プログラマブルな HTML 用穴埋め問題作成ツール

北里大学一般教育部　情報科学単位
福田　宏

要旨

Web ページに設置するプログラマブルな穴埋め問題作成ツールを開発した。プログラマブルとは，正解を JavaScript や正規表現によって記述できるという意味である。それによって，文字列としては判定しにくい，プログラムのコードや数式なども正誤判定できるようになる。学生は，この穴埋め問題で正誤判定を繰り返して正解にたどり着くことで，コードや数式を独力で書けるよう学習することができる。プログラミングの講義でアンケート調査したところ，大多数の学生がこの穴埋め問題は有効な学習補助ツールであると評価した。

キーワード：穴埋め問題，プログラマブル，JavaScript，正規表現，HTML

Programmable fill-in-the blank learning tool for HTML

Computer sciences section, college of liberal arts and sciences, Kitasato University,
Hiroshi FUKUDA

Abstract

This paper reports on our recent development of the programmable fill-in-the blank learning tool, which can be readily used on any web pages. Here, “programmable” means that the correct answers can be programmed in JavaScript or in regular expressions. This makes it is possible to check the answers not simply as a string of characters but as programming codes or as mathematical expressions. Students can arrive at the right codes or expressions by themselves through trial and error with this learning tool. Majority of students rated this tool as effective in a post-intervention survey.

Keywords: fill-in-the blank learning tool, programable, JavaScript, regular expressions, HTML

1. はじめに

著者の担当する講義では，Java [1]によってプログラミング言語教育をおこなっている。2020 年度はコロナ禍で，その講義はオンライン開講となった。オンライン教材は，課題提出で利用していた学習支援システム（Learning Management System, LMS）Moodle [2]のコースに，講義動画と小テスト [3]を追加して作成した。

翌 2021 年度，講義は対面形式に戻り，講義動画は予習復習用，小テストは講義中に学生がプログラミング課題に取り組むための補助として再利用した。その際，小テストの設定は，そのまま，受験終了後に正解が見えるようにしておいた。

ところが，対面講義が始まり，様子を観ていると，多くの学生は小テストを解かずに受験終了の操作をして正解を先に見ている。それでは，何も考えずに，ほぼコピー＆ペーストで課題を完成できてしまう。突然のオンライン講義ではそれも仕方ないとは思うが，翌 2022 年度は，正誤チェックはできるが，正解は見えないように Moodle の設定を変更した。

Moodle では正誤のチェックも受験終了操作の後で行われる。すると，誤答要素を再判定するには，小テストを再受験せねばならず，誤答要素のみ再判定することができない。これでは，穴埋め要素の数やその個々の入力量が多い場合，学生が再判定を繰り返して利用するには使いにくそうである。Moodle の設定を色々いじってみたのだが，これを克服する上手い設定はないようだった。

そこで，Moodle 小テストの利用は諦め，Web ページで提供している講義資料に，HTML の input タグで簡単な穴埋め問題を自作することにした。正解を得るまで，試行錯誤したい学生にとっては，正誤判定が簡単にできるので，期待通り使いやすそうであった。

本論文では，こうして講義中に作り始めた自作穴埋め問題を発展させ，Moodle で不便を感じていた点などを踏まえ，穴埋め要素ごとに正誤の判定ができ，1)正規表現 [4]による正解の表現，2) JavaScript [5]による正解の表現，3）順不同の穴埋め要素，へ対応した HTML のページ作成ツールを紹介する。このツールは，既存の HTML に変更を加えることなく，学生の解答状態をウェブブラウザに保持する機能，正誤判定結果を HTML の要素に付加する機能，正誤判定された解答を提出するための機能などを有している。正誤判定は JavaScript で実行されるため，解答をサーバーに送信することなく，クライアントで判定される。そのため，サーバーとの通信設定やサーバーの負荷などを考える必要はない。

第 2 章では，本ツールの詳しい内容，HTML への設置方法と使い方を示す。第 3 章では本ツール独自の JavaScript と正規表現によるプログラマブルな正解の表現方法，第 4 章では，それによって可能となる，数式をはじめとする様々な応用例を示す。第 5 章は，本ツールを使った講義を受講した学生へのアンケート結果である。第 6 章は，その後追加したクリップボードを介した解答の送信機能，第 7 章は，本ツールと既存関連システム・研究との比較，第 8 章はまとめである。

2. 設置方法

本ツールは，通常の HTML ファイルにツール固有の記述を若干追加して利用する。図 1 は簡単な HTML ファイルで本ツールを利用する例である。下線部が本ツール固有の記述である。図 2 は図 1 の HTML をブラウザで表示したところである。

```
 1 <!DOCTYPE html>
 2 <html>
 3 <head>
 4   <meta charset="UTF-8">
 5   <script src="ans.js" type="text/javascript"></script>
 6   <script src="a64.js" type="text/javascript"></script>
 7   <title>簡単な例</title>
 8 </head>
 9 <body onload="loadallvalues();">
10 <p id="Q1">
11 <b id="Q1t">問 </b>
12 1 から 10 までの和は<input type="text" size="4">, 一般に,
13 1 から n までの和は<input type="text" size="4">である。
14 <input type="button" name="check" value="check" onclick="check(this);">
15 </p>
16 <input type="button" value="clear all" onclick="clearallvalues();">
17 </body>
18 </html>
```

図 1 本ツールを利用した簡単な HTML

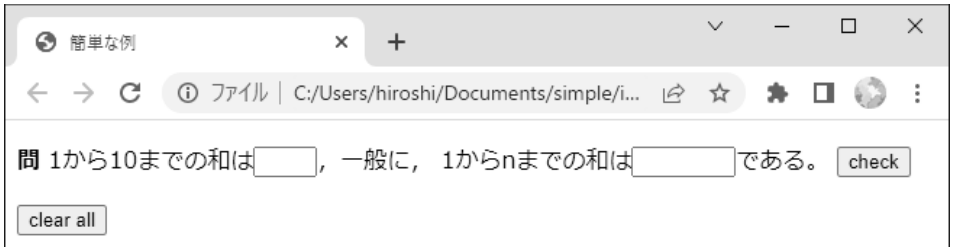


図 2 ブラウザで表示したところ

図 1 に従って，本ツールの HTML への設置方法を説明する。まず，本ツールを Web サイト [6]から取得し，ツール本体である JavaScript ファイル ans.js と，第 3 節の方法で作成する正解ファイル a64.js を，それぞれ，図 1 の 5−6 行目

```
<script src="ans.js" type="text/javascript"></script>
<script src="a64.js" type="text/javascript"></script>
```

のように，HTML のヘッダ領域で読み込む。HTML の仕様上，このようにブラウザで読み込んだファイルは，閲覧者も読むことができる。例えば，この 2 つのファイルは，下線を引

いたファイル名をクリックすると読むことができる。それは，正解ファイル a64.js をクリックすれば正解が見えてしまうことを意味する。そこで，その対策として，本ツールでは，正解ファイルは base64 [7]でエンコードして，同デコーダーを使わなければ読めないようにしてある。図 3 はエンコード前の正解ファイルであり，フォーマットを知らなくても，波線部分から正解が 55 と n*(n+1)/2 であると推測可能である。

```
ans64 = {'Q1': ["55","{var c=n=>eval(ans)==n*(n+1)/2;c(5)&&c(10)&&c(3
3);}"]};
```

図 3 正解ファイル a.js

一方，図 4 は図 3 をエンコードしたもので，たとえ base64 の定義を知っていたとしても正解を推測することは難しい。

```
ans64 = {'Q1': ['IjU1Ig==','Int2YXIgYz1uPT5ldmFsKGFucyk9PW4qKG4rMSkvM
jtjKDUpJiZjKDEwKSYmYygzMyk7fSI=']};
```

図 4 エンコードした正解ファイル a64.js

次に，図 1 の 9 行目

```
<body onload="loadallvalues();">
```

のように，HTML の body タグに解答状態をロードする関数を書く。これによって，ブラウザの Web Storage にローカルに保存される解答状態を，このページを表示する度に読み込んで，解答欄をブラウザ終了時の状態に戻す。もし，この過去の解答状態を全てクリアして，何も解答していない初期状態に戻したい場合は，図 1 の 16 行目のように，clearallvalues()関数を call するボタンを設置する。

次に，問題は，図 1 の 10－15 行目

```
<タグ名 id=”問題 id”>
    …
    <input type="button" value="check" name=”check”
     onclick="check(this);">
    …
</タグ名>
```

のように，いくつかの穴埋め要素と，その正誤を判定する check(this)関数を call する name 属性が check であるボタンを，任意の HTML タグ（図 1 では p タグ）で囲み，そのタグに

id（図 1 では Q1）をふることで作成する。この id が正誤判定で使われる問題 id となる[1]。

問題に埋め込む「穴埋め要素」は，問題 id の振られたタグと，対応する終了タグの間に，図 1 の 12－13 行目のように，単一行入力の text input タグ[2]，

```
<input type="text" size="4">
```

または，複数行入力の textarea タグ[3]

```
<textarea rows="2" cols="50" spellcheck="false" style="overflow:
 hidden;"></textarea>
```

で設置する。単一行入力の text input タグは list 属性で datalist タグの id を指定することで，ドロップダウンリストによる選択型入力要素にすることもできる。

こうして設置された問題は，check ボタンを押すことで，onclick 属性に記述された check(this)関数が call されて，問題 id の振られたタグで囲まれた範囲の穴埋め要素に対して正誤判定がなされ，正解の要素は緑（lightgreen），不正解の要素は赤（pink）になる。未解答の要素は変化しない。同時に，Web Storage に解答が記録される。

なお，問題番号などを表示する要素に，図 1 の 11 行目のような，問題 id の末尾に t を付した id を振っておくと，その問題の穴埋め要素が全て正解の場合，この要素の末尾に✅がつく。

3. 正解の書き方

正解は，JavaScript のオブジェクト ans64 に，問題 id をシングルコーテーションで囲んだ文字列キーとする配列で書く。その配列要素が順にその問題の穴埋め要素に対する正解である：

```
ans64 = {
  '問題 id1': [要素 0 の正解, 要素 1 の正解, …],
  '問題 id2': [要素 0 の正解, 要素 1 の正解, …],
  …
}
```

---

[1] 問題 id はアルファベットで始まる半角英数でなければならない。

[2] size=”4”はこの要素の横幅。readonly 属性が付与された text input は対象外。

[3] rows="2" cols="50"は textarea の高さと幅，spellcheck="false"はスペルチェックをしない設定，style="overflow: hidden;"は横スクロールバーを表示しない設定。

正解が順不同である穴埋め要素の組は，その配列内で，さらに配列にする。例えば，問題 id1 の要素 0 と要素 1 の正解が，順不同である場合は，

```
'問題 id1': [[要素 0 の正解，要素 1 の正解]，要素 2 の正解，…]，
```

となる。

正解は，1）シングルコーテーションまたは 2）ダブルコーテーションで囲った文字列か，3）スラッシュで囲んだ正規表現である。

1）シングルコーテーションで囲んだ文字列は，解答と文字列として比較され，一致すれば，解答が正解であると判定する。

2）ダブルコーテーションで囲んだ文字列は，数値・数式としてとして評価され，解答も数値・数式としてとして評価した上で比較され，一致していれば解答が正解であると判定する。

さらに，ダブルコーテーションで囲んだ文字列の先頭を{末尾を}にすると，JavaScript のプログラムとして評価される。正解なら true，そうでなければ false になるようにプログラムする。そのプログラムの中で，解答の代入された変数 ans と，同じ問題 id の 0, 1, 2, …，番目の穴埋め要素に記入された文字列が代入された配列 ansa[0], ansa[1], ansa[2],…，を参照することができる。

3）スラッシュで囲んだ正規表現は，解答とマッチすれば解答が正解であると判定する。正規表現の書き方は，JavaScript における正規表現の書き方に従う。

図 3 は問題 id が Q1 の問題に，穴埋め要素が 2 個ある正解ファイルである。第 1 の穴埋め要素は，ダブルコーテーションで囲まれた数値を判定するタイプ，第 2 の要素はダブルコーテーションと中カッコで囲まれた，変数 ans を JavaScript のプログラムで判定するタイプである。

なお，正解との比較に先立ち，解答文字列からは，先頭・末尾のスペースやタブは削除される[4]。さらに，正解が正規表現で，半角空白を含まない場合，解答文字列の区切り文字（半角空白，改行，タブなど）は，単語区切りに必要なものは半角空白 1 つに置換され，単語区切りに必要ないものはすべて除去される[5]。この前処理は，後述するように，区切り文字の入れ方が自由なプログラムコードの正解を簡潔に書くための仕様である。

正解ファイルができたら，適当な名前の JavaScript ファイル，例えば a.js として保存し，コマンド

---

[4] JavaScript の trim 関数による処理。

[5] この前処理を行うには，正規表現で半角空白を¥s と記述すればよく，逆に，この前処理を避けるには，正規表現の先頭に *と記述すればよい。

```
perl ans.pl a.pl
```

を打って[6]，perl [8]のスクリプト ans.pl によって，ans64 の配列要素を base64 でエンコードしたファイルを生成する[7]。エンコードされた正解ファイル名は，正解ファイルのベース名に 64 をつけたもの，例えば，正解ファイルが a.js なら a64.js となる。既に述べた通り，図 4 は図 3 をエンコードしたものである[8]。

4. 正解記述の例

JavaScript と正規表現を用いることで，プログラムのコードや数式をはじめとする様々なタイプの正誤判定が可能になる。以下に，講義で用いた例を正解配列と共に示す。

4.1. 文字列の判定

（例 1）正解をシングルコーテーションで囲み，文字列を判定する：

OR ゲートの動作は，スイッチをどのように接続したものと同じか。漢字二文字で記入しなさい。 [　　]

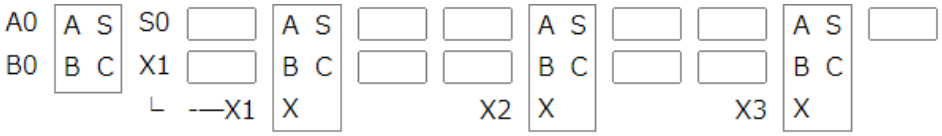
正解配列 `['並列']`

（例 2）文字列の判定も，次のように，穴埋め要素を多数 table タグで配置することで，複雑な問題に応用することができる：

4 桁の 2 進数の加算をおこなう回路を描け。 入出力

```
  A3 A2 A1 A0
+ B3 B2 B1 B0
-------------
  S3 S2 S1 S0
```

または，端子（X1～X3）のアルファベット番号を入れよ。



---

[6] Linux コマンド。Windows ではコマンドプロンプトで `wsl␣-u␣root␣--␣` に続けて打つ（WSL，Windows Subsystem for Linux がインストールされている必要がある）。

[7] ans.pl は，エディタの外部コマンド実行機能で，正解ファイルを編集しながら実行できように設定しておくと便利である。Visual Studio Code での設定は [6]参照。

[8] 正解ファイルの修正をブラウザに反映させるには，キャッシュをクリアして再読み込みする「ハード再読み込み」Shit+Ctrl+R する必要がある。

正解配列 `['A1','S1','A2','S2','A3','S3','B1','X2','B2','X3','B3']`

4.2. 数値の判定

（例 3）図 2 の最初の穴埋め要素は，正解をダブルコーテーションで囲んで，数値を判定する。正解は図 3 の配列の最初の要素”55”である。

4.3. 正規表現による判定

（例 4）入力した文字列が，アルファベット 2 文字の次に先頭が 2 の 5 桁の数字の形式の学籍番号か，正規表現で判定する：

7 桁の学籍番号(pp20100 など)を記入してください。[　　]

正解配列 `[/^[A-Za-z]{2}2¥d{4}$/]`

（例 5）入力されたプログラムコードを，空白や改行の入れ方に依らずに判定できるように，正解の正規表現にスペースを含めずに¥s で記述する：

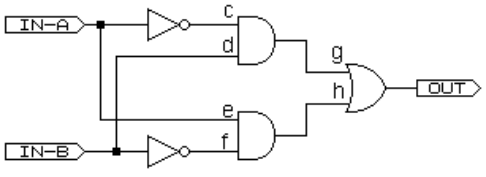
次の if 文を「s が b2 なら setBackground(Color.WHITE); 」に修正しなさい。

```
if(s==b){
  setBackground(Color.GREEN);
}
```

正解配列 `[/^((if|else¥sif)¥((s==b2|s!=b1|!¥(s==b1¥))¥)|else)({setBackground¥(Color¥.WHITE¥);}|setBackground¥(Color¥.WHITE¥);)$/]`

（例 6）文字間に空白があってもよい場合：

回路の各点の動作から XOR の動作を説明する真理値表を完成させなさい。



A	B	c	d	e	f	g	h	out
0	0	1	0	0	1	0	0	0
0	1							
1	0							
1	1							

正解配列 `[/^1 *1 *0 *0 *1 *0 *1$/, /^0 *0 *1 *1 *0 *1 *1$/, /^0 *1 *1 *0 *0 *0 *0$/]`

（例 7）記述問題などで，複数のキーワードが入っているか正規表現で順不同で調べるには次のように書く：

`(?=.*キーワード 1)(?=.*キーワード 2)(?=.*キーワード 3)`

4.4. JavaScript による判定

正解をダブルコーテーションと中カッコで囲み, 解答の代入された変数 ans を JavaScript で処理して，正誤に応じて真理値 true，false を返す。

（例 8）誤差付きの数値の判定。Math.abs は JavaScript で絶対値を計算する関数，比較式の結果は真理値である。

0.02125±0.001 か？
正解配列 `["{Math.abs(ans-0.02125)<0.001;}"]`

（例 9）図 2 の第 2 の空所は，入力された数式が，n*(n+1)/2 であるか，数式として確認している。そのために，n 次多項式の場合 n+1 点で値が一致していれば，式として等しいことから，テーラー展開できる数式は，幾つかの値を代入して計算される値が正しければ正しいと推測して，解答された数式の正誤を判定する。

図 3 の配列の第 2 のダブルコーテーションと中カッコで囲まれた要素は，引数 n の値に対して，解答の式と n*(n+1)/2 の値が等しければ true，そうでなければ false を返す JavaScript の関数 c(n)を定義して，c(5)，c(10)，c(33)の論理積を返す。eval は文字列をプログラムとして評価する関数で，eval(ans)で ans に代入された数式をプログラムとして評価している。なお，記述を短くするために，アロー記法で関数を定義している。

（例 10）f(c)(b-a)，c=(a+b)/2 という多次元の関数形が未定な関数に対しても，具体的な関数 f(c)=√c に対して，2 点(a,b)=(0.2,0.6)，(0.1,0.11)で数式の値が正しければ数式が合っていると推測して正誤判定する：

解答の数式が f(c)*(b-a)であるか（c=(a+b)/2 であることは与えられている）。
正解配列 `["{var f=x=>return Math.sqrt(x), d=(a,b)=>{var c=(a+b)/2; retu rn f(c)*(b-a)==eval(ans);}; d(0.2,0.6) && d(0.1,0.11);}"]`

同じように，条件式の場合も，解答された条件式に含まれる変数に，（幾つかのクリティカルな）数値を代入して，条件式が正しいかどうか確認することができるだろう。

4.5. JavaScript と正規表現を組み合わせた判定

JavaScript の文字列を置換する関数 replace や文字列の一致を調べる match 関数で正規表現を使うことができるので，JavaScript と正規表現を組み合わせた正誤判定も行うことができる。

（例 11）数式処理システムで用いられる形式で数式を解答してもらい，数式処理システムの形式を，置換関数 replace で JavaScript の形式に変換して評価する：

縦横の画素数が x, y，サイズ L インチのディスプレィの解像度[ppi]を計算する x, y, L の式を求めなさい。式は，和差積商,べき乗,平方根，括弧を ＋ - * / ^ sqrt，()と書いて下さい（例えば（-b+sqrt(b^2-4*a*c))/(2*a）など)。
正解配列 `["{var c=(x,y,L)=>eval(ans.replace(/¥^/g,'¥*¥*').replace(/sqrt/g,'Math¥.sqrt'))==Math.sqrt(x*x+y*y)/L; c(1280,1024,17)&&c(640,480,12)&&c(320,240,2.8)&&c(720,1280,4.6);}"]`

（例 12）単位付き数値の数値部分を，数値以外を無視する数値への変換関数 parseFloat で評価し，単位は match 関数で評価する。次の例では，数値は例 8 の方法で±0.1 の誤差は許容している。

動画は，静止画を 1 秒間に 30 枚紙芝居のように切り替える。圧縮していない 640×480 画素の動画 30 分のファイルサイズはいくらか。適当な SI 接頭辞をつけて答えよ。
正解配列 `["{Math.abs(parseFloat(ans)-49.7664)<0.1&&ans.match(/G¥s*B$/)!=null;}"]`

4.6. 別の穴埋め要素の値の参照

（例 13）n 個の数の比の判定。n 個の数を全て参照する必要があるので，以下のように，全ての穴埋め要素の値を保持している ansa 配列の要素 ansa[0], …, ansa[n-1]を参照して計算する。判定プログラムは，n 個の値をベクトルとして規格化して，同じく規格化した正解と比較するので，n 個の穴埋め要素の全てで同じである。

590nm 単色光の RGB 表色系の 3 刺激値を求めよ。
Q=([　　], [　　], [　　])
正解配列 `["{var i, n=v=>v[i]/Math.sqrt(v[0]**2+v[1]**2+v[2]**2);((v,u)=>{var r=true;for(i=0;i<3;i++)r=r&&Math.abs(n(v)-n(u))<0.01;return r;})(ansa,[0.3093,0.0975,-0.0008]);}",`以下同じものが 2 つ]

（例 14）最初の空所に入れたアルファベット大文字の姓を ansa[0]で参照して，2 進数で文字コードを入れる：

1 文字を 8 ビットであらわす。A は 01000001。B～Z は順に 1 つづ増えた数。2 進数で自分の苗字をつくってみなさい。
苗字 [　　]　2 進数のデータ [　　]
正解配列 `[/^[A-Z]+$/, "{((a,b)=>{for(var i=0; i<a.length; i++) if(a.charCodeAt(i)!=parseInt(b.substr(8*i,8),2)) return false; return true;})(ansa[0],ans);}"]`

5. 学生の反応

2022 年度講義[9]の最終日に，Moodle を使って，受講生に Moodle の小テストと本システムによる学びやすさを比較する匿名アンケートを行った。以下の Q1—Q6 の質問に，それぞれ,「そう思う」,「ややそう思う」,「あまりそう思わない」,「そう思わない」，の 4 段階選択で回答してもらった。回答率は，受講者 51 名[10]中 50 名であった。

受講生の反応は，以下の通り概ね良好であった。各質問文末に，肯定的回答,「そう思う」または「ややそう思う」と回答した学生の比率を示す。図 5 は全回答の比率を示す横棒グラフ，数字は各回答の実数である。

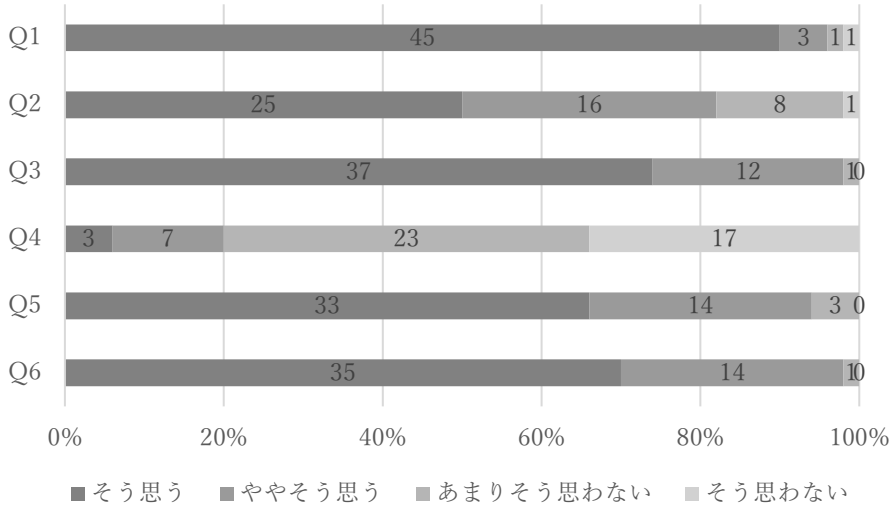


図 5 アンケート結果

Q1.『Moodle より, 授業資料で直接解答する方が, 解答の途中で何度でも判定できるので, 使いやすいと思いましたか？』96%。

この回答から，本システムの目的である「解答しながら正誤判定できる」機能は好評であったといえる。

Q2.『授業資料で直接解答する場合は，System.out.println(“最大値=”+d[0]); のようにプログラムの文全体を入力して判定できました。一方, Moodle では，System.out.println(“最大値=”+); のような，文の一部の穴埋め形式でした。前者のように文全体を答える方が，課題に取り組むのに役立つと思いましたか？』82%。

これは，本システムの正規表現による正誤判定によって可能になる解答形式で，解答は

[9] 情報科学 B

[10] 履修放棄者 3 名を除く。

難しくなるはずだが，学生はこれも肯定的に評価している。

Q3.『授業資料で直接解答する場合は，10000*10+(omosa-10000)*7 のような数式も，数式として同一ならどのように書いても（例えば 30000-7*omosa と書いても）判定できました。このような数式の判定は，課題に取り組むのに役立つと思いましたか？』98%。これは，本システムのコードによる正誤判定によって可能になる数式の判定で，肯定的に評価されている。

Q4.『授業資料で直接解答する場合，解答は教員には送られません。それでは，「解答が成績に反映されないので，問題に取り組む意味がない」と思いますか？』30%

Q5.『解答が教員に送られなくても，授業を理解するには十分役立つと思いましたか？』94%

Q4 と Q5 から，本システムがサーバーに接続されていなくてもあまり支障ないと感じていることがわかる。

Q6.『授業資料で直接解答するやり方は，他の科目でも役立つと思いますか？』98%

これから，学生は本システムが他の授業科目でも有効だと感じていることが伺える。

6. クリップボードを介した解答の送信機能

アンケートの後，次年度に向けて，学生から解答をコピーして送付してもらうための関数 copyallvalues と，それを解読するバッチファイル ansclip.bat を追加した。

関数 copyallvalues は，全ての問題の穴埋め要素の正誤判定を行い，正解でない要素を base64 でエンコードして, クリップボードにコピーする。一方, バッチファイル ansclip.bat[11] は，クリップボードを監視して，エンコードされた解答がコピーされると，それをデコードしてクリップボードの内容を，

```
score ¥t                    ¥t     ¥t 問題 id ¥t 問題 id ¥t …
正答率 ¥t 年/月/日 時:分:秒 ¥t 乱数 ¥t 値      ¥t 値      ¥t …
正答率 ¥t 年/月/日 時:分:秒 ¥t 乱数 ¥t 値      ¥t 値      ¥t …
…
```

のような，見出し行と解答行からなる，タブで区切られた tsv（tab separated value）形式に変換する。¥t はタブ，1 行目は見出しである。2 行目以降は，各解答の，正答率，コピー時刻，コピー防止のための乱数[12]と解答の値；空白は正答，値のあるものは誤答，-は空白であ

[11] 配布 zip ファイルに含まれる。同ファイルに含まれる ansclip.pl がないと動かない。

[12] エンコードされた解答をコピーすると乱数が一致するのでコピーが発覚する。

る。

tsv 形式のデータは，表計算ソフトに直接貼り付けることができる。したがって，copyallvalues 関数とバッチファイル ansclip.bat によって，以下のようにして教師は即座に解答一覧を得ることができる：

1. Web ページに copyallvalues 関数を call するコピーボタン

   ```
   <input type="button" value="コピー" onclick="copyallvalues();">
   ```

   を設置する。
2. 学生は，コピーボタンをクリックして，クリップボードにエンコードされた解答をコピーして，LMS の記述問題に貼り付けることで解答を提出する。その際，解答はエンコードされているので学生が改竄することはできない。また，乱数が埋め込まれているので他の学生の解答をコピーすることもできない。
3. 教師は，バッチファイル ansclip.bat を起動しておき，LMS から全学生の解答をダウンロードして，スプレッドシートで表示する。そして，学生が解答を貼り付けた記述問題の列を選択してクリップボードにコピーする。クリップボードのデータは ansclip.bat によって上述の tsv 形式に即座に変換される。
4. 選択を開始したセル（最初の解答）の右上のセルで「貼り付け」操作を行うと，クリップボードから変換された tsv 形式のデータが貼り付き，得点と誤答，記述問題がエンコードされた解答の横に，見出し付きで挿入される。

図 6 は，正解が図 3 である図 2 の問題に対するエンコードされた 2 名の解答をスプレッドシートで開いたところである。

	A	B	C	D	E	F	G
1							
2	c2NvcmUJCQlRMQkKNTAlCTIwMjMvMi8yNyAxNToyOjM1CTkzNgkJbi8y						
3	c2NvcmUJCQlRMQkKNTAlCTIwMjMvMi8yNyAxNTozOjI5CTg4MgktCQ==						
4							

図 6 エンコードされた 2 名の解答

図 7 は，ansclip.bat を起動して，図 6 の A 列のセルを選択して，クリップボードにコピーし，選択を開始したセル（A2）の右上のセル（B1）に貼り付けた様子である。第 1 行の見出しの下に，順に，エンコードされた解答に対応したデコードされた解答が並ぶ。C 列はエンコードした時刻，D 列は乱数，E 列は問題 id が Q1 の第 1 の穴埋め要素，F 列は第 2 の要素である。第 2 行の解答は，Q1 の第 1 穴埋め要素は空白なので正解，第 2 要素は値 n/2

が入っているので不正解，第 3 行の解答は第 1 要素が-なので未記入，第 2 要素は空白なので正解である。

	A	B	C	D	E	F	G
1		score			Q1		
2	c2NvcmU.	50%	2023/2/27 15:02	936		n/2	
3	c2NvcmU.	50%	2023/2/27 15:03	882	-		
4							

図 7 tsv 形式の解答一覧を貼り付けた様子

なお，学籍番号や氏名，記述問題に対応するために，問題 id が「id」である要素と，問題 id の末尾が doc である要素は，正誤に関わらず ansclip.bat で解答値として表示される[13]。また，必須と発展のように問題のカテゴリーを分けるために，copyallvalues 関数の引数として文字列を指定すると，問題 id の name 属性の値がその文字列である問題のみが ansclip.bat で表示され，見出しに「score for 文字列」とカテゴリーが表示される。

7. 既存のツール・研究との比較

本ツールと類似の既存ツールとして，Moodle の小テストの穴埋め問題が挙げられる。Moodle の穴埋め問題は，第 1 章で述べたように，正誤確認を小テスト受験の終了のタイミングでしか行えない点，スクリプトによる複雑な判定ができない点，任意の HTML に埋め込めない点で本ツールより使いにくい。一方で，本ツールは JavaScript を理解していないと使いこなせない点，サーバーと通信できない点が，Moodle に対するデメリットである。

また，Moodle にはシステムを後付けで拡張できる機能がある。Moodle は拡張機能によって，穴埋め問題の正誤判定に正規表現を利用できる拡張 [9]，数式処理システム Maxima の数式処理エンジンを用いたオンラインテスト STACK によって小テストの解答として数式を用いることのできる拡張を導入することができる [10]。STACK による数式の扱いは，数式処理を用いた本格的なものであるが，その反面，導入や利用は敷居が高く，また処理能力の高いサーバーが必要とされる。本システムの数式の扱いは，JavaScript による簡易的なものであるが，そのため，導入や運用は STACK よりはるかに容易であり，サーバーに本システムのための処理能力は要求されない。

次に，プログラミング学習に特化したツールとして，東京大学の PLUGS UT [11]をはじめとしたプログラム自動正誤判定システム [12, 13]がある。これは，プログラムコードを入力すると，プログラムを模範プログラムと比較して，題意を満たしているか AI によって（完

[13] 問題 id が「id」である問題は最初に表示される。

全ではないが）判定してくれるものである。しかし，やはり導入の敷居は高く，一方，本システムのプログラムコードの扱いは，JavaScript と正規表現による発見的対処療法であるが，導入は容易である。また，これら数式やプログラムコードなどの専用システムに比べて，本システムは分野を限定しない汎用システムである点は，本システムの優位点である。

8. おわりに

本稿では，正解の判定に JavaScript と正規表現を利用できる，HTML 用穴埋め問題作成ツールを紹介した。JavaScript と正規表現を活用することで，多彩な正解判定が可能になる。JavaScript コードは数値計算や工夫によって数式を扱えるとので様々な理系の講義で，正規表現は文系の講義でも利用できるのではないだろうか。

本ツールはサーバーとは通信せず単独で動作する。したがって，前節で強調したように，設置・導入は容易である。しかし，学生の問題への取り組み状況をモニターできないので，主な用途は学生自身の講義理解補助である。学生へのアンケートによると，サーバーと通信しないことが，授業理解の補助ツールとしての決定的デメリットではないようである。なお，その後追加した，解答をエンコードしてクリップボードにコピーする機能は，LMS を利用して取り組み状況を送信することを意図している。今後，実際の講義で利用法や効果を探っていきたい。特に，広く用いられている LMS，Google の Classroom [14]は穴埋め問題の機能が貧弱だが，記述問題の解答をスプレッドシートで一覧することができるので第 6 章に示した方法で本ツールを穴埋め問題の補強に使うことができるだろう。

将来への課題として，講義の理解補助のためには，現在の単なる穴埋め形式に加え，別の形で，正解に至るヒントを示す機能が追加されるとよいかも知れない。また，サーバーと通信する機能を加え，Moodle など LMS との連携もできればより便利だろう。あるいは，LMS の穴埋め問題で正規表現や JavaScript を利用できるようにする LMS の拡張機能を提供することも考えられる。実際，Moodle には正規表現で正誤を判定するプラグインが存在する [9]。

最後に，本ツールは perl を使っているので，Windows で利用するには WSL（Windows Subsystem for Linux）が必要になる。そのどちらにも馴染みのない読者のために，サイト [6]に本システムを Windows で使う手順を掲載した。

参照文献

[1] K. Arnold, J. Gosling and D. Holmes, The Java programming language, Addison Wesley Professional, 2005.

[2] Moodle community, "Welcome to the Moodle community," [Online]. Available: https://moodle.org/.

[3] Moodle community, "穴埋め問題（ Cloze ）タイプ," [Online]. Available: https://docs.moodle.org/3x/ja/穴埋め問題_(_Cloze_)_タイプ.

[4] A. V. Aho, Algorithms for finding patterns in strings, Handbook of theoretical computer science (vol. A): algorithms and complexity, Cambridge, MA: MIT Press, 1991.

[5] TC39, "Specifying JavaScript," [Online]. Available: https://tc39.es/.

[6] H. Fukuda. [Online]. Available: https://sites.google.com/view/kilin/software/ans-js.

[7] The RFC Series (ISSN 2070-1721), "The Base16, Base32, and Base64 Data Encodings," [Online]. Available: https://www.rfc-editor.org/rfc/rfc3548.

[8] L. Wall, T. Christiansen and J. Orwant, Programming perl, Reilly Media, Inc., 2000.

[9] J. Rézeau and N. Dunand, "Regular expression short answer," [Online]. Available: https://moodle.org/plugins/qtype_regexp.

[10] 中村泰之, 中原敬広 and 秋山實, "STACK と Moodle による数学 e ラーニング," *数理解析研究所講究録,* vol. 1735, p. 9, 2011.

[11] 東京大学数理・情報教育研究センター, "Python プログラミング入門," 2022. [Online]. Available: https://sites.google.com/view/ut-python/.

[12] 村山舜, 早川智一 and 疋田輝雄, "6ZC-6 プログラミング学習を支援する Moodle プラグインと UI の提案と実装," in *情報処理学会第 76 回全国大会*, 2014.

[13] 伊藤亜樹, 丸山一貴 and 寺田実, "6ZC-2 初学者のプログラムの自動正誤判定における部分点付与の提案," in *情報処理学会第 76 回全国大会*, 2014.

[14] Google, “Google Classroom,”［オンライン］. Available: https://classroom.google.com/.