Knoppix 環境におけるプログラミング演習支援環境の構築と利用実験

船越亘[†1,a)]　　中野明[†1]

プログラミング言語の習得において演習授業は非常に重要である．しかしながら，1 人の教師による演習授業の運営は容易ではない．例えば，コンパイルする段階に到達していない学生，演習において同じような間違いが多発している状況など，良い演習を行うのに有益な情報があっても，演習実施中に教師がそれらの情報収集を行うことは容易ではない．
そこで，本研究では，実際の授業において Knoppix 環境における，プログラミング学習支援エディタを用いた，プログラミング演習支援環境の構築を行った．実際に制御情報工学科 2 年生 42 人に利用させ，データを収集し，データの解析を行った．

Experiment and Construct Programming Exercise Support Environment in Knoppix Environment

WATARU FUNAKOSHI[†1]　　AKIRA NAKANO[†1]

Exercise lessons are very important for learning programing languages. However, individual implementation of exercise lessons is not easy. Students are sometimes in trouble as follows: they do not reach to the compiling stage, they make similar mistakes in the exercise lessons, and so on. There is useful information for giving good exercise lessons but it is not easy for teaches to collect the information of students' trouble in exercise lessons. In this study, we aim to construct a program learning support environment using an editor in Knoppix to apply to exercise lessons.

1. はじめに

一人の教師が多数の学生を指導することは容易なことではない．複数人の学生には当然理解度の差が生まれてしまうため，教師はその中から理解のできていない学生を発見し，指導を行うことが望ましい．プログラミング演習授業においても同様で，理解ができていない学生や，課題が進まない学生を発見し，指導することで演習を円滑に進めることができると考えられる．

本研究の目的は，プログラミング演習の状況を把握するためのシステムの開発，また演習過程において，悩んでいる学生を見つけるための指標を発見することである．

2. Knoppix

本研究では，プログラミング開発環境を整える目的で Knoppix[†2] を採用した．Knoppix は USB メモリからの起動が可能であるため，PC の環境に依存せず，C 言語，Java といったプログラミング言語の開発環境があらかじめ整っている．また，学生が USB メモリを自宅に持ち帰った場合においても同じ環境を再現できるため，近年，教育現場にて活用されている．

3. システムの構成

Knoppix をインストールした USB メモリを利用者 1 人につき 1 つ準備し，その Knoppix 上で動作するようにプログラミング学習支援エディタを設定している．

3.1 プログラミング学習支援エディタ

本研究で使用したソフトウェアは，前年度までに開発された学生のプログラミング演習における情報を取得するエディタである[1][2]．このエディタは Java 言語で作成されたソフトウェアである．Java における外部プログラムを実行させる制御により，C 言語のコンパイル処理やプログラムの実行ができる．外部プログラムとして呼び出すコンパイラを変更することにより，C 言語以外のプログラミング言語演習に対応することが可能である．このエディタはプログラミング言語開発エディタの基本的な機能である include 文等のハイライト機能，tab キーを押したときのスペース間隔取得機能，新規作成，ファイルを開く，保存，コンパイル，実行等ができる．

Figure 1 はエディタの実行画面である，Figure 1 では C 言語のプログラミングを行っている．Figure 1 中の①はプログラミング言語開発エディタとしての機能であるファイルの“新規作成”，“ファイルの読み込み”，“ファイルの保存”，“コンパイル”，“実行”，“停止”，“チェック”の 7 つの機能に対応したボタン群である．実行ボタンを押して，プログラムが実行されるまで，停止ボタンとチェックボタンを押せないようにしている．これはユーザのミスの誘発を防ぐためである．また，②はソースコード記入エリア，③は入力データのログを表示するエリア，④は入力データエリ

†1　久留米工業高等専門学校
Kurume National College of Technology
†2　USB メモリで起動する Linux ディストリビューション
a)　a2918wf@std.kurume-nct.ac.jp

アである．実行ボタンがクリックされた際には，Figure 1 中の⑤の出力データエリアにプログラムの実行結果が表示される. ここで示される実行結果とはコンパイルの結果(エラーメッセージを含む)ならびにソースコードの実行結果である．

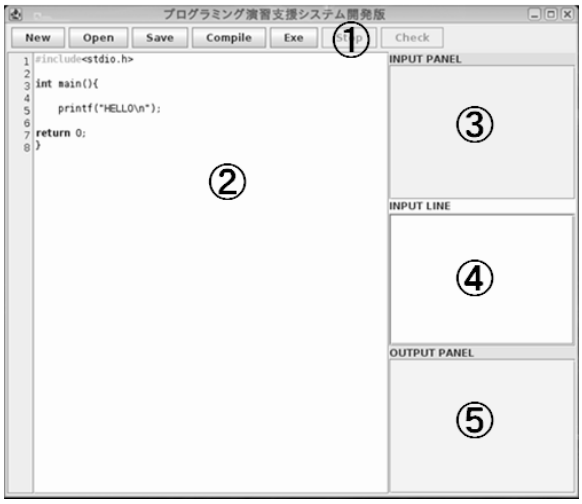

Figure 1 エディタの使用例

3.2 エディタの機能

システムの主要な機能[3]について以下に示す.

● 標準入力機能

実行ボタンがクリックされた際には，Figure 1 中の④の入力データエリアに標準入力することができる．また，③の入力データのログを表示するエリアには標準入力の履歴が表示される.

● 入出力ログ収集機能

時刻やファイル名，ソースコード，ボタンの操作の情報を各ボタンがクリックされるたびにテキストファイルに記録する.

● チェック機能

コンパイルとは別にチェックのボタンがある．あらかじめシステム側で正解の実行ファイルとデータセット用意しおり，チェックボタンがクリックされた際には，システム側でユーザが作成した実行ファイルと正解の実行ファイルにデータセットを標準入力させ，実行結果を比較し，実行結果が同じであれば正解, 異なるのであれば不正解とした. プログラミングに関するエラーについては以降のエラー判定機能の項で示す.

● 停止機能

停止ボタンがクリックされた際には，プログラムを強制終了させる．停止機能を追加した理由はプログラムが正常終了でない場合及び無限ループに陥った場合の対策である. メインプロセスによってプロセス監視スレッドが作成され, 外部プロセスが異常終了もしくは無限ループに陥った際にプロセス監視スレッドが外部プロセスを強制終了させる.

3.3 エラー判定機能

本システムでは，ソースコードのエラーを 3 つに分けて検出する．エラーの種類は，コンパイルエラー，ランタイムエラー，論理エラーの 3 種類である.

● コンパイルエラー

プログラミング言語で記述されたソースコードをコンパイラが機械語コードに変換する際に発生するエラーである．外部プロセスとして，C 言語のコンパイラを用いた際の結果から検出する.

● ランタイムエラー

プログラムの実行中に発生するエラーである．正常終了でない場合及び無限ループに陥った場合から検出する.

● 論理エラー

プログラムの実行結果が意図した通りにならないエラーである．前述チェック機能にて検出する.

エディタのコンパイルボタンを押した際に，コンパイラである外部プロセス gcc を起動させる．コンパイラがソースコードをコンパイルし，失敗した際にはテキストファイルに時刻，ファイル名，ソースコードをコンパイルエラーとして記録する．成功した際にはユーザは実行ボタンを押す. プログラム実行時にプログラムが異常終了した際には, テキストファイルに時刻，ファイル名，ソースコードをランタイムエラーとして記録する．正常終了した際はチェックボタンを押し，正解と比較し，異なるのであればテキストファイルに時刻，ファイル名，ソースコード，実行結果を論理エラーとして記録する.

3.4 取得できるデータの種類

プログラミング演習中のデータはテキストファイルに保存される.

● Buttonlog.txt

ユーザがエディタのボタンを押すたびに，ボタンを押した時刻とボタンの種類とファイル名を保存する.

● CCode.txt

ユーザがプログラムをコンパイル，実行，チェックもしくは停止させるたびに時刻とファイル名，ソースコード, 実行結果を保存する.

● Linenumber.txt

ユーザがエディタを使用し始めた時刻から, 10 秒間隔で時刻とファイル名，ソースコードを保存する.

4. 利用実験

実験は 2012 年 5 月 20 日，久留米工業高等専門学校制御情報工学科 2 年生の「プログラミングⅡ」の講義で行った. 学生数は 42 名であった. 学生にはプログラミングエディタとして，本研究で改良したプログラミング学習支援エディ

タを利用してもらった．開始前に学生に対して，本システムの使用法のデモンストレーションを行った．デモンストレーションでは，簡単な標準入出力を行うプログラムを実演した．この時，エディタの各ボタン，各入出力エリアの説明を行った．実験は久留米工業高等専門学校の演習室で行い，学生は 1 人につき 1 台の PC を利用した．

今回の実験では，制御情報工学科 2 年生が学んだプログラミングの範囲から以下 3 問を出題した．課題のプログラムは C 言語で作成するように指示した．

- 2 重の for 文を用いた標準出力(九九の掛け算)
- for 文, if 文を用いた標準入出力(2 値の和の桁数の標準出力)
- 配列を用いたソートアルゴリズム及び指定した配列の標準出力(配列に格納した値を降順ソートし，大きい方から 3 つの数を標準出力させる問題)

4.1 コンパイルの成功率

各問においてコンパイル回数の中でコンパイルに成功した割合を求めた．以降これをコンパイル成功率と呼ぶ．コンパイル成功率と課題達成時間の相関について調べた．結果を Table 1 に示す．

各問においてコンパイル成功率と課題達成時間との間には負の相関があった．このことから，コンパイル成功率が低い学生は課題達成時間が長くなる傾向があるといえる．

Table 1 コンパイル成功率と課題達成時間の相関係数

コンパイル成功率	問 1	問 2	問 3
相関係数	r=-0.8643 (p＜0.05)	r=-0.5868 (p＜0.05)	r=-0.5359 (p＜0.05)

4.2 コンパイルまでの時間

各問において初めてコンパイルするまでの時間を求めた．以降これを初コンパイル時間と呼ぶ．初コンパイル時間と課題達成時間の相関を調べた．結果を Table 2 に示す．

各問において初コンパイル時間と課題達成時間との間には正の相関があった．このことから，初コンパイル時間が長い学生は課題達成時間も長くなる傾向があるといえる．

Table 2 初コンパイル時間と課題達成時間の相関係数

初コンパイル時間	問 1	問 2	問 3
相関係数	r=0.7817 (p＜0.05)	r=0.4621 (p＜0.05)	r=0.7498 (p＜0.05)

4.3 文字数の変化

10 秒おきの文字数の変化と各問の課題達成時間との関係を見るために，相関分析を行った．Figure 2 は各経過時間における「課題達成時間」と「その経過時間までに入力した文字数」との相関係数をグラフにまとめたものである．横軸は経過時間，縦軸はその時の相関係数である．各問において文字数と課題達成時間との間には負の相関があった．このことから，入力した文字数が少ない学生は課題達成時間が長くなる傾向があるといえる．また，$p<0.05$ かつ相関が高い時間帯はおおよそ 100 秒を越えたあたりから見られる．このことは演習が始まって 100 秒経過したあたりの文字入力の量で達成時間の推測ができることを示している．

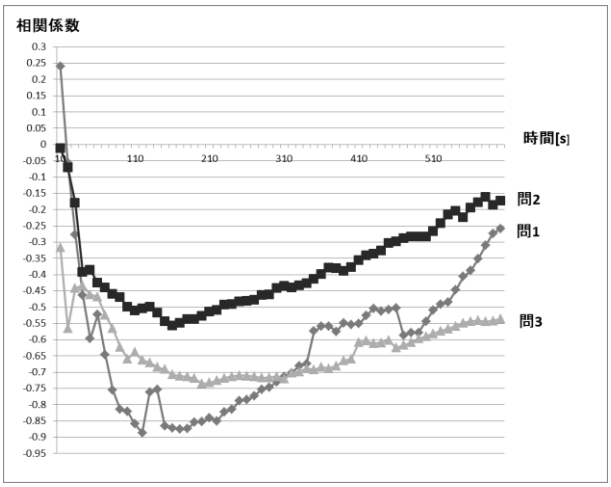

Figure 2 文字数と課題達成時間との相関係数

5. おわりに

本研究では授業において使用する実験を行い，学生のプログラミング演習におけるデータを収集した．さらにデータを解析し，悩んでいる学生を発見するための指標の可能性のあるものを得た．しかしながら，授業においてエディタを使用することはできたが，指標といえるものを得るにはデータ数が少ないといった問題や USB メモリからのデータの収集が手作業といった問題がある．

今後は授業・自宅学習でのプログラミング演習におけるデータを収集し，標本を増やす．また，サーバ・クライアント通信システムにプログラミング学習支援エディタを実装することによってデータの収集を容易にすることが課題である．

参考文献

1) 中野明，“プログラミング言語演習支援環境の設計と開発”，久留米工業高等専門学校記要第 23 巻 2 号 pp.55-62，2008.
2) 松尾知加子, “プログラミング演習における教師支援システムの作成”平成 20 年度久留米工業高等専門学校制御情報工学科卒業論文，2008.
3) 田中昭裕, “プログラミング教育におけるブロック判定による論理エラーの発見”，平成 20 年度久留米工業高等専門学校制御情報工学科卒業論文，2008.