BugTutor：誤情報を活用したプログラミング学習システム

岩渕悠太[†1]　高島健太郎[†1]　西本一志[†1]

概要：企業の ICT 新人研修において，プログラミング教育が行われている．プログラミングは，一部の初学者にとって難しい場合があり，知識の定着率格差が問題になっている．このため，プログラミング学習支援システムがこれまでに多く開発されてきた．しかし，この問題の根本的な原因は，従来の学習方法が「受動的な学習」となっていることにあるのではないかと我々は考えた．そこで本研究では，学習教材に誤情報を混入し，これを正しく修正する学習を行わせることで，受動的な学習を回避し，能動的な学習を促すシステムを開発する．本稿では，開発したシステム BugTutor について述べると共に，これまでに実施したユーザスタディに基づく初期的な有効性の検証結果を報告する．

1. はじめに

日本では今後，IT 人材の不足が深刻化する．経済産業省のデータによれば，2020 年には約 37 万人，2030 年には約 79 万人の IT 人材が不足すると推測されている[1]．そのため，IT 人材育成は必要性が高く，意義があるものである．

多くの初学者にとってプログラミングは難度が高く，知識の定着率（教授された事柄が一定期間以上記憶されている割合）が低いことが問題となっている．そこでこれまで，初学者のプログラミング学習を支援するためのシステムが，数多く開発されてきた．西田らは，ソースコードの入力支援や，プログラムの実行状態表示機能を持つ，DNCL を拡張したプログラム学習環境 PEN[2]を実装した．井垣らは，各学習者のコーディング状況を記録・可視化し，進捗状況を講師に提示することで，遅れている学習者に対して講師が個別指導を行うことができる，コーディング過程可視化システム C3PV[3]を実装した．田口らは，プログラミング教育において，学習者の理解状況，並びに学習者の学習意欲に応じて，演習科目に反映させるプログラム教育の支援[4]を提案している．

本研究では，IT 人材育成の一環として，企業の新人研修において行われている ICT 教育に着目し，そこでのプログラミング教育を支援するためのシステムを構築することを目指している．本稿第 1 著者は，新人研修でプログラミング教育を担当した経験がある ICT 系企業 3 社の社員 7 名に対して，プログラミング教育の問題点をインタビューした．共通して指摘された問題点は，社内講師 1 人に対して複数人の研修生で学習を行った場合に知識の定着率が特に低いこと，並びに，社内講師が抱えている本来業務を実施するための時間が奪われることであった．よって，ICT 教育の職場内職業訓練（OJT）における研修生の知識定着率を効率的に向上させる教育手段の実現が求められている．

OJT 教育におけるこのような問題は，講師側から一方的に指導が行われる受動的な学習に原因があり，その解決には能動的な学習手段の導入が必要であると，筆者らは考えている．杉山ら[5]は，アクティブラーニングを導入することで，受動的な学習を能動的な学習に転換することを試みた．その結果，アクティブラーニングを導入したクラスが，テスト成績において高い得点をとることを示した．しかしながら，プログラミング学習を支援する既存の研究では，受動的な形態での学習支援が多く，能動的な学習によって，学習効率の向上を目指す提案は，少ない．

本稿では，企業研修生のプログラミング教育において，能動的な学習を導入することにより，初学者の知識定着率を向上させる手法 BugTutor を提案する．提案手法を用いた実証実験を実施し，提案手法の有効性に関する基礎評価を行う．

2. 提案手法

本研究では，能動的な学習を行わせるため，誤った情報を含んだ教材で学習させる e-ラーニングシステムを提案する．プログラミング学習において一般的に採用されている講義形式を e ラーニング形式に変更するのは，社内講師の拘束時間を少しでも削減し，講師の本来業務の妨げにならないようにするためである．また，誤った情報を含んだ教材で学習させるのは，知識の定着率を向上させるためである．具体的には，学習内容の説明文には正しい情報を与えるが，サンプルプログラムに誤情報（バグ）を混ぜ込んだものを提示して学習を行わせる．

初学者の学習について，誤情報を含んだ教材の有効性を示した先行研究が Adams らによってなされている[6]．この研究では，小数点の大小を比較する問題を課題として，正しい情報に基づいて学習を進めたグループと，誤った情報を含んだ状態で学習を進めたグループを比較している．学習後，短い時間の後に実施したテストでは，両者の成績に有意な差は見られなかった．しかしながらが，数日後に再度テストを行った際には，誤情報を含んだ学習を行ったグループの成績が有意に良いという結果が得られている．このような結果となった理由として，エラー処理を行うことによって，能動的な学習が体験できた可能性が指摘されている．

†1 北陸先端科学技術大学院大学　先端科学技術研究科
Graduate School of Advanced Science and Technology,
Japan Advanced Institute of Science and Technology

なお，当然のこととして，初学者に誤情報だけを与えて学習を進めてもらった場合，正しい情報を覚えられないことが想定される．一定時間後に，誤りの情報について正しい情報を提供する必要がある．そのため，本研究では，学習が始まってから 15 分後に正しい情報を提供するようにする．

3. 実験システム BugTutor

2 章で述べた提案手法に基づき，初学者を対象としたプログラミング学習支援システム BugTutor を構築した．BugTutor は，Python 3.6.8 を使用して構築された，Windows OS 対応の GUI ソフトウェアであり，学習・解説・演習の 3 つのフェイズで構成されている．本システムは，提案手法の有効性を検証するためのプロトタイプであるため，システム内にプログラム実行環境を搭載していない．そのため，web 上で Python3 を実行できる，PaizaCloud を利用した．

3.1 実験システム構成

BugTutor のシステム構成利用環境を図 1 に示す．ノート PC にはシステム画面を表示し，ディスプレイには PaizaCloud を表示する．

3.2 学習フェイズ

BugTutor での学習を開始すると，まず学習ページが提示される. 学習ページの例を, 図 2 に示す. このページでは，2 章で述べた提案手法に基づき，if 文などの当該ページにおける学習項目についての正しい説明文と，バグが埋め込まれたサンプルプログラムが提示される．なお，埋め込むバグとしては，たとえば if 文の学習であれば，比較演算子の “==” が “=” になっていたりするような, 当該学習項目で初心者がよく作り出すバグを埋め込んでいる. 学習者は，解説文を読んで当該ページにおける学習内容を理解するとともに，示されたサンプルプログラムを PaizaCloud 上で打ち込んで，動作を確認して学習する．しかし，サンプルプログラムにはバグが埋め込まれているため，そのまま打ち込んでもエラーを起こして動作しない．学習者には，バグをとって，サンプルプログラムが動作するように修正することが求められる．最終的にサンプルプログラムを動作するよう修正できたか確認するために，実行結果を送信する機能を用意した．

3.3 解説フェイズ

解説ページでは，正しいサンプルプログラムとバグの説明を提供する．解説ページは，自動的に表示されるのではなく，学習ページに用意されたボタンを学習者が押すことによって表示される．ただし，学習ページでの学習開始から 15 分経過していない状態でボタンを押した場合は，「もっと考えてください」とだけ表示され，解説ページは表示されない．

図 1　BugTutor の利用環境

3.4 演習フェイズ

学習フェイズで与えられたバグ入りのサンプルプログラムが正しく修正され，想定通りの結果が送信された場合，演習フェイズに移行する．演習フェイズで表示される演習ページの例を図 3 に示す．演習ページで提示された演習課題について，ヒントを参考にしながら PaizaCloud 上でプログラミングを行い，正しく動作すると思われるプログラムを作成する．できあがったプログラムのソースコードを画面左下の入力領域に貼り付けて送信ボタンを押し，またその実行結果を画面右下の入力領域に貼り付けて回答ボタンを押す．入力された情報がともに正しければ，Congratulations! と表示され，次の学習項目に関する学習ページに移行する．

4. 評価実験

評価実験では，提案する学習支援システムの有用性を検証するために，プログラミング初学者に対して，BugTutor を用いる被験者 2 名（A，B）の利用グループと，BugTutor を用いない 2 名（C，D）の非利用グループに分けて比較実験を行った．非利用グループには，学習ページで提示するサンプルプログラムにバグを埋め込んでいないものを提示して学習を行わせた．実験で使用する学習対象言語は，Python3 とした．

4.1 実験手順

両グループとも, 事前準備として, Python3 でのプログラミング経験のアンケートを行った．結果，被験者全員，プログラミング経験はないと述べた．アンケート終了後，BugTutor の利用手順を説明し，学習を行ってもらうよう求めた. 学習項目は, 変数の計算・if 文・for 文の 3 つである．いずれのグループについても，サンプルプログラムと課題を正しく実行できなければ，次の学習に移れないようにした．学習フェイズでは，時間制限を設けず自由に学習できることとした．被験者には，以下の教示を与えた：

- インターネットでの検索は行わないこと
- 演習フェイズに移行する前に，PaizaCloud のエディタ内のソースコードは消去すること

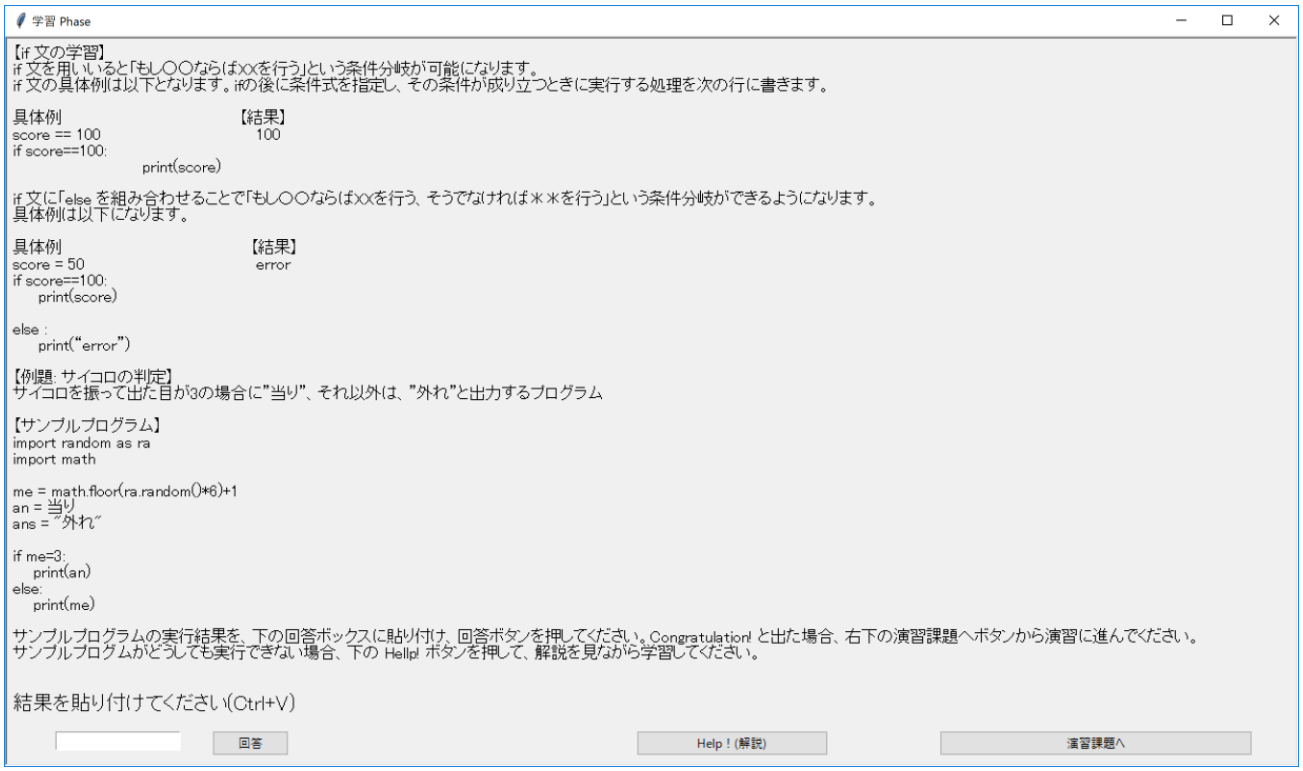

図 2　学習ページの例

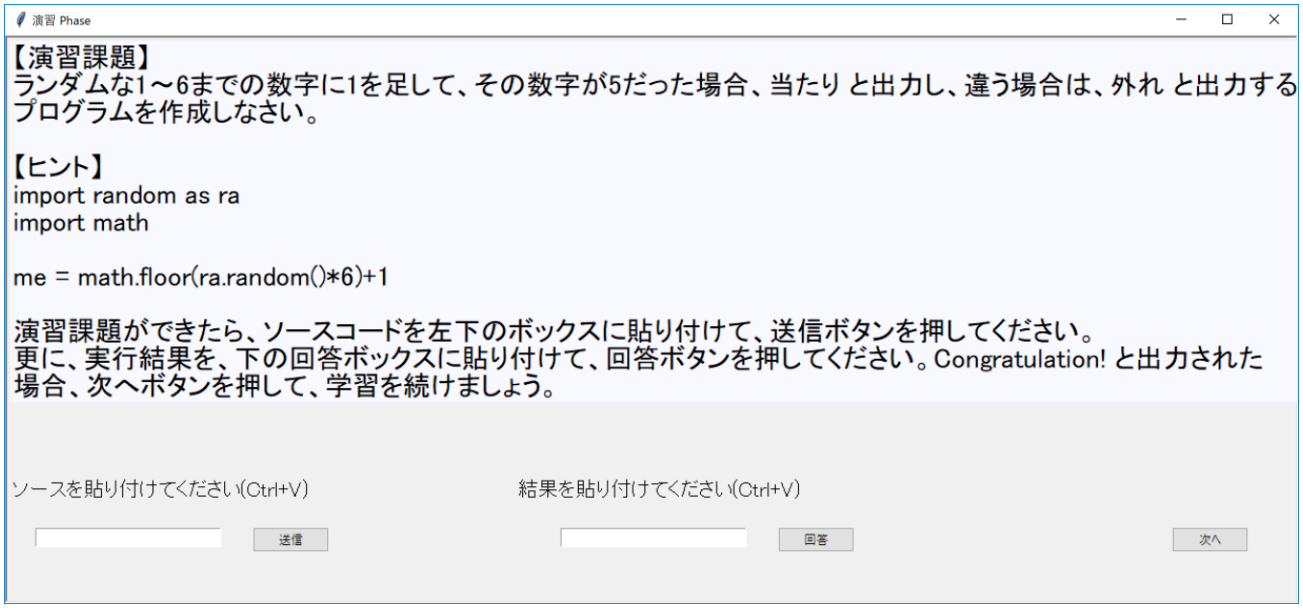

図 3　演習ページの例

- 演習フェイズで，30 分経過しても課題が解けない場合，自己申告すること

以上の学習が終了した 5 日後に，記憶の定着率を測るためのテストを実施した（以降，Delay テスト略す）．テストの内容は，for 文の課題とした．

4.2 実験結果

利用グループおよび非利用グループのそれぞれについて，学習項目ごとの学習フェイズに要した時間と演習フェイズに要した時間を，図 4 および図 5 に示す．横軸は各被験者の変数・if 文・for 文の学習項目であり，縦軸が要した時間である．利用グループの学習フェイズに要した総時間は 218 分，演習フェイズに要した総時間は 56 分であった．一方, 非利用グループの学習フェイズに要した総時間は 50 分, 演習フェイズに要した総時間は 100 分であった．被験者が

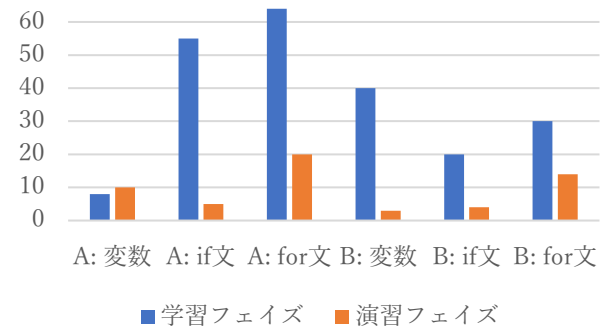

図 4　利用グループの学習項目毎の学習時間（単位は分）

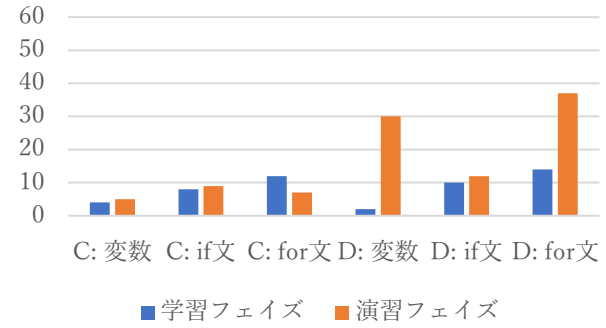

図 5　利用グループの学習項目毎の学習時間（単位は分）

それぞれのグループに 2 名ずつしかいないため，統計的な検定を行うことはできないが，学習フェイズに要した時間は利用グループが非利用グループよりも大幅に長く，逆に演習フェイズに要した時間は非利用グループの方が利用グループよりも長いという結果になった．なお，演習フェイズで，演習課題を達成できず自己申告した被験者は，システム非利用の被験者 D のみであった．達成できなかった課題は，変数の計算と for 文であった．

実験後，インタビューを行った．変数計算のサンプルプログラムを動かし，すぐ演習課題に進んだ結果，書き方を覚えていなかったと被験者 D は述べた．次に for 文の課題については, for 文の使い方を覚えたと思い課題に臨んだが, for 文で使用する range の書き方を忘れてしまったと述べた．

Delay テストの結果は，利用グループの両被験者が課題を達成することができたのに対し，非利用グループでは両被験者が課題を達成することができなかった．

学習中に与えられたサンプルプログラム以外に，コードを書き学習した被験者は，A，B，C であった（変数の計算学習中に，サンプルにはない足し算，引き算の実行など）．

5. 考察

利用グループと非利用グループの学習時間ならびに演習課題時間，および Delay テストの結果より，BugTutor の利用者は，プログラミング学習において，記憶定着率が向上したと考えられる. 向上した理由として, 受動的な学習を，能動的な学習に転換できたことが推察される．

非利用グループの被験者 C は，学習時間と，演習課題において利用グループと同等の結果を残している．この被験者は，自発的にサンプルコード以外のプログラムを作成して動かし，声に出しながら学習を行っていたことが観察された．つまりこの被験者は，自発的に能動的な学習を行うことができていたと考えられる．

6. おわりに

本研究では，既存のプログラミング学習の問題である受動的な学習を，能動的な学習に転換する，プログラミング学習支援手法を提案し，この手法に基づいて学習支援システム BugTutor を構築した. BugTutor を用いたユーザスタディを実施し，その効果や影響を調査した．結果，提案手法によってプログラミング初学者の学習量が増加し，記憶定着率が向上する可能性が示唆された．

実験は，現在も進行中であり，今後は，更に被験者を増やし，実験を行うことで，提案手法の有効性に関するより精密な検証を進めたい．また BugTutor については，被験者より改善案を頂いている．GUI デザインの改善や，ソースコードの送信後に，送信されたかどうかを判断できるフィードバック機能の追加などが求められている．これらの機能を追加し，より実用的な学習支援システムとしていきたい．

謝辞　本研究関わってくださったすべての方々に，心より感謝いたします．

参考文献

[1] “IT 分野について”. https://www.meti.go.jp/committee/kenkyukai/shoujo/daiyoji_sangyo_skill/pdf/001_06_00.pdf, (参照 2019-12-10).
[2] “プログラム入門教育用学習環境 PEN”. http://id.nii.ac.jp/1001/00054235/, (参照 2019-12-10).
[3] “プログラミング演習における進捗状況把握のためのコーディング過程可視化システム C3PV の提案 “. http://id.nii.ac.jp/1001/00088682/, (参照 2019-12-10).
[4] “個々の学習者の理解状況と学習意欲に合わせたプログラミング教育支援”. http://id.nii.ac.jp/1001/00010078/, (参照 2019-12-10).
[5] “アクティブラーニングの学習効果に関する検証-グループワーク中心クラスと講義中心クラスの比較による-“. https://core.ac.uk/download/pdf/59174936.pdf, (参照 2019-12-10).
[6] “Erroneous Examples Versus Problem Solving: Can We Improve How MiddleSchool StudentsLearn Decimals? “. http://www.cs.cmu.edu/~bmclaren/pubs/AdamsEtAl-AdaptErrExStudy-CogSci2012.pdf, (参照 2019-12-10).