プログラミング初学者のための学習支援ツールの開発

早川美徳, 酒井正夫, 三石大, 長谷川真吾, 静谷啓樹, 磯邉秀司, 小泉英介, 二階堂秀夫

東北大学 教育情報基盤センター

hayakawa@cite.tohoku.ac.jp

概要：東北大学では，大学初年次の情報教育において，アカデミックスキル習得の一環としてプログラミング演習を含んだ授業を実施している．そこで用いるために，初学者でも容易にプログラム作成とその実行が可能なテキストエディタ(簡易 IDE)，および，エディタと連携して動作し，教員が学生からのレスポンスをリアルタイムに収集するためのソフトウェア，簡易的なグラフィック描画用ソフトウェアを開発し，授業で使用した．

1 はじめに

東北大学では，大学初年時の情報教育科目として，全学教育「情報基礎」が開講されており，文系理系を問わず多くの学部・学科で必修科目に，その他でも選択科目に指定されている．この科目は,平成 20 年に策定された「全学教育『情報基礎』第３版」に沿って，共通化されたシラバスのもとで実施されており，担当教員によって扱うトピック等に多少の違いはあるものの，

(1) *情報技術を活用した基本的な知的生産活動が可能になること.*

(2) *コンピュータサイエンスの手法による論理的思考と問題解決ができるようになること.*

(3) *情報社会の一員として責任を自覚し，情報の科学・技術と人間との関係に問題を発見できるようになること.*

が共通の目標として設定されている[1].

その中でも，(2)は今日的な意味での高度な知的生産者を養成するために，特に重要性が増していると考えられるが，この科目においては，実践的なプログラミングのスキルを学ぶことに専ら重点を置いているわけではなく，プログラムミングは問題解決ための汎用的・現実的アプローチではあるものの，必ずしもそれが唯一ではない，という立場が取られている．

その意味で，ソフトウェアの使い方の修得や煩雑な操作そのものに割かなければならない時間をできるだけ減らし，手軽にコーディングと実行結果の確認が可能な環境を提供することが，特に初学者向けのこうした科目では必要とされていると考えた．

そこで，授業での使用を想定し，使用するプログラミング言語を選ばずに使うことの出来る簡易 IDE（テキストエディタ），演習室で学生のレスポンスを収集するツール，および，簡便にコンピュータグラフィックスを試すことのできるアプリケーションを開発したので，本稿ではその詳細について報告する．

2 簡易ＩＤＥ兼テキストエディタ

前節で述べた「情報基礎」は，殆どのクラスで Linux 環境(CentOS)を使って授業が実施されている．プログラミングに関係する課題を実施する際に，多く用いられている開発環境は，OS に標準的にインストールされているテキストエディタ（KWrite），端末エミュレータ（Konsole）とその中で動作する Linux コマンド，および C コンパイラ(gcc)であった．

コマンドを使ってのコンピュータ操作の経験を持たない受講者にとっては，コマンドラインインターフェースそのものに慣れるだけで少なからぬ努力と忍耐を強いられるであろうし，エディタとターミナルを行き来しながらの作業は，様々なミスを誘発する要因となり得る．

例えば，コンパイルコマンドを含む基本的なコマンドとそのオプションなどがなかなか覚え

られない，シェルの作業ディレクトリを正しく把握しておらず，エディタのファイルの保存先とは異なるディレクトリで操作する，エディタで空白を含む名前を付けてファイルを保存したため，シェル環境でうまく取り扱うことができなくなる，エディタでの編集結果を保存し忘れたまま，コンパイル・実行へと進む，といった状況をしばしば目にする．

また，エディタでコーディングを行う際，コメント部以外の場所で日本語文字を入力したため，コンパイルエラーとなってしまう，特に，コード中に紛れた全角スペースが見つけられない，行番号表示の機能に気づかず，エラー等の際に手で行数を数えている，といった様子もしばしば観察されていた．

コマンドラインベースではなく，本格的な統合開発環境（IDE）を活用することによって，上記のトラブルや課題の多くは解消可能かもしれないが，ソフトウェアの操作方法に加え，いくつかの専門的な用語や概念を修得しなければならず，全学教育の一環として行われている科目の，しかも限られた授業時間数の中では，導入しづらいのも事実である．

そこで，ごく単純な機能のテキストエディタを基本として，その中で，ファイルの保存，コンパイル，実行，さらには，結果の出力を可能とする，言わば，「教育用簡易版 IDE」を新規に開発することにした．そして，亀のようにゆったりと歩むことをイメージして，TurtleEdit と命名した．

開発環境には NetBeans を用い，全て Java でコーディングしているため（5,000 行程度），OS に依らず，各種の環境で実行可能である．GUI 部分は Swing コンポーネントを使用した．ソースコードは MIT ライセンスで公開している（NetBeans のプロジェクトファイル形式）[1]．

TurtleEdit は IDE としてはごく基本的な機能しか提供しないが，初学者の学習を支援するために，以下の特徴を持つ：

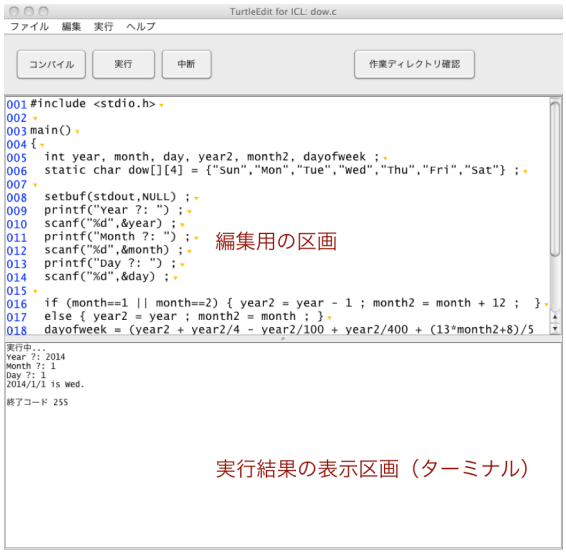

図 1: TurtleEdit の画面構成．

- 全ての全角文字は薄い色の四角で囲まれ，半角文字と明瞭に区別できる．
- 常に行番号が表示される．
- 空白を含む名前でファイルを保存しようとすると警告を出す．
- ボタンを押すと，あらかじめ登録したコマンドが実行される．その機能を用いて，コンパイル，実行，中断を，ボタン操作のみで行うことができる．
- コンパイルコマンドの終了コードが 0 以外（コンパイルエラー）であった場合は「実行」ボタンが働かない．
- 編集内容を保存しないまま「コンパイル」ボタンを押すと，ファイルの保存が促される．
- ボタンひとつで，作業ディレクトリの内容が確認できる．

TurtleEdit の画面構成を図 1 に示す．同時に開くことのできるファイル（ウィンドウ）はひとつのみで，ウィンドウ上部に「コンパイル」，「実行」，「中断」，そして「作業ディレクトリ確認」のボタンが配置されている．その下に，編集用作業区画，さらにその下に，実行結果を表

[1] http://seaotter.cite.tohoku.ac.jp/coda/tedit/

示する区画が配置され，両者の境界位置は自由に移動可能である．最下部の区画は，（ダム）ターミナルとして機能する．

ボタンを押したときに実行されるコマンドは，「設定」画面で，ユーザーが自由に変更することができる．その記述を書き換えることで，C, Java, Ruby, Python など，各種の言語に対応することができる他，TeX のコンパイルと結果のプレビューも可能である．典型的な設定はプリセットされており，使用言語の切り替えも簡単である．

ソースプログラムを記述した上で,「コンパイル」ボタンを押すと，以下の一連の動作が開始される：

1. ファイルが保存されていなければ保存ダイアログが表示される．
2. 設定されたコンパイルコマンドが別プロセスとして実行される．
3. コンパイラからの出力は，最下段の区画に表示される．
4. コンパイラの終了コードが 0 の場合は，内部フラグがセットされ，「実行」ボタンが有効となる．
5. コンパイルの途中で「中断」ボタンが押されたら，プロセスが中断される．

次いで「実行」ボタンを押すと，実行コマンドとして登録してあったコマンドが別プロセスとして起動される．プロセスの標準入出力は，画面最下段の区画に接続されるので，キー入力を伴うようなプログラムでも動作するが，端末制御を伴うようなプログラムについては，期待したように動作しないという問題がある．

3 レスポンス収集ツール

大人数がプログラミング演習を一斉に行うような状況では，学生毎に進度が大きく異なり，

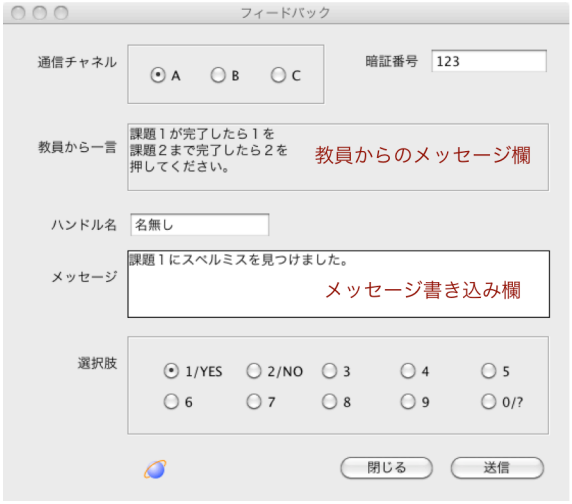

図２：レスポンス送信画面.

教員側が状況を正しく把握することが難しい場合がある．そのような際，学生のレスポンスを即時的に収集できる仕組みがあれば，「課題を終えたら，ボタン１を押してください」といった指示を与えることによって，的確に状況を知り，それをもとに適切な授業の進行の調整が可能となるものと考えられる．こうした機能を実現する仕組みとしては，「クリッカー」や，それと類似の，スマートフォン等でも利用可能なウェブベースのアプリケーションがいくつかのベンダーから提供されている．

しかしながら，PC 端末の並んだ教室で，レスポンス収集にそれらの PC を使わない手は無いし，演習中に携帯デバイスの操作を求めることは，学習者の注意を散漫にする恐れもある．

そこで，前節で述べた TurtleEdit の機能の一部として，学生が教員にレスポンスを送信する機能を実装し（図 2），合わせて，教員側がそれを収集するソフトウェア（TurtleMessenger, 以下 TM，図 3）を開発した．TM は，GLP3.0 のもとでソースコードを公開している[2].

このソフトウェアの基本的な動作は，以下のとおりである．

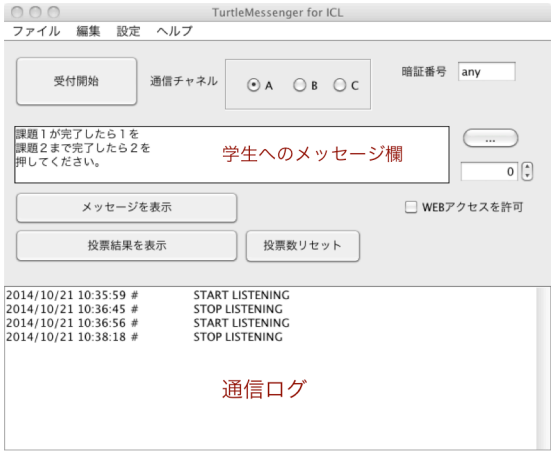

図 3: TurtleMessenger(TM)の画面構成.

1. 教員が TM（図３）を起動し，教員から学生へのメッセージ(指示等. 140 文字以内)を記入して,「受付開始」ボタンを押すと，定期的にサブネット内に UDP でブロードキャストが送信される．パケットには，教員からのメッセージと，公開暗号鍵が含まれる．
2. 学生が TurtleEdit のヘルプメニューから「フィードバック」を選択すると，回答用のウィンドウ（図２）が現れる．
3. TurtleEdit 側はブロードキャストパケットを受信し，教員側 PC の IP アドレスを取得するとともに，ウィンドウ内に教員からのメッセージを表示する．
4. 学生は，0 から 9 までの 10 個のラジオボタンの一つを選択し，「送信」することができる．また，ハンドル名と 140 文字を超えないショートメッセージを合わせて送信できる．送信パケットには，暗号化された学生のユーザーID と端末のホスト名が含まれる．
5. TM は TCP ポートをリスンしており，学生からの回答を収集し，それを表示する．

以上のように，このツールを用いることで，10 の選択肢からの「投票」と，Twitter のようなメッセージ投稿が可能となる．

このソフトウェアは，演習室内の端末が全て同一のサブネットに収容されていることを前提に設計されているが，同じサブネット内に異なるグループ（クラス）が存在する場合にも対応できるよう，使用する通信ポート（管理画面の「通信チャネル」）の切り換え，および，投稿用の暗証番号の設定が可能である．TM 側で暗証番号を設定しておくと，異なる暗証番号を持つメッセージが送信されても，それらは無視される．

レスポンスの結果は，送信者のログイン ID によって区別されるので，同じ ID でログインした者が，例えば，最初に選択肢「1」 を，次いで「2」を送信したとすると，集計結果としては，「1」の投票数が 1 つ減り，「2」が増える，という動作をする．この性質を使って「課題 1 まで完了したら 1 を，課題 2 まで完了したら 2 を送信すること」等の指示をすることによって，教室全体での進捗状況を教員がモニターすることが可能となる．

また、演習等の進行状況の把握の他にも，ピアインストラクションを取り入れた授業の実施，簡易アンケートやミニットペーパー，出席状況の把握等への利用も考えられる．

教員用ソフトウェア TM は，レスポンス結果を受講生と共有するために，所謂画像転送システムと併用することを想定して，簡便な操作で，投票の集計結果を別ウィンドウ上にグラフ表示したり，投稿されたメッセージを投稿時刻とハンドルネームと共に表示することができる．

また，TM はウェブサーバとしても動作させることが可能で（図３の管理画面で「Web アクセスを許可」をチェック），受講者側のウェブブラウザからアクセスすると，ブラウザ上に投票の集計結果，および，メッセージのタイムラインが表示される．そして，これらの情報は，JavaScript によって動的に更新される．

収集されたデータは，受信時刻や発信者のユーザーID，端末のホスト名とともに，時系列的に記録され，CSV 形式でファイルとして保存す

[2] http://seaotter.cite.tohoku.ac.jp/coda/tfbk/

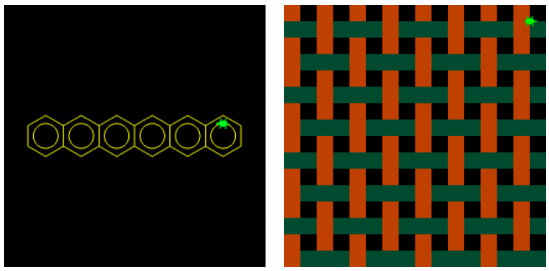

図 4: TurtleField(TF)による描画の例.

ることができる．担当教員は，それらを授業の振り返りや分析の参考とすることも可能である．

4 グラフィックス描画ツール

問題解決の手法としてプログラミングが有効なのは，単に「答え」が得られるだけでなく，結果を多様な形態で表現・表示できる点にもある．特にグラフィックス表示はその重要な手段であることは言うまでも無いが，汎用的なプログラミング言語を用いてLinux 端末等でCGを体験することは，初学者にとっては，かなり煩雑なコーディングを強いることになる．

そこで，TurtleEdit 等と併用して，初心者でも容易にタートルグラフィックスを可能にするためのソフトウェア TurtleField(以下，TF)を併せて開発した．TF は C で記述され（コード部分のみで 6000 行程度），Linux，Mac OS，Windows 用の実行ファイルを配布している[3]．

TF は TCP サーバーとして独立して動作し，クライアントからコネクションが張られると，画面上にカメのオブジェクトがひとつ生成される．オブジェクトはコネクションが張られている間だけ「亀場(turtle field)」と名付けられた二次元的な世界に「生存」し，クライアントプログラムからのコマンドによって，移動や描画を行う．複数のクライアントが同時に接続することも可能なので，ひとつの画面上で複数のオブジェクトが協調して描画作業をしたり，オブジェクトをロボットに見立てて「群れ」行動や「対戦」させたりすることも可能である．

[3] http://seaotter.cite.tohoku.ac.jp/coda/tfield/

クライアント・サーバー間は極めて単純なリクエスト・レスポンス型のプロトコルによって通信を行う．クライアントが描画等の命令を送信すると，サーバーはリクエストを処理し，結果とオブジェクトのステータスを返す．ステータスには，カメのオブジェクトに生じたイベント情報も含まれ，例えば，他のオブジェクトと衝突した際等に，それをイベントとして検出することができる．

サーバーは画面を常に繰り返し更新し，１回の画面更新あたり，クライアント側のリクエストを１つ実行してはクライアントにレスポンスを返す．従って，複数のクライアントが同時に接続しているような場合でも，画面の更新に応じたタイミングで，複数のオブジェクトの動作が（強制的に）同期される．

プログラミング初学者が TCP クライアントを作成するのは容易ではないので，あらかじめ各種言語（C，C++，Java，Python，Ruby，Scheme (Racket)，JavaScript (Node.js 上で動作)）用 API を提供することで，学習者は簡単な関数呼び出しだけで，「亀場」を扱うことができるようにした．描画コマンド（関数）は，教育用として実績のある，タートルグラフィックスの流儀に従って設計した．

例えば，C 言語を用いる場合，API 用ファイル tutle.h をダウンロードした上で，プログラムの冒頭に

```
#include "turtle.h"
```

と記述し，いくつかの関数呼び出しを行うだけで，TF を操作できるようにした．簡単な例として，「亀場」に四角形を描画する C プログラムの例を以下に示す：

```
#include "turtle.h"
main() {
   CON("localhost") ; /* TF との通信を開始する */
   CLR() ;     /* 画面をいったん消去する */
   RST() ;     /* 亀の位置を初期設定する*/
   PD() ;      /* ペンを下げる */
   FD(100.0) ;   /* 前進 100 */
   LT(90.0) ;    /* 左回転 90 */
```

```
    FD(100.0) ;   /* 前進 100 */
    LT(90.0) ;    /* 左回転 90 */
    FD(100.0) ;   /* 前進 100 */
    LT(90.0) ;    /* 左回転 90 */
    FD(100.0) ;   /* 前進 100 */
}
```

TF は単にグラフィック表示だけでなく, 例えば，課題結果の提出のため，キーまたはメニュー操作によって，描画の内容を (ユーザーID 等の情報と共に) PNG 形式でファイルに保存する機能なども組み込んである.

5 授業実践の例

開発したコードは，東北大学教育情報基盤センターが運用する情報教育システム（ICL システム）の Linux ブートイメージに含め，KDE の K メニューから開けるようにデスクトップ設定ファイルをインストールした．バイナリーファイルや画像等のデータを全て含めても，ファイル容量は 4M バイト程度にしかならないため, ディスクスペースへの負担はほぼ無視できる.

2014 年 4 月から 7 月にかけて, 東北大学の全学教育として実施されている「情報基礎」の 2 つのクラスにおいて，アカデミックスキルの修得を目的としたプログラミングに関係する授業において，TurtleEdit 等のツールを使用した．対象のクラスは農学部, 薬学部, 理学部 (生物) の 1 年生で構成されている．クラスあたりの学生数は 130 名を超え，160 台の PC 端末が設置された大教室で授業は実施された.

授業の大まかな流れは,〔1 回目〕基本的な概念と関連ソフトウェアの操作方法,〔2 回目〕C 言語の基本構造，画面出力,〔3 回目〕条件分岐,〔4 回目〕反復処理,〔5 回目〕多重ループ,〔6 回目〕関数で，C 言語の標準的かつ入門的な内容と言ってよい．6 回目に課題の説明を行った．それぞれの回では，プログラミング言語の説明や技法よりも，具体的な問題解決の手順を意識した例題に取り組むために，より多くの時間を割いた.

ほぼ同じ内容で実施した前年度の授業との違いについて，前年度もこの科目を担当した経験のあるティーチングアシスタント(TA) 3 名に質問したところ，以下のような回答（一部要約）を得た：

【質問 1】TurtleEdit を用いなかった前年度の授業と，TurtleEdit を用いた今年度の授業とで，受講者の質問やトラブルの内容に違いはあったか：

「あまり違いがなかったように感じる. ヘッダファイルの置き方の質問が増えたが，それ以外について，質問の内容は変わらなかったと思う.」
「『コンパイルってどうすればいいのか(コマンドを忘れた)』という質問はなくなった．その一方,『変更が反映されない』という質問はあった.」

【質問 2】TurtleEdit を用いたことで, TA の仕事が「やりやすくなった点」，および「やりにくくなった点」:

「端末上でコマンド操作を教える必要がなくなった点はやりやすかった．一方，端末を用いた説明ができなくなった.」
「コンパイル等の概念を説明せずに済んだ, という点においてはやりやすかったが，一方で，カレントディレクトリの概念が分かっていないせいか, 保存したファイルの紛失があった.」
「コンパイルや実行のやり方が分からないという質問は減った。初めて使った機能で不慣れな部分はあったが、（やりにくくなった点は）特になし。」

【質問 3】その他，授業で用いたツール（TE, TM, TF）とその利用法について，意見・感想：

「『フィードバック』機能は，講義進行にメリハリを与えると TA をやっていて感じた.」
「フィードバック機能については，（課題が）終わったら押すというのがあまり適当ではないかと思ったので, エディタと連携させて, 課題の亀の動作ができたら自動的に終了を送信した方が良い. それが技術的に難しいの

であれば，せめてエディタと同画面にボタンを配置するべきで，新たにウィンドウを開くのは煩わしい.」

また，受講者全員に対して毎年実施している学生による授業評価アンケート調査の自由記述欄には，著者(早川，酒井)の授業に対して，例年,「プログラミングは難しすぎる」といった主旨の記述が数件程度寄せられるが，TE の使用の如何に関わらず，その内容や傾向に目立った変化は見られなかった．また，ツールの使用についての記述も無かった．

以上をまとめると，少なくとも，学生は特段の困難やトラブル無く TurtleEdit 等のツールを使用できたものと考えられ，その意味で，こうした支援ツールが授業の中で前景化せず「黒子」的な役割を果たしたという点では，当初のねらいは果たされたと解釈することができる．

一方で，TA のコメントは，演習を通じて,「コンパイル」「作業ディレクトリ」など，ソフトウェア開発において必要とされてきた基本的な用語や概念を修得する機会が失われてしまった可能性も示唆している．ただし，このことが TurtleEdit 等のツールを用いることの弊害と言えるかどうかについては，科目の目指すところに依るところであって,「情報基礎」の目標のひとつである「コンピュータサイエンスの手法による論理的思考と問題解決ができるようになること」にとって，高い優先度で修得すべき事項とのバランスのもとで評価されねばならないだろう．また，反省点のひとつとして，TA に対して，こうしたツールを使用するねらいと，ツールの細かな動作について，事前に踏み込んで説明すべきであったと考えている．

また，授業を担当した TA からは「もっと踏み込んで，学生がどんなエラーを出しているのかもリアルタイムで集計できれば，面白いかもしれない.」とのコメントも寄せられた．授業目的とはいえ，受講者の活動を本人が知らぬ間に収集し利用することが適切か否かも含め，今後の検討事項のひとつと考えている．

6 まとめ

東北大学教育情報基盤センターでは，目下，情報教育用システムの更新作業中で，2015 年 4 月からは新システムが稼働を開始する予定である．

さらに,「情報基礎」の科目の内容も，新しい学習指導要領のもので「情報」を学んだ新入学者に対応すべく,「コンピューテーショナル・シンキング」[2]をひとつのキーワードとして，現在内容を見直している途上である．

本稿で紹介したソフトウェアツールは，次期システムにもインストールする計画であり，新しい「全学教育『情報基礎』第 4 版」の実施を支援するツールとしても，担当教員に活用を呼びかけ普及を図るとともに，アンケート等を通じて，その効果や改善点について分析を進めたいと考えているところである．

参考文献

[1] 磯辺秀司，小泉英介，静谷啓樹，徳山豪「情報基礎 A 講義ノート(平成 26 年度版)」および「情報基礎 B 講義ノート(平成 26 年度版)」，(学外未公開)，2014.

[2] 磯辺秀司，小泉英介，静谷啓樹，「コンピューテーショナル・シンキング」，共立出版，2014.