CHaserWeb：ブラウザ上で動作する対戦型プログラミング学習環境の提案と評価

井上 慎之介[1]　湯村 翼[1,a)]

概要：プログラミング学習には様々な環境が用いられており，そのひとつに U-16 プログラミングコンテストで用いられる対戦型プログラミングプラットフォーム CHaser がある．従来の CHaser は，実行環境のインストールや CHaser サーバーの起動など煩雑な初期設定が必要であり，これが学習者の意欲を削ぐ要因となっていた．また，環境構築における OS や端末への依存性が，導入の障壁となっていた．そこで本研究では，Web ブラウザ上で CHaser のプログラム記述から実行，デバッグ，盤面表示をワンストップで行える CHaserWeb を提案する．CHaserWeb は，Web ブラウザ向け Python ランタイム Pyodide を用い，Python で記述したプログラムを実行可能とした．高校生と大学生を対象に評価実験を実施し，簡便性と使い勝手が高く評価された．CHaserWeb が学習者の開発負荷軽減および学習モチベーション向上に寄与し，プログラミング初学者が挫折しにくい学習基盤として有効であることを示唆する結果を得た．

1. はじめに

CHaser は U-16 プログラミングコンテスト [1](以降, U-16 プロコン) の競技部門で用いられる対戦型プラットフォームである．図 1 の碁盤状のフィールド上で参加者が作成したプログラム同士を 1 対 1 で対戦させることで，戦略的思考力とプログラミング能力を養う教育的環境を提供している．U-16 プロコンは旭川市発祥で，ものづくり教育やプログラミング教育の普及を目指し，2018 年以降は札幌でも開催されている．CHaser を通じた学習は，参加者が自らプログラムを書き，対戦を通じて戦略改善を試行錯誤することが求められ，プログラミング初学者の学習意欲向上に一定の効果が期待できる．

しかし，従来の CHaser 開発では，プログラム実行環境のインストール，エディタやサーバーの準備，IP アドレスやポート番号の指定など，多段階のセットアップを必要とする．こうした初期設定の煩雑さは，初心者にとって学習意欲を削ぐ大きな要因となっていた．また，マニュアル等の設定のための情報の多くが Windows を対象としているため，GIGA スクール構想端末として配布される Chromebook や iPad などでは開発環境整備が困難という問題もある．結果として，CHaser に挑戦しても環境構築段階で挫折し，実際の対戦ロジック開発に十分時間を割けないケースがある．近年，オンライン開発環境の普及により，Paiza.io[2] や Replit[3]，Google Colab[4]，p5.js[5] どを活用することで，ブラウザ上でプログラムの編集や実行が行える．これらは環境構築の手間を省き，場所や端末を問わずにプログラミング学習を行える利点を持つ．しかし，CHaser に関して，従来型のローカル環境構築を必要としていたため，同様の利便性を享受できていなかった．そこで本研究では，ブラウザ内に Python 実行環境を構築し，対戦ロジックやマップ更新を仮想的に管理することで，環境構築不要の CHaserWeb を提案する．

[1] 北海道情報大学
[a)] yumu@yumulab.org

2. CHaser

CHaser は，キャラクターが交互に行動するターン制で進められる．キャラクターは，ユーザーが事前に書いたプログラムに従って行動する．フィールドには，アイテムとブロックが設置される（図 1）．決められたターン内に，多くのアイテムを集めた方が勝利となる．また，ターンが終了する前に，相手キャラクターがブロックと重なってしまうなど，一定の勝利条件を満たした場合にも試合終了となる．

キャラクターの行動は，種類と方向を組み合わせた命令にて指示される．命令の種類は以下の 4 つである（図 2）．

- walk：移動する
- search：一方向の 9 マスの情報を取得する
- look：隣接する正方形状の 9 マスの情報を取得する
- put：ブロックを設置する

また，方向は「up」「down」「left」「right」の 4 通りである．これらを組み合わせた「walk_up」や「look_left」など

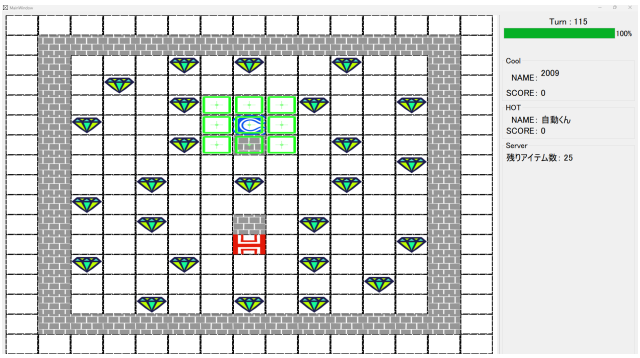

図 1　従来の CHaser 対戦画面

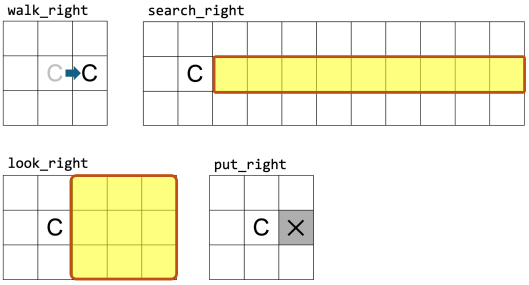

図 2　CHaser の 4 種類の命令．search_right と look_right の黄色い領域は，情報を取得するマスを示す．

の全 16 通りの命令がある．なお，これに加え，ターン毎に必ず実行する get_ready という命令もある．get_ready は，次のターンまで待機を行い，待機終了時に周囲の 8 マスの情報を取得できる．

3.　CHaserWeb

CHaserWeb は，ブラウザアクセスのみで CHaser のクライアントプログラムの作成，実行，盤面の可視化を行える Web アプリケーションである．CHaser では様々なプログラミング言語でクライアントプログラムを記述できるが，CHaserWeb では Python を対象とした．図 3 に CHaserWeb の画面を示す．従来は Windows 環境を前提にした Python インストールや CHaser サーバーの起動が必須だったが，CHaserWeb ではそれらの手順を一切不要とした．これにより，Chromebook や iPad などの端末を含め，OS を問わず多様なユーザーが統一的な学習体験を得られる点が大きな特長となっている．具体的には，コードエディタ，即時実行とデバッグ機能，盤面表示の 3 つの主要機能を中心に構成している．コードエディタは Web 上にコードエディタを実装するためのライブラリである CodeMirror[6] を活用した．

CodeMirror をベースとしたエディタでは，Python コードのシンタックスハイライトや行番号表示，全角スペース可視化など，初心者がつまずきやすい要素を補助する仕組みを備える．黒背景の UI とモノスペースフォント，インデントガイドの併用によって可読性を高める工夫も行った．ファイルのアップロードとダウンロードにも対応して

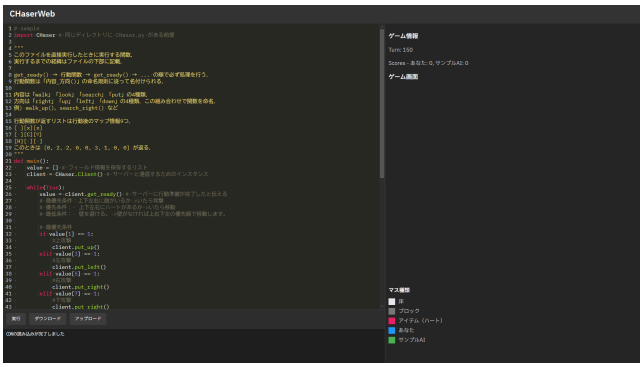

図 3　CHaserWeb の画面

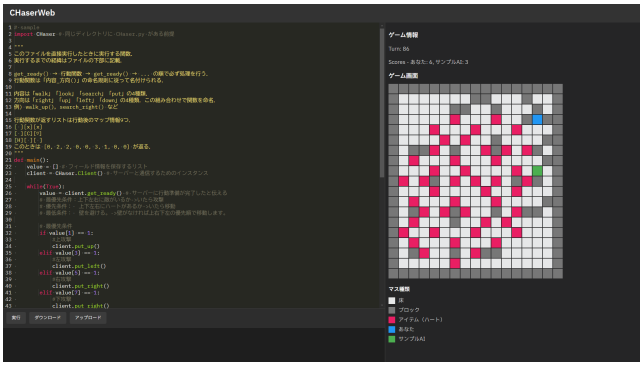

図 4　CHaserWeb で実行ボタン押下後

おり，ユーザーはローカル上のコードを気軽に読み込みつつ，編集したファイルを保存して持ち帰ることができる．

次に，即時実行とデバッグ機能では「実行」ボタンを押すだけで，ブラウザ内の Python インタプリタ (Pyodide[7]) がコードを解釈し，同一画面で実行結果をコンソール表示する．print などの出力はリアルタイムに反映され，エラーメッセージが発生した場合にはどの行でエラーが起きたかが即座に示されるため，初心者の段階的な試行錯誤をサポートする．さらに対戦ロジックでは，サーバーとクライアント間の通信をブラウザ内部で仮想化する形を取り，Pyodide 上でサーバー機能を模して進行管理が行われる．ユーザーはクライアントプログラムを書くのみでゲームが成立し，従来必要とされたポート番号やサーバーアドレス入力などの設定が不要になった．

最後に，盤面表示機能では，図 4 のように「実行」ボタンを押したあと，HTML 要素をグリッド状に並べてターンごとの盤面状態をビジュアルに描画する．プレイヤーやブロック，アイテムなどを色分け表示し，ターン経過やスコアの変化がすぐに把握できる仕様とした．上部にターン数や得点を表示することで対戦の進行状況が一目で分かる設計とした．

4.　設計と実装

4.1　全体アーキテクチャ

CHaserWeb は「フロントエンド」と「サーバー

ロジック」で構成される．フロントエンド側では HTML/CSS/JavaScript をベースにした UI やエディタ画面を提供し，ユーザーの操作を受け付ける．一方，サーバーロジックは WebAssembly 上で動作する Pyodide を介して実行され，CHaser におけるゲーム進行管理を担当する．

4.2 フロントエンド

フロントエンドは主に以下の機能を担う．

- **エディタ管理**: CodeMirror を用いてブラウザ上に Python コード編集環境を実装．シンタックスハイライトや行番号表示，全角スペースの可視化，インデントガイドなどを提供し，初心者でも見やすいエディタ体験を目指す．
- **GUI 操作・DOM 操作**: ユーザーが編集したソースコードを「実行」ボタン押下時に JavaScript の API 経由で Pyodide へ送る．また，ゲーム盤面を更新する際はサーバーロジックからの通知を受け取り，DOM 要素を格子状に再配置し，プレイヤーやブロック，アイテムの描画を行う．
- **ファイル入出力**: 複数ファイルを扱うことを想定し，ファイルアップロード用の input type=”file”要素を非表示で持ち，選択されたファイル内容を CodeMirror エディタへ読み込む．ダウンロード時には Blob オブジェクトを生成し，URL を経由してユーザーのローカル環境へ保存させる．

4.3 サーバーロジック

Python で記述された CHaser クライアントプログラムの実行は Pyodide が担う．Pyodide は WebAssembly 上に Python インタープリタを構築する仕組みであり，ブラウザだけで従来の Python スクリプトを実行可能にする．CHaserWeb では，従来ローカルで動作させていた CHaser サーバープログラムを Pyodide 内に移植し，ブラウザ内でクライアントプログラムとサーバーロジックの仮想的なやり取りを行う形をとる．

具体的には，ユーザーが「walk」「put」「look」「search」などのメソッドを呼び出すと，サーバーロジックがこれを受理し，マップ上の状態更新やスコア計算，勝敗判定を進める．この結果はフロントエンド側の関数に通知され，JavaScript が DOM を再描画して盤面を更新する．ゲーム終了条件を満たした場合はコンソールへメッセージを出力し，ユーザーは画面とコンソール出力の両面から結果を把握できる．

4.4 モジュール構成

本システムでは，内部実装を以下の 4 つのモジュールに分割して管理している（図 5）．

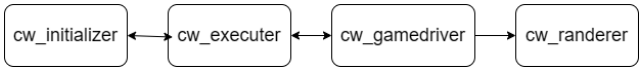

図 5 CHaserWeb のモジュール構成図

- **cw_initializer**: AST（抽象構文木）を用いてユーザーコードを解析し，while True: による無限ループを検知し変換する．具体的には while True: 内の処理を do_turn() 関数として切り出し，ブラウザから 1 ターンごとに呼び出す構造へ書き換える．これにより，ユーザーが意図せず無限ループを発生させた場合でもフロントエンド（JavaScript）側から制御を回収しやすくし，ブラウザフリーズを防止する．
- **cw_executer**: 「実行」ボタンの操作受付やターン進行を管理する司令塔モジュール．まず cw_initializer にユーザーコードを渡し，変換後のコードを Pyodide 上で exec() する．同時に cw_gamedriver を初期化してゲーム開始し，JavaScript のタイマー機能を利用して 1 ターンごとに do_turn() を呼び出す．ターン終了後，勝敗がついていればコンソールに終了メッセージを表示し，ユーザーに結果を通知する．
- **cw_gamedriver**: CHaser の対戦ロジックを内包する．マップデータや各プレイヤーの位置，アイテムの配置，スコアなどを保持し，ユーザーのクライアントプログラムからのコマンドを受理してマップを更新する．また，相手プレイヤー AI の行動アルゴリズムも本モジュール内で定義され，ターンごとに両者の行動をシミュレートしてゲーム状態を変化させる．
- **cw_renderer**: cw_gamedriver が更新した盤面情報を受け取り，HTML 要素のクラス付け替え等によって視覚的に描画する．JavaScript を用い，グリッド状にマス目を生成してプレイヤーやブロック，アイテムを色分け表示する．ターン数やスコアの表示もこのモジュールが担う．

4.5 シーケンス図

図 6 に，CHaserWeb の実行時シーケンス図を示す．ユーザーが Web 画面上のエディタにコードを入力し，「実行」ボタンを押下する流れは以下の通りである．

(1) ユーザーがフロントエンド上の CodeMirror エディタに Python コードを入力し，「実行」ボタンを押す．
(2) cw_executer がコードを cw_initializer に渡し，while True: の AST 解析とコード変換を行う．
(3) 変換後のコードを cw_executer が Pyodide 上で exec() し，cw_gamedrive を初期化．
(4) 1 ターンごとに cw_executer が cw_gamedriver へ「1 ターン実行要求」を送り，行動結果を受け取る．
(5) cw_gamedriver はターン処理ののち cw_renderer を呼び出して描画更新を行い，ユーザーへ現在の盤面が可

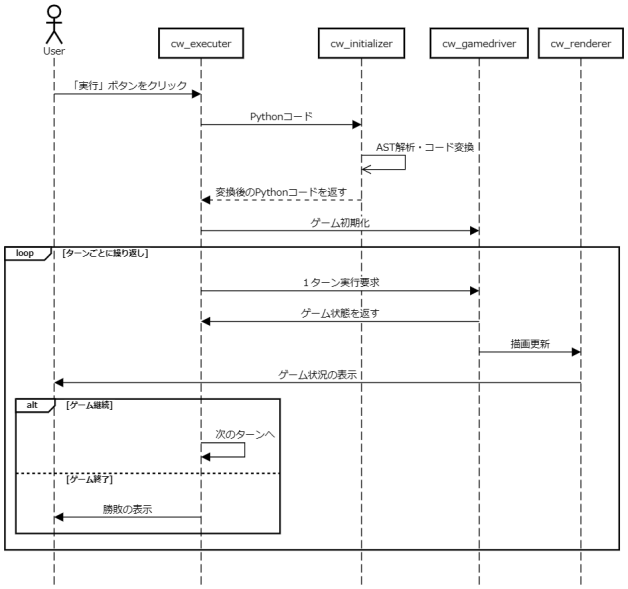

図 6　CHaserWeb のシーケンス図

視化される．

(6) 勝敗がつかなければ次ターンへ，ゲーム終了ならコンソール出力などを通じて結果を通知し，シーケンス終了．

4.6　実行フローと API 連携

JavaScript 側のタイマー処理は setTimeout や setInterval 等で実装し，1 ターン終了後に次ターンを呼ぶことでブラウザフリーズを防いでいる．コードの実行時には runPythonAsync()API を用いて Pyodide 内にクライアントプログラムとサーバーロジックをロードし，ターンごとの盤面更新が行われるたびに updateGameDisplay 関数を呼び出して DOM を操作する．もしターン終了後に勝敗条件を満たせばサーバーロジックが「ゲーム終了」のメッセージをコンソールに出力し，ユーザーはブラウザ上で結果を即座に把握できる．

このように，フロントエンドの JavaScript と Pyodide 内の Python が協調することで，従来の CHaser 環境で必須だった Python インストールやサーバープログラムの常時起動，ポート番号の指定などが不要になり，ブラウザさえあればどの端末でも開発・対戦が完結する仕組みを実現している．

5.　評価実験

5.1　方法と対象

開発した CHaserWeb を札幌龍谷学園高等学校の高校生（20 名）と北海道情報大学の大学生 (11 名）に実際に体験してもらい，従来の CHaser 環境との比較を中心にアンケート調査を行った．高校生への調査（図 7）では，既に CHaser の開発経験を持つ層を対象とし，従来の煩雑なセットアップ手順との比較評価を求めた．一方，大学生の調査

図 7　札幌龍谷学園高等学校での評価実験の様子

では CHaser 未経験者も含み，環境構築不要性や汎用性を評価するよう促した．なお，高校生の被験者数は 20 名であるが，回答者不明の重複回答があったため，回答数は 21 となっている．

5.2　高校生アンケートの結果

従来の CHaser で開発する際に面倒と感じた点として以下の意見が挙げられた．

- アドレスを入力するのがめんだくさいです．
- 打つのが多いきつい
- プログラムを理解すること

CHaserWeb の良かった点として以下の意見が挙げられた．

- 従来のものよりも断然操作がやりやすくて，実行までの段取りが省かれてやりやすかった．
- ポート番号などを省いたらすごくやりやすくなったところ
- プログラミングとゲーム画面がひとつのページで見れること
- インデントの位置がわかりやすいようにラインが引かれていたのがありがたかった
- いちいち何回も設定したりしなくて良かったから楽

CHaserWeb の改善点として以下の意見が挙げられた．

- ゲーム画面をもっと見やすくしてほしいです．
- iPad でやる時に対戦画面が少し見えないところがある
- 自動セーブ機能がほしい
- 試合結果に score の表示があると判断が早い　AI のプログラム，マップ機能などの変更

CHaserWeb を活用できそうな場面として以下の意見が挙げられた．

- 大会などでそのまま使えそうだと思います．
- 授業で活用できる
- 中学生のプログラム授業の課題として簡単なプログラ

表 1　アンケート項目[*1]

番号	質問項目	選択肢・回答欄
Q1	プログラミングを授業以外でしていますか？	している / していない
Q2	従来の CHaser で開発を行う際にめんどくさいと思うことや嫌だと思うことがあれば教えてください[*2]	自由記述欄
Q3	CHaserWeb は従来の CHaser と比べて開発がしやすいと感じましたか？	1(開発しにくい)〜5(開発しやすい)
Q4	CHaser のプログラムを開発する際 CHaserWeb を活用したいと思いましたか？	1(活用する必要ない)〜5(活用したい)
Q5	CHaserWeb を活用し iPad で開発ができそうですか？[*3]	1(開発できない)〜5(開発できる)
Q6	CHaserWeb のここが良かったなどがあれば教えてください	自由記述欄
Q7	CHaserWeb で改善したほうが良いところがあれば教えて下さい[*4]	自由記述欄
Q8	CHaserWeb を活用できそうな場面があったら教えて下さい[*5]	自由記述欄
Q9	その他気になることがあれば教えてください	自由記述欄

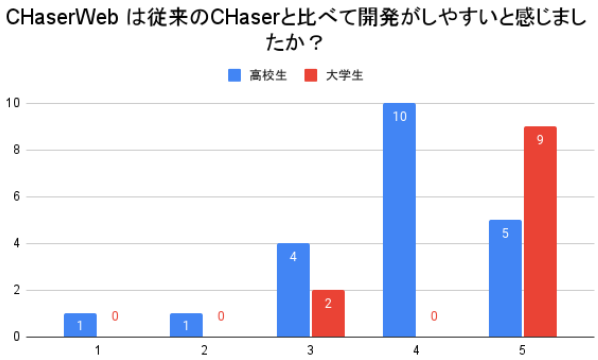

図 8　Q3 の回答結果

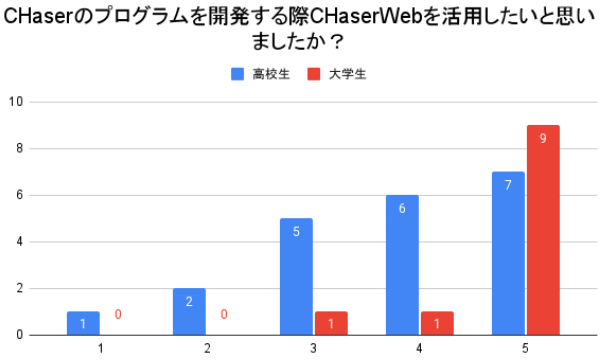

図 9　Q4 の回答結果

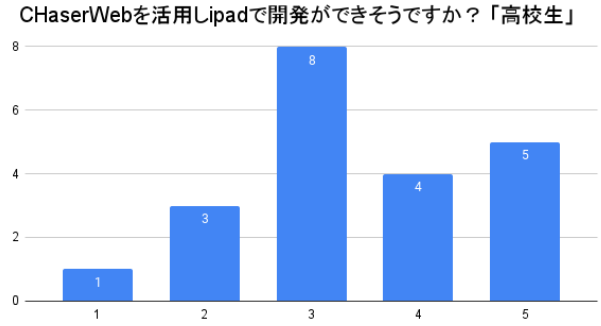

図 10　Q5 の回答結果

ミング言語，方法が挙げられていたので，そこから少しステップアップした授業はできそうだと思いました

- 小さい子の家庭学習

高校生を対象としたアンケートでは，「従来の CHaser で開発する際に面倒と感じた点」としてサーバーアドレスの入力，サーバー起動などの CHaser の仕様に加えて，コードの長さなどの初心者特有の難しさからくるものも見られる．CHaserWeb に対しては，「実行までの手順が少ない」といった肯定的な意見があり，従来環境よりも開発がしやすい，CHaserWeb を活用したいと感じる回答が図 8 と図 9 のように 3〜5 段階の上位評価に集中した．従来より設定不要で即実行が可能な点は大きな強みとして受け止められている考えられる．iPad での開発可能性についても図 10 のように一定の評価がされているしかし「iPad でやる時に対戦画面が少し見えないところがある」などの UI 面での最適化不足に起因する意見もあり改善の余地が見られる．改善要望としてはログやマップ表示のさらなる可視化，対戦相手 AI のカスタマイズやマップ構成の変更機能，モバイル端末向け UI 最適化などが挙げられた．対戦機能の充実や自動セーブ機能などを求める意見もあり，より多機能な学習環境への期待がうかがえる．

*1　大学生を対象とした評価実験時に一部の設問の文言が異なる
*2　大学生対象時：従来の CHaser で開発を行う際にめんどくさいな嫌だなと思うことがあれば教えてください
*3　高校生のみ
*4　大学生対象時：CHaserWeb で改善したほうが良いところはありますか？ある場合，その理由も教えてください
*5　大学生対象時：開発以外で活用できる可能性があると思ったら教えてください

5.3 大学生アンケートの結果

従来の CHaser で開発する際に面倒と感じた点として以下の意見が挙げられた.

- PC に Python の環境が整っている必要があること.
- 従来のでは python の環境がないと実機では動かせないのが，初心者には厳しいなと感じました.
- コマンドプロンプトを起動して，1 回 1 回起動しなければならないこと.

CHaserWeb の良かった点として以下の意見が挙げられた.

- CHaser 自体の用意自体がめんどいと感じたが，Web の方はすぐに用意でき，面倒さがない他またやりたいときにはソフトの場合は再起動が必須だが Web なら再読み込みで十分なのがいい.
- アップロード，ダウンロード機能がついているので場所・端末選ばずに開発することができる.
- アップロードして実行するのが簡単だった
- Web 版ではワンクリックで動作を見れるため，とにかく操作が簡単なのがいいと思いました.

CHaserWeb の改善点として以下の意見が挙げられた.

- マップの切り替えの実装
- アイテムだったりのデザインをもう少し視覚的にわかりやすいほうが楽しそう.
- ログにエラーの部分は赤文字などの変化がついていて欲しい.
- 相手側 bot のスクリプト編集機

CHaserWeb を活用できそうな場面として以下の意見が挙げられた.

- 個人大会が開きやすいかもしれない

大学生を対象としたアンケートで，「Python 環境構築が必要」「サーバーやポートの設定に時間を取られる」といった従来の CHaser 特有の問題点が指摘された．CHaserWeb については「ワンクリックですぐ実行できる」「複雑な設定なしに動くのは初心者にも優しい」と高評価を得ており，従来環境よりも開発がしやすい，CHaserWeb を活用したいと感じる回答が図 8 と図 9 のように 5 段階評価で 4 や 5 をつける回答が大多数を占める．改善要望としては，アイテムのデザインの変更，マップのランダム化，相手 AI スクリプトの自由編集など発展的機能への期待が示された.

5.4 総合評価

高校生と大学生のアンケート結果を総合すると，CHaserWeb は従来の CHaser 環境に比べて，環境構築の煩雑さを大幅に低減し，即時実行や単一画面での盤面表示など UI/UX が改善された点で非常に高く評価されている．高校生の多くが「従来より開発がしやすい」と回答し，大学生からも即時活用が可能と好意的な意見が得られた.

一方で，対戦機能の強化やログの表示のさらなる改善，対戦 AI 変更機能，マップ変更機能など拡張要望も数多く見られたことから，今後はこうしたニーズへの対応を優先的に行う必要がある.

6. おわりに

本研究では，U-16 プロコンに用いられる CHaser の学習環境において，環境構築の煩雑さや端末依存の問題を解決するために，ブラウザ上で動作する CHaserWeb を提案し，その有用性を検証した．Pyodide を活用することで Python 実行環境をブラウザ内に完結させ，OS や端末依存を解消しつつ，即時実行とデバック機能，盤面表示を提供している．アンケート調査の結果から，従来環境と比べて簡単に導入できる点や UI/UX の分かりやすさが高く評価された一方で，相手 AI やマップ構成のカスタマイズ機能，ログの可視化などの要望が挙げられた.

CHaserWeb は，Web サービスとして公開しており[*1]，アカウント不要で体験できる．また，MIT ライセンスのオープンソースソフトウェアとしてソースコードを GitHub に公開している[*2]．今後は，こうした機能拡張やモバイル端末向けの UI 最適化，動作の安定性向上などを進めることで，より幅広い学習者が CHaser の対戦型プログラミングを気軽に体験できる環境を目指す.

謝辞

本研究は，札幌龍谷学園高等学校に評価実験のご協力をいただきました．心より感謝を申し上げます.

参考文献

[1] U-16 プロコン札幌大会実行委員会：U-16 プログラミングコンテスト札幌大会公式ホームページ，https://sapporo.u16procon.org/. 参照日: 2024-12-16.
[2] paiza 株式会社：paiza.io, https://paiza.io/ja. 参照日: 2024-12-20.
[3] Replit: Replit, https://replit.com/. 参照日: 2024-12-20.
[4] Google LLC: Google Colab, https://colab.research.google.com/?hl=ja. 参照日: 2024-12-20.
[5] Processing Foundation: p5.js, https://p5js.org/. 参照日: 2024-12-20.
[6] CodeMirror Project: CodeMirror, https://codemirror.net. 参照日: 2024-12-21.
[7] Pyodide Project: Pyodide, https://pyodide.org. 参照日: 2024-12-21.

[*1] https://shin3391.github.io/CHaserWeb/
[*2] https://github.com/shin3391/CHaserWeb