指向プログラミングが行えていたことが分かった. このことからもシステムを運用することでオブジェクト指向の学習に対して有効であることが考えられる.

```
public class RoboR {
    ver3 boos= new ver3();
    MiddleSensor middlesensor = new MiddleSensor();
    bom bomb2 =new bom();

        public void go(Anime field){

            middlesensor.setfield(field);
            middlesensor.ready();
            boos.firstset(field);
            bomb2.firstset(field);
            boos.setspeed(100);
            bomb2.setdistance(middlesensor.getdistance());

            if(middlesensor.getdistance()>100){
                  boos.setdirection(middlesensor.getdirection());
                  bomb2.setdirection(middlesensor.getdirection());
                  bomb2.setdistance(middlesensor.getdistance());
                  bomb2.fire();

            }
```

図6: 学習者が作成したプログラム

また, アンケートでの感想として, java のコンパイラーをそのまま利用しているため, どの部分でプログラムが間違っているかが解りにくい. 継承ではなく完全なオリジナルのクラスを作成できるようにし欲しいなどの多くの要求があり, これらが今後の課題となった.

4. まとめ

本研究ではオブジェクト指向の概念を理解できるような教育支援システムの構築を目的とし, 目的の達成にあたり, ゲームベースの教育支援システムを導入した. 支援システムでは, 学習者にエージェントの作成, またそのエージェントの動作を学習者にプログラミングさせるというアイディアを導入した. 学習者がプログラミングを行う際には既存のクラスのメソッド呼び出しなど, オブジェクト指向の機能を用いることにより, 実践的で効果的な支援ができたと考えられる. ロボットはいくつかのパーツを組み合わせることで作成できるようにした. これにより学習者のオリジナルティをだすことができ, より楽しめるようなシステムが作成できた.

オブジェクト指向の一部である既存のクラスを利用してプログラミングを行うという点について注目した. エージェントのパーツをクラス, エージェントの作成（パーツの実装）をインスタンス, パーツに動作をさせる命令をメソッドとおき, 学習者がもともとあるパーツから新たなパーツを作成するを継承におきかえることによりゲーム内でのモデルとオブジェクト指向の考えとうまく対応することができた. それによりゲームを行いながら, オブジェクト指向のクラス, インスタンス, メソッド, 継承などの概念を学習者に理解させることができた. また, システムを使うことで従来の学習方法とは違う, オブジェクト指向の学習方法を提案することができた.

オブジェクトを利用したプログラミングの一部を本システムを使うことで理解できるが, まだ多くの課題が残されている. 例として挙げられるのが自作クラスの作成といった機能についてである. 特にオブジェクト指向を行う上で自作クラスの作成は必須であり, 多くの場面で使われている. 今後は学習者が作成したクラスが RFG 内で使えるようにするなどの拡張が必要である.

本研究ではオブジェクト指向という幅の広い概念を支援するシステムを構築したため多数の課題を残したが, 今回の研究においてオブジェクト指向の一部を理解できるようなシステムを構築することができた. また, 本システムを用いたオブジェクト指向の学習方法が提案できた.

参考文献

[1] 松田 武, 斎藤 健司, 斎藤 一, 前田 隆:学習者の興味を引き出す学習支援環境について, 報処理学会全国大会講演論文集 Vol.67th, No.4, pp429-430. (2005)

[2] 長 慎也, 日野 孝昭, 前島 真一, 小田嶋 祐介, 佐々木 康太朗, かけひ 捷彦：プログラミングの入門に適した, 支援システムとコースデザイン, 情報処理学会研究報告 Vol.2004, No.100(CE-76) (2004)

[3] 税所 幹幸:プログラミング教育に利用する再起プログラム学習支援, アドミニストレーション Vol.11, No.1-2, 45-55 (2004)

付録 A

表 5: RFG の記述ルール

番号	要素		記述方法
1	＜プログラム＞	→	public class ＜エージェント名＞ { ＜インスタンス＞* public Void go() {＜ Action ＞*}}
2	＜エージェント名＞	→	RoboL ‖ RoboR
3	＜インスタンス＞	→	＜クラス名＞＜オブジェクト名＞= new ＜クラス名＞();
4	＜クラス名＞	→	＜ Weapon クラス＞ ‖ ＜ Sensor クラス＞ ‖＜ Mover クラス＞
5	＜ Weapon クラス＞	→	Laser ‖ Bomb ‖ Missile ‖ Shield ‖ Sword ‖＜継承クラス＞
6	＜ Sensor クラス＞	→	ShortSensor ‖ MiddleSensor ‖ LongSensor ‖＜継承クラス＞
7	＜ Mover クラス＞	→	Booster ‖ WalkUnit ‖ ＜継承クラス＞
8	＜継承クラス＞	→	学習者が作成したクラス
9	＜オブジェクト名＞	→	学習者が自由につける
10	＜ Action ＞	→	＜メソッド呼び出し＞;
11	＜ Action ＞	→	if(＜条件＞) ＜メソッド呼び出し＞;
12	＜メソッド呼び出し＞	→	＜オブジェクト名＞.＜メソッド名＞(＜式＞,…)
13	＜メソッド呼び出し＞	→	＜メソッド名＞(＜式＞,…)
14	＜メソッド名＞	→	各オブジェクトにより決定
15	＜式＞	→	＜式＞＜四則演算子＞＜式＞
16	＜式＞	→	＜項＞＜四則演算子＞＜項＞
17	＜式＞	→	＜項＞
18	＜項＞	→	整数
19	＜項＞	→	＜オブジェクト名＞.＜変数名＞
20	＜式＞	→	＜変数名＞
21	＜変数名＞	→	各オブジェクトにより決定
22	＜四則演算子＞	→	＋ ‖ － ‖ ＊ ‖ /
23	＜条件＞	→	(＜項＞＜比較演算子＞＜項＞)
24	＜比較演算子＞	→	＞＝ ‖ ＞ ‖ ＝ ‖ ＜ ‖ ＜＝