6ZD-6

初学者のためのプログラミング言語学習支援システムの構築

富川葵†　須藤智†　樫村雅章†　恩田憲一†
尚美学園大学芸術情報学部情報表現学科†

1. はじめに

本学には、必修の科目として C 言語を学ぶ実習授業があり、多くの学生はその授業で初めてプログラムに触れる。

その授業のアシスタントの経験から、学生によって理解度に差があることがわかった。理由としては、プログラムを理解しながら入力できていないことが大きい。

今回はこのような、自分がどのようなプログラムを打っているのかを理解できていない初学者に向けての、システムの構築をめざす。

2. なぜ理解できないのか

サンプルプログラムをまる写しし、内容を充分把握せずに入力しているため、プログラムの理解が進まず、その結果、応用が効かないという実態がある。例えば、printf 関数内の”(ダブルクォーテーション)の後には%d がなくてもとりあえず「,変数名」を書くなどの自分勝手な決まりを作っている例が多い。「¥n」や「%d」の使い方などもなんとなくこのあたりに書けばいいだろうというように意味をよく理解せずに書いている。また、標準関数名や予約語だけではなく、プログラム内の１つ１つの要素に関しても、意味を考えずに入力している傾向がある。

そのような学生は、「変数を宣言するにはどのようなプログラムを書けばいいのか」、「文字を出力するためにはどのようなプログラムを書けばいいのか」というように日本語でやりたいことを言うことはできるが、プログラム上での表現の仕方がわからず、結果的にプログラムに対して苦手意識を持ってしまう。

プログラムの中の１つ１つの動きを理解し、覚え、使えるようにならなければ、プログラムを思い通りに書くことはできない。そのためには、まず最低限のサンプルを理解できるようにならなくてはならない。

A learning support system of programming language for beginners
Aoi Tomikawa , Satoshi Sudo ,
Masaaki Kashimura , Norikazu Onda
†Shobi University

3. システム

3.1 本システムの対象および範囲

対象はプログラミングに苦手意識のある初学者とする。

学習内容の範囲は、C 言語の習得をめざす初学者向けということで、宣言文、代入文、printf 関数、if 文に限定する。

3.2 可視化

学生の理解度を向上させること、および苦手意識の克服を実現させるためにどのような手法が望ましいのかを考えた結果、可視化という手法を用いた。

プログラミング学習における可視化では以下のような効果が期待できる。

- 目で見てプログラムの動きを理解できる
- 自分が組んだプログラムを、頭で考えるだけでなく見るという別の視点から考察することができる

本システムの具体的な可視化方法は大きく２点である。

① １つ１つの要素がプログラムの中でどのような働きをするのかを日本語により表示させる
② 逐次表示を行う。

①は、アルファベットや記号の並びであるひとつひとつの要素を日本語で表示させることにより、初学者でもプログラムの理解ができやすくなるという効果が見込まれる。

②はプログラムを書く順番がわからず、苦戦するという初学者が陥りやすい問題を解決するために行う。逐次表示を行うことにより、自分がどのような流れでプログラムを書いたのか理解できるようになるという効果が見込まれる。

エラーが起きた場合は、可視化を正しく動いている箇所までで中断する。これによりエラーが起きた箇所を明確に示すことができ、初学者の理解度向上を促すことができる。

また、表示にてプログラムを書く際のアドバイスを示し、自分の書いたプログラムの記述方法が本当に正しいのかを確認させる。

3.3 表示画面

メインウィンドウ(図１)でプログラム実行後に可視化ウィンドウ(図２)を表示させる。

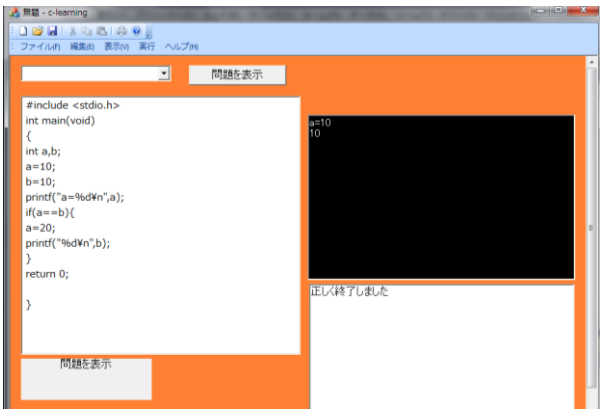

図１.メインウィンドウ

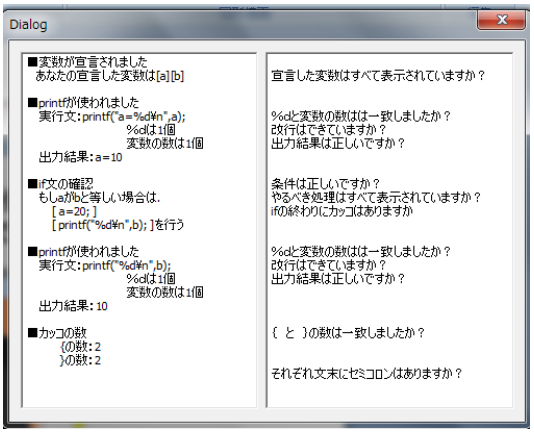

図２.可視化ウィンドウ

4. 解析方法

初学者にありがちなエラーをあらかじめ設定し、学習者が書きこんだソースコードをキーワードで探索して、見つかったキーワードごとにエラーの解析を行う。

宣言文や代入文には必ず文末に「;」(セミコロン)が必要であり、「｛」と「｝」の数は合わせなくてはならない。printf 関数には「printf(“書式制御文字列”,変数リスト);」、if 文には、「if(条件)｛実行文｝」というような決まった文法が存在する。学習範囲を限定することにより、初学者にありがちな文法のまちがいを推測することができ、エラーの指摘が可能になる。

例えば、printf 関数であれば、「printf」の直後には必ず「(」「”」と続き、文末には「;」が必要となる。printf 関数をキーワードとして探索し見つかった場合には printf に続く 1 文を切り取り、正しい文法に必要な要素があるか調べる。もし要素が見つからなかった場合はエラーとなるので、メインウィンドウでエラー表示を行い、可視化を停止する。

これらの解析をキーワードごとにすべて行いエラー表示、可視化の停止を実行する。

5. ヘルプメニューの作成

どのようにプログラムを書けばいいのかわからないという初学者特有の問題を解決するために、プログラム作成に活かせるヘルプメニュー(図３)の作成を行った。

キーワードで用語を検索することができるもので、項目としては以下の 4 点に分けられる。

- ●名前で探す
  …[printf][int]などの使い方を表示
- ●やりたい事柄から探す
  …[変数を宣言する][条件を指定して処理を行う]等の事柄の解説
- ●意味を調べる
  …[変数とは][初期化とは]などの意味を解説
- ●その他
  …演算子の使い方などの解説

図３.ヘルプメニュー

それぞれの要素の使い方を解説する際には、例題をいくつかのせ、実践に活かせるようにする。

6. まとめ

本システムは、自分がどのようなプログラムを組んだのかを可視化によってわかりやすく理解することができ、理解度向上やプログラミングへの苦手意識克服にもつながるものである。

授業アシスタントをしている際に、おなじような質問を繰り返す学生や、ずっと悩んでいた結果練習問題が一つも完成しなかった学生、また、そもそも学習への意欲が見られないような学生を多く見てきた。そのような学生にプログラムを理解させるためにはどうすればいいのかを考えたことから本システムは生まれた。

本システムを使用することで、学生がそれぞれどのようなプログラムを組んでいるのかを理解でき、結果としてプログラミングへの苦手意識がなくなり、プログラムを組むことが楽しいと思ってもらえるようになることを期待したい。