プログラミング導入教育を目標にした SIST オンラインジャッジシステムの試作と提案

Development of SIST online judgement system and its Proposals for introductory education in programming

山下　颯也*、國持　良行*

Soya YAMASHITA and Yoshiyuki KUNIMOCHI

Abstract : In order to utilize Online Judgment Systems, which is often used in programming contests, for introductory education in programming, We have been prototyped the system on the local campus network. In this paper, we describe the outline of this system and the verification plan for educational effects.

1. はじめに

プログラミングコンテスト等でよく使われるオンラインジャッジシステム(Online Judgement System，以下 OJS)をプログラミングの導入教育に活用するために，OJS を学内ネットワーク上に試作した. 本論文では，この OJS のシステム概要と教育効果の検証計画について述べる.

内閣府は Society 5.0 を提唱し，ICT 技術を利用して仮想空間と現実空間の融合する，日本が目指すべき未来社会を描いている．また，2020 年度から小学校でプログラミングが必修化され，2021 年度中にはデジタル庁の設置が予定されている．ICT，AI，DX を駆使する高度情報技術者への需要も今後一層高まるであろう.

高度情報技術者に必要なプロググラムの設計と実装の能力を身に着けるためには，まず最初に，プログラミングの学習からスタートする．そこで，お手本となる多くのプログラムを実際に入力して，覚えることが記述能力の向上につながる．学習者には，正規の授業時間内で学ぶだけでなく，自学自習でよい問題を解く必要がある. 本研究では，学習者の能力を詳細に測定したり，問題の良し悪しを評価したり，モチベーションを維持する枠組み提供したりするために独自に OJS を開発し，その運用計画を立案する.

既存の OJS としては，会津大学の AOJ[1]，北京大学の POJ，AtCoder などがよく知られており，ランキング，コンテストやリクルートなどで活用されている．文献[1]は OJS やその構築法をわかりやすく概説している．本研究では比較的ポータブルな独自の設計をした．また，Moodle のプラグインとしての OJS なども提案されいるが，サーバに負荷をかけ過ぎる不安がある.

また，OJS をプログラミング教育に活用する報告もみられる．不正コピー防止や作問の効率向上についての提案もされている[8][9]． 今回は，試作した SIST OJS(以下 SOJ)の概要を紹介し，2021 年度の導入計画を立案する. そして，計画では，以下の点を考察したい.

(1) 受講者のモチベーションの向上

(2) 受講者のプログラミング能力向上(読解，記述)

(3) 指導者(システム管理者)の作業負担

現在，学内ネットワーク上で SOJ を 1 台の PC 上に作成して，各種動作テストを実施している．基本的な機能はほぼ実装されているので，2021 年度前期中に(プログラミング初心者用の)問題一式を作成して格納し，1 年生後期必修「プログラミング演習」での試験運用を準備する．そして，演習の進捗管理，成績管理を可視化して，効果的な学習の実現を目指す．また，学生ごとの理解度に応じた問題を選定する際には，AI の活用も試みたい.

2. OJS の概要と本システムの外部設計

プログラミングの導入教育を目的とした OJS を試作した．この節では，プライベートな環境で運用できる OJS の設計を述べ，運用方法を提案する.

2021 年 2 月 26 日 受理

* 情報学部 コンピュータシステム学科

2.1 OJS とは

OJS は，一般に Web 上のサーバで学習者から提出されたプログラムを自動採点し，その結果をデータベース(以下 DB)へ記録，学習者へ表示するシステムである．

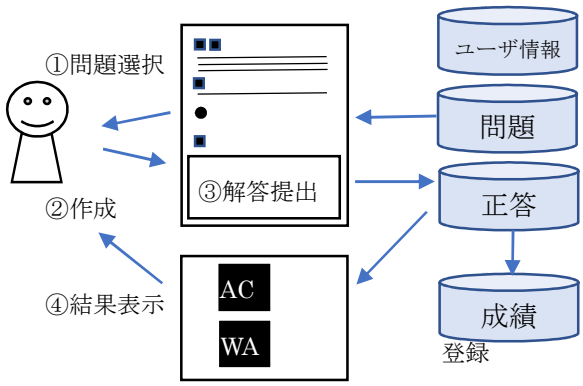


図 2.1 OJS での学習方式

学習者は，① 用意された問題の中から 1 つの問題を選択して，プログラムを作成する．そして，②プログラムを作成したら，③ そのプログラムを提出欄に貼り付けて提出(**submit**)する．プログラム提出後，④システムは自動採点し，結果を返す．最後に，⑤データベースに問題番号，正解不正解の状態，実行時間，最終日時などを登録する．

上記の学習を繰り返し，成績は DB へ記録される，成績記録から，学習者のプログラミング能力，モチベーション，全体成績，問題難易度などを可視化し，学習者の学習意欲を引き出す．

2.2 システムの外部設計

本システムの利用者(ユーザ)を，学習者，教師，管理者の 3 つに分ける．システムには最初ただ一人の管理者 root/admin が存在するものとする．管理者はユーザ登録ができ，教師と学習者をそれぞれ 1 名以上登録していると仮定する．図 2.2 は開発中の SOJ のトップ画面である．以下では，学習者，教師，管理者が実行できる操作を述べる．

① **学習者モード**

プロファイル設定

問題選択，問題解答・提出，採点結果閲覧

正解プログラムの公開の可否設定

個人成績表示

全体成績サマリ表示

図 2.2 SOJ のトップ画面(開発中)

② **教師モード**

学習者モードの全機能

問題作成

サンプル入出力の作成・変更・削除

クラス編成(担任学生の設定)

クラスの成績サマリ表示(DL)

③ **管理者モード**

教師モードの全機能

ユーザ管理(登録/削除/変更)

2.2.1 学習者モード

学習者はシステムにログイン後，問題解答，成績表示が行える．

問題選択

プログラミングコンテストでは制限時間が設けられていることが多いが，本システムでは学習を対象としてそれは設けていない．個人成績一覧から正解していない問題を選ぶのが一般的である．

学習者のレベルに応じた問題や苦手分野の問題などをシステムが AI やデータサイエンスを利用して選択し，効率の良い学習法は今後の課題として残されている．

問題解答・提出

学習者は，問題選択画面において問題を選択する．問題解答・提出画面では，問題番号と問題文，サンプルの入力とそれに対する出力の例がいくつか記述されてい

る(図 2.3)．現在 C/C++言語のみを採点できるが，将来的には他言語が選択できるようにしたい．学習者は解答プログラムを作成し，それを解答提出画面の解答欄へ貼り付け，提出する．

図 2.3 SOJ の問題出題画面(開発中)

図 2.4 SOJ の解答提出画面(開発中)

その後，SOJ は解答を採点して，以下の結果を判定する．

・正解(Accepted，AC または Correct Answer，CA)
・不正解(Wrong Answer，WA)
・コンパイルエラー(Compile Error, CE)
・実行時エラー(Runtime Error, RE)
・制限時間超過(Time Limit Exceed，TLE)
・記憶制限超過(Memory Limit Exceed, MLE)

個人成績表示

学習者のプロフィール，

1) 提出回数，正解回数，得点，順位，レベル(級・段)などの累積の成績を表示する

2) 今回の提出に関して，採点結果(AC，WA，CE，RE，TLE，MLE)を表示する．AC の場合は，実行時間も表示する．

全体成績サマリ成績表示

全学習者の成績サマリを一覧する．サマリ一覧の一行には，順位，ユーザ名，正解問題数，提出問題数，総得点などを表示する．この際，問題ごとに付与される得点(レート)は現在検討中である．得点の付与の仕方としては，問題作成者が得点を(静的に)設定する，その問題の正解者数等から得点を(動的に)算出する，などが考えられる．難しい問題ほど高得点になるようにする．

2.2.2 教師モード

1 人の教師は，複数の学習者を担任し，クラスを構成する(担当学生の設定)．また，問題作成が一番負荷のかかる作業となる．

クラス編成

一人の教師が，担当する学生を選択する機能である．選ばれた学生がクラスを編成する．教師は担当する学生の学習状況を常に把握しなくてはならないので，それを可視化できる方がよい．また，成績を付与する場合には，自動的に総得点などを集計できることが望ましい．

クラスの成績サマリのダウンロード

クラスの成績サマリを，成績処理などのために加工・集計したい場合がある．CSV ファイルなどの形式でこれらをエクスポートする予定である．

問題作成

OJS の採点は，提出されたプログラムに対して，テスト入力を読み込ませ，プログラムからの出力を記憶する．その出力を，あらかじめ用意されたサンプル出力と比較する．すべてのテスト入力に対する出力がサンプル出力と一致し，かつ実行時間が制限時間内である場合に，正答 AC を与えるものとする．

OJS の採点法の特徴により，問題作成の要件としては，入力を読み込んで，結果を出力する仕様にする必要がある．以下に，問題例をあげる．

問題 正整数nと n個の整数a_i $(1 \leqq i \leqq n)$ を読み込み，1 回だけしか現れない整数が何個あるかを出力しなさい．

制約条件：$1 \leqq n \leqq 10^5$

$1 \leqq a_i \leqq 10^5 \ (1 \leqq i \leqq n)$

実行制限時間： 1000ms

例 1. 3 個の整数のうち 1 回だけ出現するのは 9 だけであるので，正解は 1 である．
入力
3
7 9 7
出力
1

例 2. 5 個の整数のうち，8 は 2 回，7 は 3 回出現するので，答えは 0 個となる．
入力
5
8 7 7 7 8
出力
0

問題文に続き，入力データの制約条件や実行制限時間が記述される．これらは，データ構造を決定するのに必須の情報である．次に，入出力例があり，学習者はこれにより問題への理解を深める．

サンプル入出力の作成・変更・削除

提出されたプログラムが正しいかどうかを判定することは計算理論上では不可能な問題である．教師が作成する正答プログラムやサンプル入出力は，正しいと仮定している．また，OJS では与えられたサンプル入力のすべてに，制限時間内に出力した内容が，サンプル出力と一致しれば正解(AC)であると仮定する．すなわち一種のブラックボックステストである．このように仮定を設けても実用上プログラムの正当性を十分に保証できる．

なお，サンプル入力は，(1)特殊なパターン(境界値，1 元，冗長，自明)，(2)制約条件の上限，(3) 一般のテストデータなどを網羅する．(2)や(3)のケースについては，乱数などを使ってプログラムで自動生成する．

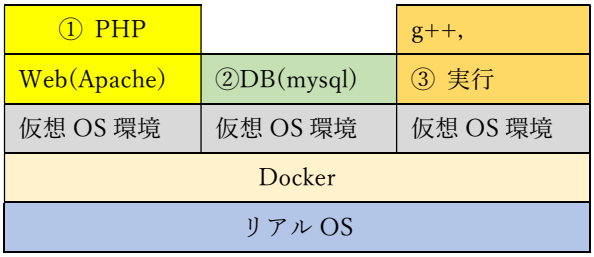
3. システム概要

ここでは SOJ のシステムの開発概要を述べる．

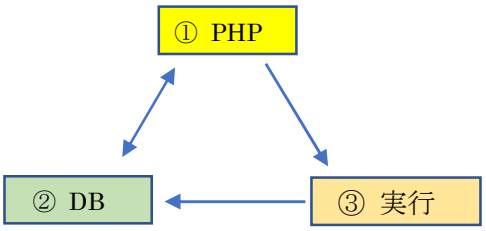
3.1 システム概要

本 SOJ は，学内ネットワークのみからアクセス可能な 1 台サーバー上に実装している．本システムは，主に ①インターフェースを担当する **PHP コンテナ**，② ユーザ情報，管理情報，問題情報を管理する **DB コンテナ**，③ プログラムを起動し，正誤を判定する**実行コンテナ**からなる．①～③は **Docker** の仮想環境で動作するコンテナとして実現されている．したがって，ハードウェアから切り離され，Docker がインストールされていればどのようなリアル OS でも運用が可能である．

仮想 OS 環境＝コンテナを Web，DB、実行ごとに別々の OS 上で動作させること．



(a) システム動作環境



(b) システムの処理の流れ

図 3.1 システム概要図

① PHP コンテナは，Web サーバ上で動作する．管理者やユーザ(教師と学習者)との窓口となる．ユーザからは，プロファイル設定(登録・削除・変更・プロファイル確認)や操作要求(ログイン・ログアウト・問題選択・採点・成績表示・履歴おて表示)などの要求を処理する．管理者からは，ユーザ情報管理，システム情報管理(DB 管理)，ジャッジ運用(問題と正答データの作成，コンパイラ選択，コンパイラ実行，結果表示など)等の要求を処理する．

② DB コンテナは，ユーザ情報，管理情報，問題情報などを管理する．ユーザ情報はプロファイル，提出履歴，成績などが含まれる．管理情報には，クラス編成，コンテストの情報などがある．問題情報には，問題集とその各問題に対する正答データが含まれる．問題は分野と難易度がつけれるが，難易度は解答正誤履歴を加

味することが望ましい．問題は，学習者(ユーザ)に理解できる言語(日本語や英語)で書かれ，解答されるプログラム言語に依存しないことが望ましい．

③ 実行コンテナは，学習者(ユーザ)が解答(プログラム)提出時に選択した言語処理系へ渡して，実行結果と正答データと比較して，正誤を判定する．現在選択できる言語処理系は C/C++コンパイラのみであるが，順次言語処理系を増やしていく予定である．

正答データは，テスト入力(ファイル)とそれに対する正しい出力が格納されたファイルの複数の集まりである．すべてのテスト入力に対して，提出プログラムが正しい結果を出力した場合に，正解と判定する．テスト入力に対して制限時間内に実行が終わらなければ，その実行を Docker コンテナ(内の OS)が強制的に打ち切り，「制限時間オーバー」(TLE)を判定する．また，実行中にエラー(例外など)が発生したときにも，強制的にプロセスを停止し，「実行時エラー」(RE)と判定する．これらの結果は DB の提出履歴と成績に反映される．

悪意のあるプログラムが入力される恐れがあるので実行コンテナの実行権限，ファイルアクセス権限は最低に設定する．

セキュリティ機能を搭載後に，インターネット上に公開し，将来的にはグローバル環境の負荷や信頼性に耐えられるように，複数台の分散サーバーに移行したい．

3.2 システムの実行環境

SOJ のシステム構成は，2021/2/25 現在以下の通りである．

【ハードウェア・OS】

OS: Ubuntsu 18.04.3 LTS (BIONIC BEAVER),

(プライベート IPv4 アドレス)

【ソフトウェア】

Docker(プロセス管理): 19.03.13

Apache(Web サーバ): 7.2

PHP(インターフェース): 7.2.34

MySQL(データベース) : ,8.0.21 - MySQL Community Server - GPL

GCC(コンパイラ): 7.5.0 以上(予定)

3.2.1 Docker とは

Docker とは，コンテナ仮想化によってアプリケーションを開発・実行するためのオープンソースソフトウェアである．この機能を用いることによって，簡単かつ高速にコンテナを立ち上げることができ，コンテナを分散することによって，意図せぬ動作による不具合の影響も軽微に抑えることが可能になる．

3.2.2 Docker-compose とは

Docker-compose とは，1 つのアプリケーションを複数のコンテナとして組み合わせて定義するファイルである．これを用いて，PHP コンテナ，DB コンテナ，実行コンテナを合わせて 1 つのアプリケーションとして定義することができる．

3.2.3 提出プログラムの採点待ちと割り付け

PHP コンテナに提出されたプログラムは，まず実行コンテナに送られる．実行コンテナに送られた後は，拡張コマンド task-spooler などを用いてキューに登録し，実行用のプロセスが空き次第実行に移す．採点の実行後は、採点の結果をデータベースに格納しプロセスを終了する．

現在は実行用のコンテナが 1 つであり直列での処理になっているが，ゆくゆくは 3～5 個程度の実行コンテナを用いて並列処理を行う予定である．コンテナの状況を見て割り振りを行う必要があり，課題である．

3.3 データベースの構成

データベースのテーブルのうち主なものは，ユーザ管理用テーブル，問題管理用テーブル，提出結果管理用テーブルの 3 つである． 提出結果においては，大量の提出によるデータベースの容量不足が危惧されるため，User_id と Problem_id の組み合わせ 1 つにおいて，正答かつ最新の提出を保持するものとひとまずは定める．

表 3.1 ユーザ管理用テーブル

列名	データ型	説明
Id	**Int**	[主キー]ユーザの id
Name	**Varchar(64)**	ユーザ名
Auth	**Int**	ユーザの持つ権限
Pass	**Varchar(256)**	パスワード(ハッシュ化)

[Auth について] DB のアクセス権限は以下の順である．

0 -> 管理者， 1 -> 教師， 2 -> 学習者(学生)

表 3.2 問題管理用テーブル

列名	データ型	説明
Id	**Int**	[主キー]問題の id
User	**Int**	問題作成ユーザの id
Title	**Varchar(64)**	問題名
Time	**Int**	実行時間制限(秒)
Memory	**Int**	実行メモリ制限(MB)

表 3.3 提出結果管理用テーブル

列名	データ型	説明
Id	**Int**	[主キー]提出結果の id
User_id	**Int**	提出ユーザの id
Problem_id	**Int**	提出問題の id
Lang	**Int**	言語
Status	**Int**	実行結果
Time	**Int**	実行時間
Memory	**Int**	使用メモリ
Timestamp	**Date**	提出日時

4. 運用に向けての計画

現在，SOJ を試作し，基本的な動作を確認している．今後の運用に向けての計画を述べる．また，運用中の課題をシステムへフィードバックする．

4.1 動作確認(妥当性，負荷テスト)

現状では，①③の動作確認が終わっており，②の一部確認が終わっている．

① ユーザ登録，削除，変更など (各種の操作は正しいか)

② 妥当性のテスト : テスト問題，正答データ，提出状況，正誤判定，制限時間干出，エラー検出，DB の動作は正しいか？ 負荷情報(Docker)

③ DB のテスト :

④ 負荷テスト(ユーザ n 名)の場合

4.2 実験(学習効果)の計画

2021 年度からカリキュラム改定により，情報学部のプログラミング導入科目である 1 年後期Ⅱ類必修「プログラミング入門」は，1 年後期Ⅲ類必修「プログラミング概論」と「プログラミング演習」に移行する．つまり，半期コマ数は 15 から 30 へ倍増する．このため，「配列」と「関数の基本」が新たに学習範囲へ加わった．

情報学部 1 年生必修科目「プログラミング演習」の國持担当クラスでは，シラバスに書かれた範囲の問題を十分な数(150 題程度)用意する．本システムの学習者として，履修者を登録する．1 回目の授業(ガイダンス)のときに，システムへアカウントを周知し，使用法を説明する．その際に，各回の授業ではこの問題範囲を自習するように指示を出す．

初回にはアンケートをとり，プログラミングの経験の有無などを調査する．8 回目と 15 回目には習熟テストとアンケートを実施することにする．習熟テストでは，可能であれば知識，理解，記述，応用・発展の各能力を測れる作問をしたい．

【試行 1】モチベーションの観察

2 回目以降毎回，提出問題数，正答問題数の記録をとり，OJS へのモチベーションを観察する．学生には，直接に OJS のデータが成績と関係するとは伝えない．この指標は，自学自習に OJS が役立っているかを測るものである．

【試行 2】習熟度へ OJS の寄与

提出問題数，正答問題数，総得点，レベルなど指標と習熟度の関係を調べる．また，1 回目の習熟度と 2 回目の習熟度の差をとる．総得点，レベルなど指標と 2 回の習熟度テストの差の関係を調べる．この指標は，各レベルの学生についても OJS が役立っているかを測るものである．

【試行 3】成績への寄与

提出問題数，正答問題数，総得点，レベルなど指標と実際の成績標語を比較する．正の相関があるものと予測する．この指標は，プログラミング作成能力に OJS が役立っているかを測るものである．

できる学生は，本システムでもよい結果を残すことは容易に想像がつくので，正の相関になると思われる．他方，試行 2 と試行 3 で，負の相関や無相関であれば，本システムはプログラミング能力の向上に役に立たないか逆効果ということになる．

4.3 作問方針と習熟度テスト

OJS の作問方針について述べる．「プログラミング演習」の 1 コマ当たり 10 題，半期で 150 題の問題が第 1 目標として想定している．1 つの問題を作成するには，(プログラミングとオンラインジャッジの経験をもっていると仮定して) 平均して 1 時間程度はかかるであろう．工数としては 150 時間人を見積もる．

まず，スタッフの作問手続きの方針を策定し，スタッフに理解してもらう必要がある．

OJS 作問方針

1 つの問題は，以下のものから構成される．

(1) 問題ファイル(HTML),

(2) 模範解答プログラム

(3) 複数個のサンプル入出力

2.2 節で述べたように教師が，問題およびその模範解答プログラムを作成する．さらにサンプル入力(複数のファイル)を用意する．サンプル入力を模範解答プログラムに読み込せて，サンプル出力(複数のファイル)を生成する．また，サンプル出力を自動生成するためにスクリプトを作成する(図 4.1).

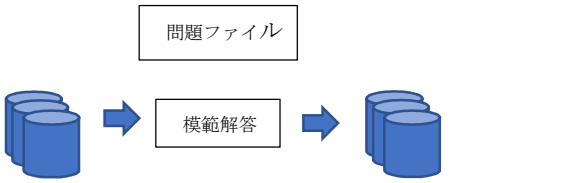


図 4.1 問題作成過程

付録 A に「プログラミング演習」の学習項目を示し，それを理解するのに適した問題例を示す．模範解答を記述し，サンプル入出力を作成する，という手順で作業を行う．

教師の作業をサポートする，作問スタッフを養成することは必須である．サンプル入力を作成する場合にはできるだけ自動化を図り，作業効率を上げるのが得策と思われる．

サポート計画

学習者向けの学習支援については，動画教材をパワーポイントで作成し，予復習に活用する．SOJ に掲載された問題を PDF として配布する．

5. まとめ

プログラミングの導入教育に活用するための SOJ を，ローカル(学内)ネットワーク上に試作した．そして，この SOJ のシステム概要と教育効果の検証計画を提案した．

情報学部 1 年後期「プログラミング演習」での運用を目指している．講義開始前までの課題としては，

(1) 外部のホスト(専有物理マシンをホスティングする，固定 IP)上にシステムを移行する．そのためのセキュリティ対策を強化する必要がある．

(2) 付録 A で示す問題を作成する．作問スタッフを確保して，問題，正答プログラム，サンプル入出力を数百題用意する．

(3) 習熟度テストとアンケートを制作する．

また，技術的な課題としては，DB 内のテーブルの型，PHP と実行コンテナの相互処理，正答データの置き場(専用コンテナ)などを解決する．

将来的には，ユーザの学習状況に応じた最適な問題を選択するための AI 機能を機械学習を使って実現すること，「プログラミング実践演習 2」「実用プログラミング 1, 2」レベルの問題を作成すること，他言語(Python, Java など)のプログラムでも採点できること，負荷が増した場合に並列処理で負荷分散を図ること，などが挙げられる．

謝辞

本論文を精読して有益なご助言をいただいた査読者に深謝する．

参考文献

[1] 渡部有隆．″オンラインジャッジの開発と運用−Aizu Online Judge.″ 情報処理 56.10 (2015): 998−1005.

[2] 松永賢次．″導入プログラミング教育におけるオンラインジャッジシステムの活用の試み.″ 情報科学研究 31 (2011): 25−41.

[3] 古谷勇樹，林 真史，山本 隆弘，長尾 和彦，″RK−003 オンラインジャッジシステムと連携可能な Moodle プラグインの実装と比較 (K 分野: 教育工学・福祉工学・マルチメディア応用，査読付き論文).″ 情報科学技術フォーラム講演論文集 14.3 (2015): 89−94.

[4] 長尾和彦，古谷勇樹，峯脇さやか，”オンラインジャッジシステムのプログラミング演習への導入と評価.” 第 78 回全国大会講演論文集，1，pp.537 - 538 (2016)

[5] 古谷勇樹，林真史，山本隆弘，長尾和彦，”オンラインジャッジシステムを用いたプログラミング学習環境の構築

と比較.” 教育システム情報学会 2014 年度学生研究発表会
[6] 松本彩花, 松原南美, 渡邉遥輔, 多田拓, 倉光君郎, ″Sumomo: ブロックチェーンを用いた教育用オンラインジャッジの提案.″ 情報教育シンポジウム論文集 2019 (2019): 321-325.
[7] 岩本舞, 中村真人, 小島俊輔. ″不正コピー検出手法を備えたオンラインジャッジシステムの開発.″ 情報処理学会論文誌教育とコンピュータ (TCE) 1.4 (2015): 38-47.
[8] 関根遼, 伊藤恵, 奥野拓. ″数学文章題を利用したオンラインジャッジシステム向け問題自動生成手法の提案.″ (2020).

付録 A.「プログラミング演習」の各回の学習項目と出題例.

以下に「プログラミング演習」の授業回毎の問題例を示す．その他に SPI 試験，小中学校の数学問題集にはプログラミングに活用できる良問が多く，これらを参考にして問題を作成する．

第 1 回 ガイダンス

学習項目：SOJ の説明，アカウント周知

第 2 回 プログラムの基本構造(4)

学習項目：変数，入出力(scanf，printf)

とくに書式文字列，変換指定

出題例：数字/文字の出力，エコーバック，8/10/16 進数の変換

第 3 回 四則演算と演算規則

学習項目：四則演算，代入演算

出題例：図形(長さ,面積)，道付き長方形，整数四則演算(時給×時間, 単価×数量, 速さ×時間等), n 変数多項式の値，整除，下位 n 桁の数字，特殊文字の出力，含除原理

第 4 回 変数と型

学習項目：データ型，初期化

出題例：四則演算(割引，割増，公共料金，消費税，摂氏華氏，温度熱量，BMI，為替計算)，丸め，一次方程式の解，小数四則，精度(0.1 を 20 桁表示)，本を読み終える日数

第 5 回 代入，型変換，その他の演算子

学習項目：代入，型変換

出題例：大小判定，奇偶判定，消費税，税込みになれない価格，四捨五入，値の交換

第 6 回 if 文

学習項目：**if** 文, **if else** 文，等価演算子，関係演算子

出題例：等価，大小，奇偶，約数，絶対値，床関数，天井関数，3 変数の最大最小，ツェラーの公式(曜日)，三角/四角形の判定と分類，直線/円上の n 点の最短訪問時間,

第 7 回 複雑な条件の設定

学習項目：論理演算(**&&**，**||**，**!**)

出題例：うるう年，数の範囲，n 変数の等価，n 変数の大小順,

第8回 switch 文

学習項目：**switch** 文, **break** 文

出題例：月日数の計算，春夏秋冬の判定，曜日表示,

第 9 回 for 文の基本

学習項目：**for** 文, **continue** 文

出題例：総和，累乗，累積和，奇数総和，統計量，素数判定，時分秒年月日変換，数値積分，虫食い算，積み残し，多項式の整数零点，条件を満たす数の列挙,

第 10 回 while 文の基本

学習項目：while 文

出題例：Euclid の互除法, 近似(Taylor 展開), Newton 法，桁数(整数対数)，各桁の数字(和)，複利計算，テキスト処理，繰返し二乗法，分数計算，約数の個数

第 11 回 複雑な繰り返し演算

学習項目：多重ループ

出題例：描画(直線，三角形)，九九の表，素数判定，虫食い算，条件を満たす点の列挙

第 12 回 配列

学習項目：配列

出題例：配列操作(等価，交換，複写，移動，回文)，総和，線形/二部探索，動的計画法，整列，数列中の数の出現，計算機幾何学，線形代数(積，消去法，行列式)，鉄道料金，双六，ボウリングスコア，描画(ヒストグラム)，グラフ(Union-Find 木，最短経路)，貸借清算，×文字列操作関数

第 13 回 関数の基本

学習項目：関数定義，関数呼出し

出題例：統計量計算，再帰呼び出し(階乗，二項係数，フィボナッチ数列)，ハッシュ関数，電卓(数式処理)，約束記号

第 14 回 include と基本ヘッダファイル

学習項目：

出題例：数学関数，二次方程式の解,

第 15 回 総合演習

学習項目：

出題例：多角形の求積，丸め誤差(0.1 の加算)