NORTHWEST FLORIDA STATE COLLEGE
Department of Mathematics

Partial Credit for Text Fill-in Questions

D. P. Story

June 1, 2009

dpstory@uakron.edu
Version 2.0

1. Introduction

We introduce a new command `\RespBoxTxtPC`, for processing text fill-in questions so that partial credit is awarded for the answer. The idea and the syntax was suggested to me by Kate of Kenya. Documentation is not available yet, we are content with presenting an example.

2. A Quiz

Start **Quiz 1**

1. (2^{pts}) Who was the first President of the United States?

2. (8^{pts}) Write a short paragraph (in complete sentences) on the defeat of Napoleon Bonaparte by the Seventh Coalition. Include the name of the famous battle, the country in which the battle occurred, the year of the battle, and the name of the commander of the Anglo-Allied army.

End Correct

Answers:

3. Syntax

Exerquiz can now create text fill-in questions that awards credit each time one of the key words are found in the student's input string.

The command `\RespBoxTxtPC` is the one that creates a text fill-in question. Its syntax is

`\RespBoxTxt[#1]#2[#3]#4[<num1>]{<word1>}...[<num_n>]{<word_n>}`

The new command `\RespBoxTxtPC` behaves differently than `\RespBoxTxt`. JavaScript performs a regular expression search for each word listed (the word, can be a regular expression), if the word is found (the search is successful), the total credit for this problem is incremented by the amount associated with the word.

Parameters:

`#1:` Optional parameter used to modify the appearance of the text field.

`#2:` This required parameter is a number indicating the filtering method to be used. Permissible values of this parameter are

- `-1:` (The default) The author's and user's answers are not filtered in any way. (Spaces, case, and punctuation are preserved.)
- `0:` The author's and user's answers are converted to lower case, any white space and non-word characters are removed.

`1:` The author's and user's answers are converted to lower case, any white space is removed.

`2:` The author's and user's answers are stripped of white space.

`3:` Same as -1, but a case insensitive search is performed. *This is the recommended value for this function.*

See the JavaScript function `eqFilter` in `exerquiz.dtx` for program code details. Additional filtering options may be added.

Recommendation: For `\RespBoxTxtPC`, use either -1 (case sensitive search) or 3 (case insensitive search).

`#3:` Optional, a named destination to the solution to the question. If this parameter appears, then a solution must follow the question, enclosed in a `solution` environment. If the third parameter is a '`*`', then an automatic naming scheme is used instead.

`#4:` This required parameter is the number of alternative answers that are acceptable. The alternative answers are listed immediately after this parameter.

The parameters that follow `#4` are of the form `[<num>]{<word>}`. The `[<num>]` is the amount of credit the user gets if his answer contains `<word>`. Actually, `<word>` can be a regular expression so that a more

sophisticated search criteria can be set up. This is illustrated in the examples above, and explained below.

A `\RespBoxTxtPC` question is deemed correct (and marked with a green rectangle, in the case of a quiz) if at least one of the words is found.

If a word does not have a partial credit `[<num>]` before it, that word has a partial credit of zero, and may as well not be included in the list of words.

4. Examples and Discussion

In this section, we present examples with extended comments on the code.

Start

1. (2^{pts}) Who flew the airplane at Kitty Hawk, North Carolina, on December 17, 1903?

End Correct

This is the simplest example of partial credit.

```
\RespBoxTxtPC{3}{3}[1]{Orville}[-.5]{Wilbur}[1]{Wright}
```

The correct answer is "Orville Wright." (One point for Orville, and one point for Wright.) Wilbur did not fly the plane that day, so, I've assigned -.5 credit. If the user enters "Wilbur Wright," only .5 points are given.

If the user enters "xOrville xWright," full credit is awarded, this is because the strings "Orville" and "Wright" were found. Within the `\RespBoxTxtPC` command, `\\` is redefined; `\\` takes one argument, and is used to insert escape sequences into the string that have meaning to a regular search. We want to enter `\\b`, which stands for a word boundary.

Start

1. (2^{pts}) Who flew the airplane at Kitty Hawk, North Carolina, on December 17, 1903?

End Correct

This is the simplest example of partial credit.

```
\RespBoxTxtPC{3}{3}
    [1]{\\bOrville\\b}
    [-.5]{\\bWilbur\\b}
    [1]{\\bWright\\b}
```

Now, a response like "xOrville xWrght" awards no points, "OrvilleWright" awards no points as well.

The compare text routine (`compareTxt()`) breaks the author's answer words into an array (breaks a `<word>` by at each space; for example, if `{Orville Wright}` is one of the words, the compare breaks this into an array, `["Orville","Wright"]`, and searches for each of these words. It is a successful search, and any credit is awarded both words are found. For example,

Start

1. (2$^{\text{pts}}$) Who flew the airplane at Kitty Hawk, North Carolina, on December 17, 1903?

End Correct

This is the simplest example of partial credit.

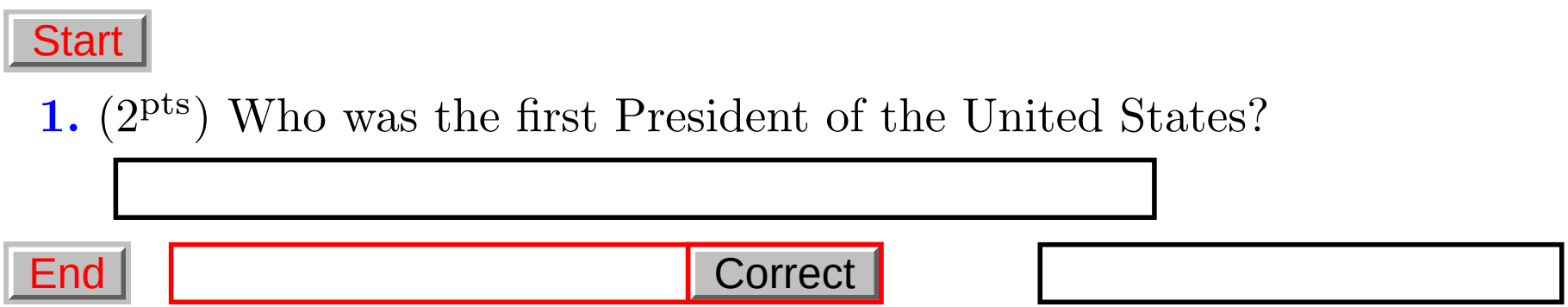
```
\RespBoxTxtPC{3}{2}
    [2]{Orville Wright}
    [1.5]{Wilbur Wright}
```

Here, you should get 2 points for "Orville Wright" and 1.5 points for "Wilbur Wright." These words can appear in any order, for example,

"Wright Orville" is awarded 2 points. (This feature was part of the `\RespBoxTxt` scheme of comparison.)

Important: I don't really recommend having spaces in the `<word>` for `\RespBoxTxtPC`, except in this simple case. Placing spaces in the `<word>` causes problems when more sophisticated searching (such as *alternation*) is used.

Alternation is useful for offering several alternative versions of the same word, including alternate spellings. Let's look at the George Washington question.




```
\RespBoxTxtPC[\rectW{3in}]{3}*{3}
    [0.5]{(\word{President}|\word{Pres.{0,1}}|      % (1)
        \word{General}|\word{Gen.{0,1}})}           % (1)
    [0.5]{(\word{George}|\word{Geo.{0,1}})}         % (2)
    [1.0]{\word{Washington}}                        % (3)
```

Note: I've wrapped the first word around to fit it on the page, this wrapping should not be done in the source file.

I've placed each `<word>` in parentheses, this is called *grouping*. Within each group, I've placed several `|` characters, this is the symbol for *alternation*.

As we've seen in the previous examples on the Wright brothers, in most cases, it is important that some key words *not appear* as a substring of another word; consequently, we mark such as word with `\\b`. As a convenience, `exerquiz` defines the `\word`; the code `\word{AcroTeX}` expands to `\\bAcroTeX\\b`.

Comments on verbatim code:

- In line (1), we offer 0.5 points if the user enters "President", "Pres.", "General", or "Gen.". We use `{0,1}` following the period (.), this means, match 0 or 1 periods (.); consequently, "Pres" and "Pres." are both matches. (This is not really needed here, credit would be given for "Pres" and "Pres." if the matching word was just `Pres`.)
- In line (2), we accept either "George" or "Geo." (with or without the period).
- In line (3), we accept the word "Washington" and award 1 point.

Start Now, we analyze the Napoleon question.

1. (8$^{\text{pts}}$) Who was the first President of the United States?

End Correct

This is the simplest example of partial credit.

```
\RespBoxTxtPC[\Ff{\FfMultiline} % <-- multiline text field
   \rectW{.9\linewidth}\rectH{4\baselineskip}]{3}*{7}
   [1]{(Napoleon|Bonaparte)}
   [.5]{Battle}
   [.5]{defeated}
   [2]{Waterloo}
   [1]{Belgium}
   [1]{1815}
   [2]{((Duke\\s+of\\s+){0,1}Wellington|Arthur\\s+Wellesley)}
```

We award 1 point for either "Napoleon" or "Bonaparte," and various points for various key words that are expected to be in a complete sentence. More credit for the requested information, less credit for other words in the sentence.

When using alternation, do not use spaces in your words. In the earlier examples, this was not a problem, here, we demonstrate how to deal with spaces. In the last word,

```
[2]{((Duke\\s+of\\s+){0,1}Wellington|Arthur\\s+Wellesley)}
```

if a space is expected use `\\s+`, this represents one or more spaces (in a regular expression). Here, we accept "Duke of" followed by "Wellington". I've insert grouping `(Duke\\s+of\\s+)` followed by the repetition character `{0,1}`; altogether

```
(Duke\\s+of\\s+){0,1}
```

matches zero or one occurrences of the phrase "Duke of". with one or more spaces between "Duke" and "of" and one or more spaces following "of". Note, you may want to require "Duke of" to appear before "Wellington", in which case, our word would be

```
Duke\\s+of\\s+Wellington
```

We also accept the Duke's real name "Arthur Wellesley".

Solutions to Quizzes

Quiz 1: Question 1. One response to this question is

> President$^{0.5\text{pt}}$ (or Pres. or General or Gen.) George$^{0.5\text{pt}}$ (or Geo.) Washington$^{1\text{pt}}$

Quiz 1: Question 2. One such paragraph is

> Napoleon[1pt] (or Bonaparte) was defeated[0.5pt] at the Battle[0.5pt] of Waterloo[2pt], Belgium[1pt], in 1815[1pt], by the Duke of Wellington[2pt] (or Arthur Wellesley).