N-003

Cプログラミングにおける定型コーディングフォームを用いたスタイル学習支援システムの開発
Development of style learning support system using standard coding form in C programming

橋本 浩規† 納富 一宏†
Kohki Hashimoto Kazuhiro Notomi

1. はじめに

神奈川工科大学の情報工学科では，統合開発環境（IDE: Integrated Development Environment）を用いたC言語プログラミング教育が行われている．このようなプログラミング教育では，講義と演習を組み合わせ，配列や関数といった文法についての講義を行い，それらに関する演習問題に解答するという流れが一般的である．

一方でこうした類型の教育では，プログラミングにおけるソースコードの記述方法に関するルールである「コーディングのスタイル」を身につけることは難しい．初学者を対象とするプログラミング教育ツールや学習システムの開発の先行研究としては，立花ら[1]は特定のプログラムの読解補助システムを，大門ら[2]は特定のプログラムの問題自動生成・自動採点システムを提案している．しかし，コーディングのスタイル学習を目的とした汎用的なシステムを提案する例は見られない．

上記のような教育で文法を理解できても，自由自在にコーディングできるとは限らない．文法を学んだ後の効率的なスキルアップの方法として，コーディングスタイルを学習できるようサポートすることが本システムの目的である．

2. コーディングフォーム

2.1 コーディングフォームの定義

ある特定のプログラムを記述する際に，そのソースコードの形（構成）を決めるものを「コーディングフォーム」と呼ぶことにする．

コーディングフォームは作成するプログラムの形（構成）を決めるための「ガイド」であり．ガイドに従ってコードを記述することでソースプログラムを完成させるようにアシストする．ここで，ガイドという用語をひな型やデザインパターン，あるいはフレームワークと同義または類義語として用いている．コーディングフォームはある特定のプログラムごとに用意し，それを「定型コーディングフォーム」と呼ぶ．特定のプログラムに属するものであれば，どのようなプログラムでも対応可能な汎用的なものを目指す．

プログラミング言語によってはソースコードの構成要素の種類や形は異なるため，それらを考慮してコーディングフォームを構成する必要がある．本研究ではC言語を想定した定型コーディングフォームを検討している．

C言語によるソースコードの構成要素を表1に示す．

表1　C言語によるソースコードの構成要素

グループ名	構成要素
プリプロセッサ	インクルード
	マクロ
データ構造	データ構造（列挙/構造体/共用体, クラス）
	グローバル変数宣言
関数	プロトタイプ宣言
	ファンクション/関数定義
	メイン関数定義

2.2 デザインパターンとの違い

デザインパターンと根本的なアイディアは同じだが，コーディングフォームはスタイル学習のためのものであり，初心者が中級レベルに到達しやすくするためにサポートする仕組みである．デザインパターンは，規模の大きい実践的なプログラム設計・開発に対応するものだが，初心者には複雑で種類が多いため敷居が高い．コーディングフォームは，デザインパターンよりは制約や制限が大きく限定的であるが，シンプルである分，学びやすいと思われる．

3. コーディングフォームを用いた学習システム

システムの構成について図1に示す．

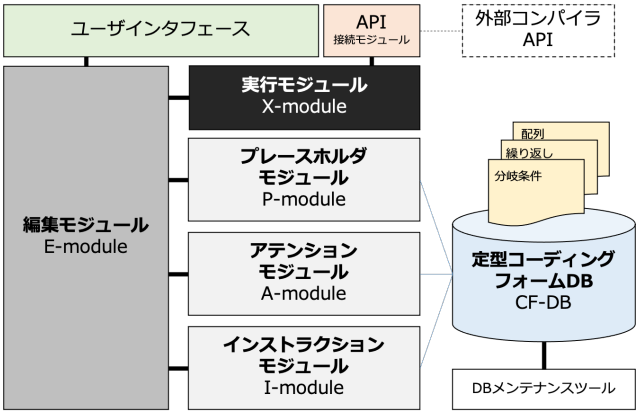

図1　システムの構成

システムは，下記のモジュール（機能）で構成される．

①プレースホルダ（placeholder）：ソースコードの各構成要素を記述する場所をユーザに提示・指示し，コーディングの誘導を行う

②アテンション（attention）：コーディングに対する誘導や指示，説明や情報の提供を行う

†神奈川工科大学　Kanagawa Institute of Technology

③インストラクション（instruction）：サンプルコードの提示や解説動画の提供を行う

提案システムでは表 2 に示す 9 つの定型コーディングフォームを搭載予定である．

また，現在，言語の文法を学ぶ入門者にとって，ソフトウェア設計やプログラミングは難解である．文法により規定される最低限のコーディングのルールの習得からスタートするが，コーディングスタイルの学習は重要である．そこで，入門者に対応した機能をシステムに追加することが重要である．

表 2　搭載予定のコーディングフォーム

番号	単元名	内容
CF01	条件分岐	if, switch
CF02	繰り返し	for, while, do while
CF03	配列	1 次元配列, 2 次元配列
CF04	文字・文字列操作(1)	ctype.h 内の関数利用
CF05	文字・文字列操作(2)	string.h 内の関数利用
CF06	関数	関数宣言, 呼び出し
CF07	ファイル処理	ファイルオープン, クローズ, 読み書き
CF08	構造体	構造体定義/利用
CF09	ポインタ	ポインタ変数・配列, 関数での利用

4. 実験

3 で述べたシステムの機能のうち，①と②を持つプロトタイプ（以下，CF という）を開発している．

本実験では，大学の授業で行われるプログラミングの演習問題と類似した問題を解く際に，CF の機能によって Visual Studio（以下，VS という）よりも適切なプログラム構造へ誘導し，プログラムを組みやすくなるかを実験した．

実験協力者として本学学生 6 人を 2 つのグループに分け，表 3 に示した順番でコーディングを実施してもらった．

表 3　グループごとのコーディングの手順

グループ	問題 A		問題 B	
	1 回目	2 回目	1 回目	2 回目
G1	CF	VS	CF	VS
G2	VS	CF	VS	CF

CF：コーディングフォーム　VS：Visual Studio

実験協力者には，それぞれのコーディング時間と正解までのコンパイル・実行回数を記録してもらった．実験終了後はアンケートを実施し，協力者ごとの C 言語のスキルと，CF の評価（プログラムの組みやすさ，ヒント提示の評価，今後の利用意欲，改善点）を訪ねた．

実験は，実験者の PC を使って行い，CF を含む開発サーバーをローカルで起動させ，ブラウザから localhost へアクセスして使用してもらった．

5. 結果

問題ごとの「VS と CF のコーディング時間の差と，後述するアンケートの評価値との散布図を図 2 と図 3 にそれぞれ示す．これらからコーディング時間は，VS で先に解いた人は同じか VS の方が，CF で先に解いた人は同じか CF の方が時間がかかっていることがわかった．

次に，実験後のアンケートでは，「VS と CF のどちらの方がプログラミングしやすい（コードを組み立てやすい）か」の問いに対し，全員が，「VS」または「やや VS」と回答した．また，「ヒントの提示は適切であったか」の問いに対し，約 6 割が「適切」または「やや適切」と回答した．また，「今後もこのシステムを使ってプログラミングしたいか」の問いに対し，約 6 割が「思わない」または「やや思わない」と回答した．

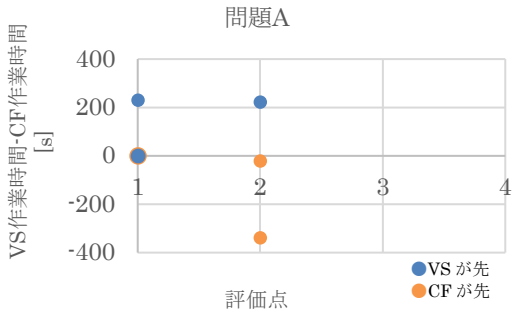

図 2　問題 A の散布図

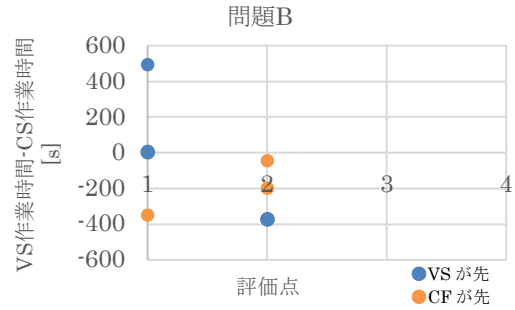

図 3　問題 B の散布図

6. 考察・まとめ

5 に示した結果のうち，図 2 と図 3 で先に解いた環境の方が時間が長いのは，同じ問題を 2 回解いたためであり，CF によって速まったとは言えないと考える．またアンケートの結果からは，プログラミングの構造を誘導するヒントは一定の評価を得られたと考えるが，今後も機能の改善が必要と考える．また VS には自動インデントや予測変換の機能があることから，プログラミングのしやすさは VS が有利だったと考える．また「ある程度ヒントのバリエーションが欲しい」といった意見があり，考えている構造と提示する構造の相違が利用意欲を下げている可能性がある．

本実験では学習システム開発の第 1 段階として，3 に述べたモジュール①と②を持つプロトタイプを試作し，適切なプログラム構造への誘導効果について検証した．

今後はモジュール③の実装や，モジュール①と②の改善，学習機能の強化を目指す．

参考文献

[1] 立花 昂己, 松澤 芳昭, “プログラミング初学者のための読み下し文提示システムの開発と評価”, 研究報告コンピュータと教育（CE）, 2022-CE-164, 8 (2022).

[2] 大門 巧, 大西 建輔, “Google フォームを利用したプログラミング問題の生成システムの開発と評価”, 研究報告コンピュータと教育（CE）, 2021-CE-161, 1 (2021).