社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2006－CE－86 (2)
2006／10／21

実行テストを用いたコンテスト形式の入門的Cプログラミング演習の大会運営サーバの開発

倉田 英和, 富永 浩之, 林 敏浩, 山崎 敏範

香川大学工学部 〒761-0396 香川県高松市林町 2217-20
E-mail: s06g462@stmail.eng.kagawa-u.ac.jp

あらまし 大学情報系学科の入門的C言語教育において，毎回の授業の活性化のため，コンテスト形式によるプログラミング演習を提案する．解答となるソースコードをアップロードさせ，コンテストの途中経過を公開する大会運営サーバを開発する．評価は，入出力サンプルによる判定テストで行い，解答時間に基づいて得点を定める．初心者への中間目標として，複数の予備テストも用意し，テストを習慣化させる．これらにより，競争による学習意欲の向上と，問題解決の達成感を与えることを目指す．

キーワード C言語演習，プログラミングコンテスト，実行テスト，チーム競争，大会運営サーバ

Contest Style Exercise and Support Server with Execution Tests for Introductory C Programming

Hidekazu KURATA, Hiroyuki TOMINAGA, Toshihiro HAYASHI, Toshinori YAMASAKI

Faculty of Engineering, Kagawa University 2217-20 Hayashi, Takamatsu, Kagawa, 761-0396 Japan
E-mail: s06g462@stmail.eng.kagawa-u.ac.jp

Abstract C language is the necessary subject for computer engineering education. For effective and enjoyable exercise in daily lesson, we propose C programming exercise with group contest in classroom. We adjust a contest style for beginners. We develop tProgrEss, the contest management Web server. In the contest, a student uploads his source code to the Web server. We offer two stages of execution test by given some input and output sample data. The several preparation tests are step-by-step subgoals. They suggest a guideline of coding with program structure. They make students have custom of testing. The current situation of the tests is opened in the Web page to raise competition volition. The achievement test is the final check to give adequate points concerned with answering time and trial frequency. The purpose of our research is to promote their learning motivation with team competition. And it is to give sense of satisfaction and achievement with solving problems.

Keyword C language exercise, programming contest, execution test, team competition, support server

1. はじめに

大学の情報系学科の多くは，初年次にC言語の入門的な科目を必修としている．C言語には，説明すべき文法事項が多く，授業中に十分な演習時間を確保できないことが多い．また，個人による理解度の差が大きく，多人数の授業を円滑に進めることが難しい．そのため，各自のレベルに応じた課題の提示や，段階的な目標を与えて，限られた授業時間内に達成感を与えることが必要である．

また，演習の効果を高めるためには，授業中の取組み方が大切となる．演習後，予習復習としての自宅学習を行うことも重要である．しかし，演習授業では，次のような場合が多く見られる．授業中に演習解答を諦めると自宅学習も行わない．自由な演習では授業中に真面目に取り組まない．他人との比較が難しく，学習の過程や成果を把握しにくい．コンパイル成功が目的となりがちで実行結果の吟味が不十分である．これらの状況を改善するため，3〜4 人のグループを組んで，互いに協力させることも考えられる．しかし，不適切な協力状態に陥り，他人任せになったり，不公平さを感じる恐れがある．そこで，分担による責任の明確化と，モチベーションを維持したまま授業を締めくくる方法が必要である．本研究では，グループコンテストを取り入れた C プログラミング演習を提案し，支援システム tProgrEss を開発する．本論では，コンテストの運営方法と実行テストによるソースコードの評価方法を検討する．

2. プログラミングコンテスト

2.1. 競技系コンテストの教育的意義

近年，高校生や大学生を対象とするプログラミングコンテストが盛んになってきている．大学の応用的な演習授業においても，これらのコンテスト形式を導入するケースが見られる．ここでは，特に，与えられた問題を限られた時間内に解く競技系のコンテストを取り上げる．競技系では；正解数，解答時間，実行結果，実行速度など，評価基準が明確で，具体的な数字で勝敗が決定される．そのため，授業における最終的な実力試験などで利用されることがある．

一方，成績のための総括的評価を目的とせず，学習意欲を高めるための形成的評価として，競技系のコンテストを用いることも考えられる．通常の小テストよりも，学生の心理的な拒否感が少なく，ゲーム感覚で取り組める．また，成績に関係しなくとも，勝つという目的が明確なため，他人との競争意識が真面目に取り組ませることにつながる．

2.2. プログラミングコンテスト ACM-ICPC

競技系コンテストの代表的なものとして，米国計算機学会が主催する大学対抗国際プログラミングコンテスト ACM-ICPC がある[1]．このコンテストでは，インターネット国内予選から，アジアなどの地区大会を経て，世界大会までがある．筆者を含むチームも，日本の国内予選とソウル大会に出場した[2](図 1)．いずれも，3 人 1 組で 1 台の PC を用い，数学パズルのような問題に取り組む．制限時間内にできるだけ多く早くプログラムを作成することを目指す．プログラム言語は，C または Java を選択する．プログラミングエディタと標準のライブラリ以外は，いかなる電子的な補助手段も用いてはいけない．オフラインであれば，参考書や事前に作成したソースコードの印刷物などを参照しても構わない．ソースコードは，大会サーバに送信し，サーバ側でコンパイルと入出力チェックを行う．誤答の場合は，正答に達するまで，再提出を行う．

ACM-ICPC では，基本的な順位は，正解数によって決める．同じ正解数ならば解答時間の合計が短い方が上位となる．誤答の場合は，その回数だけ解答時間にペナルティが加算される．また，各問題の解答時間は，着手から正答までの時間ではなく，コンテスト開始時からの経過時間となる．例えば，問題 A を 20 分，その後問題 B を 30 分で解いた場合，解答時間は，20+(20+30)=20+50=70 となる(図 2)．問題 B を最初から並列に解き始めていれば，解答時間はより短くなる．そのため，チーム内での協調が高得点につながりやすい．

2.3. 初心者のためのコンテスト

香川大学工学部の情報系学科の 1 年次には，「プログラミング 2」を必修科目としている．2 時限連続(180 分)の C プログラミングの演習授業であり，表 1 のようなカリキュラムとなっている．特に，入門編の後半から中級編にかけて，反復構造を中心とした集中的な問題演習が必要となる．

このような状況を踏まえ，本研究では，大学初年時の入門的な C 言語教育において，コンテスト形式のプログラミング演習を取り入れることを考える．総括的評価のための期末のコンテストではなく，毎週の授業を活性化させる形成的評価のための小コンテストを想定する．コンテストでは，即時的で明確な評価により，達成感や満足感を得やすい．勝つという目的が明確であり，競争意欲やゲーム感覚を持った学生間の対戦など，ゲーム感覚の楽しさを持ったイベントとして利用できる．

実際には，コンテストの多くは上級者向けのものであり，初心者への敷居が高く，授業時間内にそのまま組み込むことは難しい．また，コンテストでは結果だけを重視する傾向があるが，教育的配慮からは，過程を評価することも必要となる．これらの問題を解決し，適切なコンテストを運用して，授業全体を盛り上げ，積極的な自己学習に繋げるきっかけとしたい．

表 1　授業計画と学習内容

授業計画		学習内容
入門編	第01回	DOS環境、C言語の概要、言語処理系
	第02回	変数とデータ、演算子と式、入出力、コメント
	第03回	選択構文、論理式、数学関数、定数定義マクロ
	第04回	反復構文、構文要素と書式、基本制御構造
	第05回	多重反復、二分法、局所脱出、基本データ型、文字
初級編	第06回	関数の定義と呼出、基本的な数値算法、時間と乱数
	第07回	配列の宣言と参照、配列の基本操作、データ型名定義
	第08回	多次元配列、行列と整式の算法、ソースの分割
	第09回	単純整列算法、集合演算の算法、ビット処理
	第10回	再帰的処理、変数と宣言、関数形式マクロ
中級編	第11回	文字列、書式付入出力、文字種判定、端末制御
	第12回	文字列とポインタ、文字列処理、文字列検索
	第13回	構造体、共用体、列挙体、データの抽象化
	第14回	配列とポインタ、構造体とポインタ、メモリ管理
	第15回	ファイル入出力、実行時引数、テキスト加工

図 1　ACM-ICPC の参加風景

図 2　ACM-ICPC の解答時間の計測法

3. コンテスト方法

3.1. 本コンテストの概要

本研究では，入門的 C 言語演習の毎回の演習を活性化させるためのコンテストを提案する[3][4][5]．コンテストの実施形態は，以下のようなものである．1 クラスの学生を 3〜4 人のグループに分け，15 チーム程度にする(図 3)．毎回の授業の演習の最後に，総まとめ的な位置付けとして，30 分程度のコンテストを取り入れる．複数の問題を与え，グループ内で分担する．問題文の通りにプログラムを作成し，実行テストにパスすれば得点が与えられる．問題を早く多く解いたチームが上位となる．このような実施形態の上で，グループ編成，テスト方法，評価手法などを検討する．

本コンテストによるプログラミング演習の流れは，教師の事前準備の後，4 つのフェーズに分けて捉える(図 4)．

(0) 教師による事前準備

教師は，授業内容に合わせて，問題データベースからいくつかの問題を選び，出題セットを構成する．各問題には，実行テストのための入力出力サンプルが用意されている．

(1) グループ協調

学生は，チーム単位で，サーバ側に提示され

た出題セットを取得する．メンバーの能力や問題の難易度を元にグループ内で検討を行い，各リーダーが中心となって問題の分担を決定する[6]．グループを組む目的は，学生に共通の目的を持たせることである．この段階は，コンテストとしての時間には含まれず，十分にグループ内で議論してから個人作業へと移る．

(2) 個人作業

1人1台のPC環境で，各自が分担した問題のコーディングとデバッギングに取り組む．この段階では，中間目標を設定し，ローカルな環境でソースコードの作成を行う．

(3) チーム競争

各自がソースコードをサーバに提出し，サーバ側でコンパイルされたバイナリを用いて，実行テストを行う[7]．用意された幾つかのサンプルデータを用い，サーバ側で段階的に予備テストを行う．学生が最終解答と判断したソースコードについては，判定テストに挑戦する．正当性を評価するのに十分な量のサンプルデータを用いて評価を行う．その結果は本人に通知されるだけでなく，Web上に公開される[8][9]．

(4) 全体評価

各問題の得点を総合し，チーム全体の評価を行い，チームの順位が決定する．コンテスト後に，教師がコンテスト結果の分析や講評を行う．

3.2. 実行テストの概要

本研究で扱うコンテストでは，必ずしも全員が問題を完答できるとは限らない．そのため，プログラミングが不得意な学生でも，部分的に解答すれば，チームに貢献できるような評価方法が必要となる．そこで，入出力サンプルによる実行テストを採用する．さらに，実行テストとして，複数の予備テストと最終の判定テストの2種類を用意する．

どちらのテストも早く正答すれば，より多くの点数を得ることができる．ただし，誤答の場合の扱いが異なる(図 5)．予備テストでは，誤答は減点せず，何度でも挑戦できる．エラー情報も提示するため，積極的にテストに取り組み，その結果を元にデバッギングを進める．判定テストでも，正答するまで挑戦することができるが，誤答は減点される．正誤のみしか通知せず，出力データも公開しないため，一発勝負のような側面を持つ．そのため，学生は繰り返し予備テストを行うようになり，十分なテストとデバッグを行う動機付けとなる．最終的には，両方の実行テストを総合して評価を行う．得点の比率は，判定テストの方を大きくし，最後に逆転を狙うことも可能とする．

実行テストは，サーバ側にソースコードを送信してコンパイルし，サーバ内で入出力サンプルによる実行として行う．予備テストでは，入出力サンプルを公開する．そのため，ローカルにテストを繰り返し，出力を想定したコーディングやデバッグを行うことができる．一方，予備テストにも加点することで，サーバ側でのテストを促し，進捗状況を明示的に意識させる．また，進捗状況を教師が把握したり，チーム同士の競争の目安としても用いる．

3.3. コンテストの得点ルール

コンテストの評価手法は，ACM-ICPCの解答時間の換算方法をベースにする．例えば，表2のような得点ルールを考える．問題正答に対する得点は，解答時間に関係なく得られる基本得点と，コンテストの残余時間 T を基本とした得点である．予備テストでは，得られる得点は小さく，残余時間は 1/10 と計算される．そのため，難しい問題でも，予備テストに合格することで部分点を稼ぐことができる．判定テストでは，残余時間がそのまま計算される．ただし，正答までに誤答があった場合は，1 回につき，正答時の得点から 2/10 を減点とする(図 6)．コンテスト終了直前に判定テストに正答した場

合でも，基本得点があるため，得点は0点とはならない．基本得点を調整することで，コンテストの性格付けを行うことができる．例えば，基本得点が大きければ，正答の早さによる得点の比重は小さくなる．そのため，出題された問題を多く解くことが重要になる．逆に，基本得点がほとんど無ければ，早く解けば高得点となるため，簡単な問題を選び，素早く解答することが目的となる．

3.4. 実行テストの教育意図

コンテストを通じ，サーバ側で行う実行テストには，3 つの狙いがある．(1)実行テスト系列でテストを習慣化させる．(2)入力部，処理部，出力部のプログラム構造を意識させる．(3)番兵，例外処理，局所脱出などのコードパターンを理解させる．(1)は，コーディング，コンパイル，テストの繰返しを，実行テストを手がかりに身に付ける．学生は，自分で考えた入力サンプルだけではなく，教師が用意した的確な入出力サンプルを知ることで，適切なテストデータの必要性を学ぶ．(2)と(3)は，予備テスト系列の提示方法で理解を助ける．すなわち，予備テストを中間目標と捉え，自分の進行状況に応じて，適切な予備テストを選択する．この過程で，段階的にコーディングを行っていく．

3.5. 問題情報と出題例

出題する問題は，文法事項，題材，プログラム構造の3つの観点で分類する．本コンテストでは，実行テストによってソースコードを評価するため，プログラム構造による分類を出題意図として重視する．問題情報を構成する要素は，表3の5部からなる．これらの情報は，問題の選択の目安として出題時に公開される．ただし，採点に関わる，検証情報の一部と解答情報については，コンテスト終了まで公開しない．

典型的な問題例としては，「10個以下の整数値の平均値」「年末までの日数計算」「配列を用いた整列処理」などが挙げられる．ここでは，「10 個以下の整数値の平均値」という問題を取り上げる．この問題の出題においては，表4の情報が表示される．また，予備テストとして，表5のような3組の入出力サンプルを提示する．

3.6. 緩和系と部分系の予備テスト系列

予備テストの方法には，図7のような緩和系と部分系の2種類が考えられる．緩和系では，表5中の入出力サンプルのように，パスすべきデータの難易度を段階的に高めていく．例外処理を除々に追加し，完成度を高めていくような問題に適している．学生に段階的なプログラム作成を意識させることができる．部分系では，機能ごとに部分的な動作を別の問題を通して実現させる．例えば，番兵式入力に関するコード，和算と商算に関するコードなど，別々に作成させ，最終的に1つのコードに必要な処理を統合する．これは，処理をモジュール単位に分割しやすい問題に適している．別段階で作成したコードを，最終的に1つに統合し，プログラムが機能の集まりで構成されていることを意識させることができる．一方，判定テストでは，プログラムの正誤を適切に検証できるよう，幅広い入出力サンプルを用意しておく．

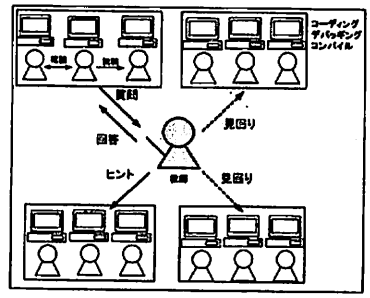

図3　グループ演習の風景

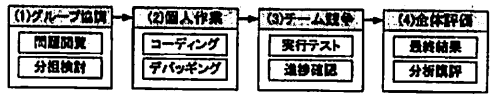

図4　4つの演習フェーズ

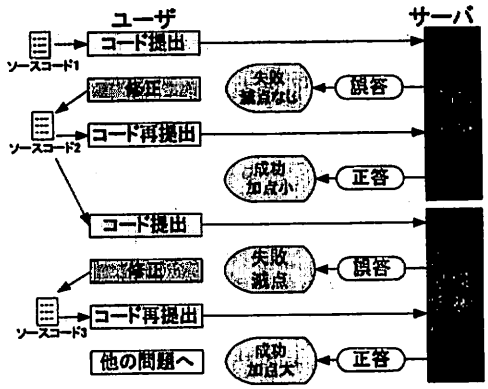

図5　予備テストと判定テストの実施手順

表2　解答時間による得点ルールの例

予備テスト	正答	T×0.1 の加点
	誤答	減点なし
判定テスト	正答	T×1.0 の加点
	誤答	判定テスト正答時に R×0.2 の減点

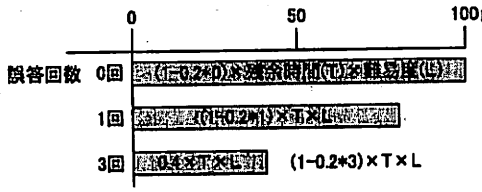

図6　判定テスト誤答後の得点

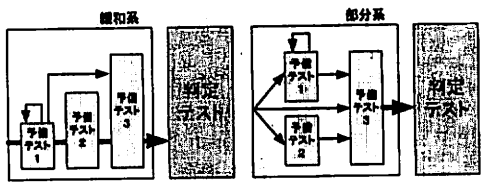

図7　予備テスト系列

表3　5種類の問題情報

情報名	内容
表紙情報	問題名、出題意図、分類、難易度
出題情報	問題文、算法ヒント、予備知識
実装情報	入出力条件、入出力例、実装ヒント
検証情報	入出力サンプル(一部非公開)
解答情報	模範ソースコード、解説

表4　出題情報の例

表紙情報	問題名	10個以下の整数値の平均値
	出題意図	番兵式入力、反復処理、出力整形
	分類区分	文法:反復構文 題材:数学問題 難易度:易
出題情報	予備知識	平均値…各値の和を要素の数で除算した値
	問題文	10個以下の入力に対し、平均値を求めよ。 入力の終了は"-1"で与えられる。
	算法ヒント	入力個数が11以上の場合は考慮しなくてよい
実装情報	入力条件	不定個の正数 整数型
	出力条件	小数点以下2桁までの平均値 実数型
	入出力例	入力:1 2 3 4 5 6 7 8 9 10 -1　出力:5.50
	実装ヒント	出力桁数の調整は printf("%.2f",ans); とせよ

表5　入出力サンプル例と確認項目

予備テスト1. 入力データ数が1個	
基本的な入出力、出力フォーマットの確認	
5 -1	5.00
予備テスト2. 入力データ数が入力制限いっぱい	
定反復による処理への対応	
1 2 3 4 5 6 7 8 9 10 -1	5.50
予備テスト3. 入力データ数が入力制限の半分	
番兵式による入力の打切への対処	
1 4 7 3 2 -1	3.40

4. 大会運営サーバ

4.1. 全体構成

本研究で提案するコンテスト形式の C プログラミング演習において，問題配布や実行テストを実施する大会運営サーバ tProgrEss を開発する．現在，図 8 のシステム構成で，tProgrEss を試作している．大会運営の Web ページは，Ruby 言語による CGI として作成している．コンパイルやテストなどのサーバ側のバックエンド処理も Ruby スクリプトで実装している．

コンテスト中に利用される問題情報，出題セット，採点結果などの情報は，XML 形式で格納している．入出力データ，答案ソースコード，コンパイルと実行結果のリダイレクトなどは，テキストファイルとして保存し，XML データベースから外部リンクとして参照する．また，コンパイルや実行は，安全性の面などから，ファイルを提出する場所とは異なる一時的な場所で行うことになる．学生の解答は，提出記録

DB として保存され，その結果を基にして進捗状況を表示する．問題分担や教師用の Web ページなどは，プロトタイプの段階である．

4.2. 実行テストの処理手順

tProgrEss の中心的な機能である実行テストの処理手順は，図 9 のモジュールで実現する．実行テストの結果は，図 10 の手順により，6 段階で判定する．判定結果を基に，表 6 の 8 項目のうち必要な情報を，結果確認ページとして学生に提示する．

受取検査では，ファイルサイズや拡張子の異常を検査し，不正ファイルを除外する．静的チェックでは，コンパイルを行い，実行バイナリを生成する．静的エラーとなれば，コンパイル結果として，エラー内容を表示する．テスト実行では，実行バイナリに入力サンプルを与えて実行し，出力結果を保存する．一定時間(3 秒程度)を経過して終了しなければ，打切終了とみなし，判定コメントとして，無限ループの可能性を指摘する．また，実行時エラーの場合，異常終了と判定し，エラー情報を表示する．これらの場合でも，途中までの出力データがあれば，出力結果として表示し，デバッグの参考にする．正常終了の場合，出力サンプルを用いた正誤確認を行う．出力結果との照合に成功すれば正答，失敗すれば誤答と判定する．

4.3. tProgrEss の GUI

大会演習の過程において，tProgrEss の主な GUI について述べる．図 11 は，グループ協調フェーズにおける出題セットの取得ページである．このページでは，表 4 のような情報が表示される．出題意図や難易度を基に，グループ内で問題の分担を検討する．図 12 の問題分担ページで，各問題に対する担当者を決定する．図 13 は，チーム競争フェーズにおける進捗状況の確認ページである．予備テストのパス状況，判定テストへのトライ回数，得点や解答時間などが表示される．また，チームごとに，メンバーの分担状況も示す詳細な進捗状況の確認ページも用意する．

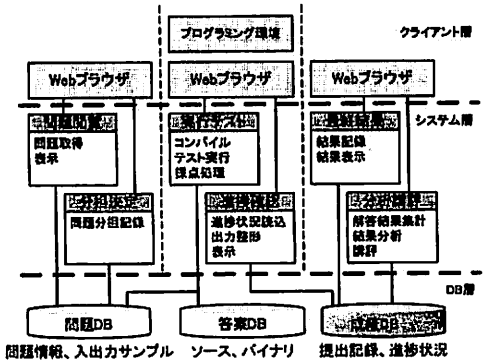

図 8　tProgrEss システム全体構成

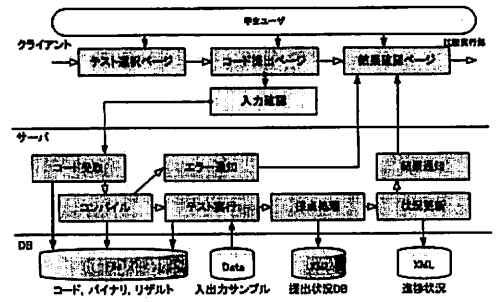

図 9　実行テストの処理モジュール

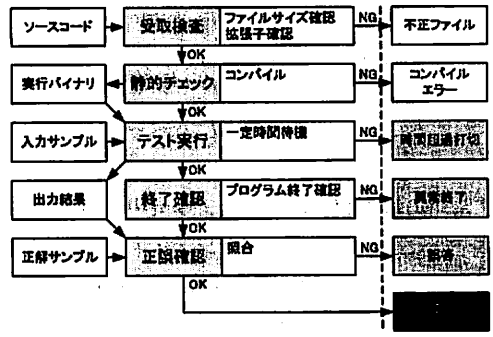

図 10　実行テストによる 6 通りの判定

表 6　8 項目のテスト結果情報

項目名	表示内容
識別情報	提出者、問題名、テスト番号
判定結果	6通りの判定結果
判定コメント	判定結果に対するアドバイス
コンパイル結果	コンパイルエラー、警告
入力サンプル	バイナリ実行に与えた入力
出力サンプル	正解として照合すべき出力
出力結果	実際の実行による出力
エラー情報	異常終了時のエラー情報

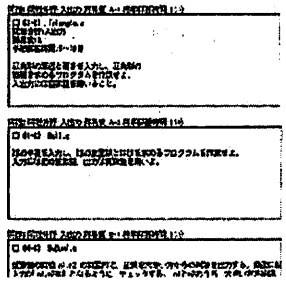

図 11　出題セットの取得ページ

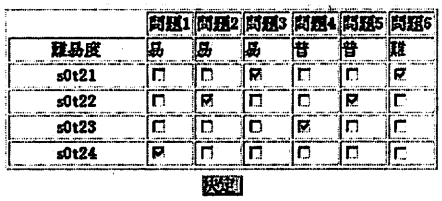

図 12　出題セットの問題分担ページ

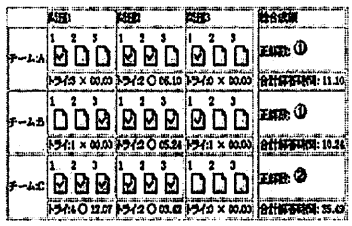

図 13　進捗状況の確認ページ

5. おわりに

C 言語の導入教育において，授業中の演習効果を高めるため，グループコンテスト形式のプログラミング演習とその支援環境 tProgrEss を提案した．出題は，配列操作や統計処理などの簡単な問題を複数与え，グループ内で分担させる．ソースコードの評価は，Web サーバ側へアップロードし，2 種類の実行テストを用いて行う．予備テストでは，複数の入出力サンプルにより段階的に中間目標を与え，途中経過を公開して，競争意識を高める．判定テストでは，最終的なソースコードの正誤を判定し，解答時間や正答までの試行回数などを考慮した得点を与える．問題分担による責任の明確化と，コンテスト形式による競争意欲の刺激により，授業中の演習を活性化させる．また，段階的な目標設定や部分的な評価により，問題解決の達成感を与え，持続的な学習意欲を高める．今後の課題として，実行テスト系列による評価の妥当性と実効性を確認する．また，コンテスト後の講評を提示するようにし，演習効果を高めたい．

謝辞　本研究は，日本学術振興会科学研究費補助金基盤研究(B)(課題番号 17300269)の助成による．

文　献

[1] ACM-ICPC, http://icpc.baylor.edu/icpc/

[2] KAIST, "ACM-ICPC 2005 Asia Regional (Seoul)", http://acm.kaist.ac.kr/forum/

[3] 倉田英和, 富永浩之, "グループコンテストを取り入れた C プログラミング演習における実行テストと評価ルールの検討", ゲーム学会 全国大会 2005, pp. 35-38, (2005).

[4] 倉田英和, 富永浩之, "グループコンテスト形式の C プログラミング演習支援環境 tProgrEss- コンテストの運営方法と実行テストの評価方法の提案", 信学技報, Vol.105, No.632, pp.129-134, (2006)

[5] 倉田英和, 富永浩之, 林敏浩, 山崎敏範, "実行テストを用いたコンテスト形式の入門的 C プログラミング演習", 教育システム情報学会 第 31 回全国大会, pp.533-534, (2006).

[6] 側由美, 神田奈美, 渡辺成良, "CSCL におけるグループ構成と役割配分の手法", 教育システム情報学会 第 29 回全国大会, pp.59-60, (2004).

[7] 田上恒大, 阿部公輝, "比較的大きなプログラミング課題のための指導採点システム", 情報処理学会, CE83 研究報告, pp135-140, (2006).

[8] 村井万寿夫, "学習意欲を高めるための手立てについて", 日本教育工学会研究報告集, JET03-4, pp.31-36, (2003).

[9] 安留誠吾, 内藤広志, "プログラミング演習における進捗状況表示システム", 教育システム情報学会 第 29 回全国大会, pp.171-172, (2004).