解答まとめ README

解答まとめ作成の方法やフォルダの中身をここにまとめます。運用には bash を利用しています。'$' はコマンドプロンプトです。正直なところ動作確認は作成者の環境でしか行えておりませんので、バグや不具合の報告をお待ちしています。

作成者: @math_Hurdia
第 1 稿: 2020/03/20

1　ディレクトリ構造と内容物のリスト、簡易目次

```
DochaBot_Ans
├── main ..... 2 節参照、pdf を出力するための作業場
│   ├── 00main.(sh, tex, pdf, log, ist, ilg) ..... 2.1 節参照、解答まとめの出力
│   ├── 00NotSolved.list ..... 2.1 節参照、解答未作成問題のリスト
│   ├── 00debug.(sh, pdf, log) ..... 2.2 節参照解答の出力チェック
│   └── *.sty ..... 一般的でなさそうなスタイルファイル群。ほとんど emath からの引用。全部は使っていない。
├── problems ..... 3 節参照、問題の格納庫
│   ├── preamble ..... 3.3 節参照、プリアンブルを格納
│   └── Q_nnn ..... 3.1,3.2 節参照、各解答の tex ファイル置き場
├── org ..... はじめに頂いた tex ファイルのバックアップ
├── ver.20190224 ..... β 版のバックアップ
├── mailbox ..... 解答の草案置き場
│   └── trash ..... 解答まとめに反映が済んだ画像置き場
├── UpdateLog.txt ..... 更新履歴
├── README.v1.pdf ..... このファイル
└── work ..... この README を作成するなどのスクラップ
```

2　解答まとめの出力

この節では、解答まとめを pdf ファイルとして出力する方法を記述します。全ての問題を 1 つのファイルにまとめての出力だけでなく、1 つの問題だけを出力することも可能です。

なお、実行ファイルや tex ファイルの詳細については、別の節に記述します。

ファイル名の先頭に '00' とついているのは、単にソートしたときに先頭にくるための調整です。

2.1　全ての問題と解答を出力

全ての問題の解答と、タイトル・索引などがまとめて 1 つの pdf ファイルに出来上がります。

2.1.1　main ディレクトリに入る

2.1.2　00main.sh を実行する

単に実行するだけでよいです。

```
$ ./00main.sh
```

この実行ファイルは、解答とプリアンブルのバージョン確認を行って、本体である 00main.tex のコンパイルを行います。バージョン確認に少々時間がかかります。正常に処理が終了すると、pdf ファイルが生成されるほか、ログファイルや索引の処理ファイルが出力されます。また、おまけとして、未だ解答が作成されていない問題番号を羅列したテキストファイル '00NotSolved.list' が出力されます。

このとき、00main.tex については編集の必要はありません。

なお、platex, dvipdfmx の結果として得られる dvi, aux ファイルと、索引作成のための一時ファイル (ind, idx) は、pdf

の出力が正常に終了するとこのスクリプトが削除します。

2.2 1 つの問題についてのみ解答を出力

1 つの問題についてのみ、問題と解答を pdf ファイルに出力します。これは 1 つの問題に着目したい場合や、解答を作成した際に pdf の出力を確認するために用います。

2.2.1 main ディレクトリに入る

2.2.2 00debug.sh を実行する

この実行ファイルは出力したい問題番号を引数にとります。

```
$ ./00debug.sh nnn
```

問題番号は必ず 3 桁で入力してください。100 に満たない場合は 0 で埋めてください。007, 023, 189 など。挙動は 00main.sh とほとんど同じです。索引はつきませんが、出力確認用に、中央に縦線が引かれるようになっています。

3 解答と問題の追加

この節では解答を追加する方法と、新しく問題を追加する際の注意事項を記述します。

3.1 解答の追加

既にある問題に、解答を追加する方法を記述します。

3.1.1 problems ディレクトリに入る

3.1.2 Q_nnn ディレクトリに入る

問題番号は常に 3 桁で管理されています。100 に満たない場合は先頭を 0 で埋めてください。このディレクトリの中には、2 種類の tex ファイルが存在します。

1 つは、'Q_nnn.tex' で、これは先の '00main.sh' によって解答まとめのために作成される仮のファイルです。これを編集する必要はありません。

3.1.3 ファイルのバージョンを管理する

もう 1 つは、'Q_nnn.yymmdd.tex' で、こちらが編集するべきファイルです。解答の修正、別解の追加に対応できるように、編集した年月日でバージョン管理されています。年は西暦の下 2 桁、月・日は 2 桁になるように、10 に満たない場合は 0 で埋めます。新しく編集する場合には、古いファイルを直接編集せずに、また削除しないように、コピーしたものを編集します。例えば、2019 年 4 月 1 日に作成されていたファイルを 2020 年 3 月 20 日に編集する場合は、

```
$ cp Q_nnn.190401.tex Q_nnn.200320.tex
```

のようにしてから、新しいバージョンの 'Q_nnn.200320.tex' を編集します。

もちろん bash でなくとも、windows などの GUI からコピーしていただいても構いません。

3.1.4 tex ファイルを編集する

新しく作成したバージョンの tex ファイルを編集し、解答を追加します。問題と解答の間に空行を (tex 内に) あけておくようにしてください。

このとき、まったく新しく解答を追加する場合には、「ここに解答を記述。」という文言がありますので、削除してください。別解を追加する場合には、何か分かりやすく目印をつけるようにお願いします。後述の '
syoumon' コマンドの利用をお勧めします。

3.1.5 図を追加する際の注意事項

解答に添付する図は同じ問題のディレクトリ内に入れるようにお願いします。また、tex ファイル内では、

```
\includegraphics{../problems/Q_nnn/figure.png}
```

のように、参照するファイルのディレクトリ構造に注意してください。これは、tex のコンパイルを main ディレクトリ内で行うように設計したことによる弊害です。

3.2 問題の追加

新しく問題を追加する方法を記述します。

3.2.1 problmes ディレクトリに入る

3.2.2 Q_nnn ディレクトリを作成する

bash なら、

```
$ mkdir Q_nnn
```

でよいです。

3.2.3 tex ファイルの作成

新しく tex ファイルを、'Q_nnn.yymmdd.tex' のようにバージョンを含めて作成します。問題の記述には、ユーザー定義環境 'thm' を利用しています。書式は次のようになっています。

```
\begin{thm}{nnn}{\hosi n}{ 出典 }

\end{thm}
```

thm 環境の詳細は別の節に記述しますが、第一の引数は問題番号、第二の引数は星の数、第三の引数は出典を要求します。第二、第三の引数に適当な内容がない場合 (例えば小問毎に設定するなど)、括弧の中を空にすることが可能です (括弧そのものは省略しないでください)。また、お勧めを意味する二重丸は、'\maru' で出力できます。

小問を設定する場合には、enumerate 環境を利用してください。

問題を記述する環境のあと、一行空けて (tex ファイル内で)「ここに解答を記述」 と記入しておいてください。こうしておくと、00main.sh が 00NotSolved.list に新しい問題を正しく反映できます。

3.2.4 総問題数の更新

解答まとめに全部でいくつの問題が収録されているかを、更新する必要があります。これはループ処理によって tex ファイルやシェルスクリプトが動いているためです。更新が必要なのは、2 箇所です。

1 つめは、'00main.sh' 内の 6,7 行目

```
qnum=239
echo ``total number of problems is ``${qnum}
```

で、変数 'qnum' を更新します。2020 年 3 月 20 日現在 239 問収録されていますので、'qnum=239' となっています。

2 つめは、プリアンブルファイル内の 85,86,87 行目

```
% 問題の総数
\newcounter{qnum}
\setcounter{qnum}{239}
```

で、カウンタ 'qnum' の値を更新します。更新の際には、問題と同様にバージョン管理を行うようにしてください。

3.3 プリアンブルファイルと、ユーザー定義コマンド

プリアンブルファイルは preamble ディレクトリ内に、問題同様のバージョン管理のうえで格納されています。特に編集の必要はありませんが、書式やスタイル、ユーザー定義を変更したい場合には編集する必要があります。この節では、主なユーザー定義したコマンドなどをまとめます。

名称	内容	書式
thm	問題を記述する環境	\begin{thm}{ 問題番号 }{ 星の数 }{ 出典 }
subthm	補題を記述する環境	\begin{subthm}{ 問題番号. 補題番号 }
Rnum	ローマ数字/大文字	\Rnum{n}
rnum	ローマ数字/小文字	\rnum{n}
syoumon	小問見出しの出力	\syoumon{n}
vvv	ベクトル上付矢印の出力	\vvv{hoge}
combi	組み合わせ C の出力	\combi{ 左下 }{ 右下 }
permu	順列 P の出力	\permu{ 左下 }{ 右下 }
disp	数式モードを displaystyle に	\disp
dou	同値矢印の出力	\dou
marunum	丸囲み数字の出力	\marunum{n}
mr	数式モード中で文字を立体に	\mr {hoge}

4 スクリプトや tex ファイルの詳細情報

この節の内容は知らなくても直ちに影響はありませんが、より詳細な編集を行う際に参考にしてください。

4.1 00main.sh について

解答まとめ pdf を作成するためのシェルスクリプトです。実行に必要なファイルは、‘00main.sh’, ‘00main.tex’, ‘00main.ist’, preamble ファイル、各問題ファイル、です。最終的な出力ファイルは ‘00main.pdf’ です。

プリアンブル及び各問題ディレクトリに移動してバージョン確認を行います。具体的な挙動は、

1. 以前の仮ファイル ‘Q_nnn.tex’, ‘preamble.tex’ が存在すれば削除する
2. 各ファイルのバージョン情報 ‘yymmdd’ を抽出し、最も値が大きいもの (=直近のもの) を選択する
3. 最新バージョンを ‘Q_nnn.tex’, ‘preamble.tex’ という名前にコピーする。

となっています。特に問題ファイルについては、このとき同時に中身を確認し、「けつばん」「未完」「ここに解答を記述」と同一の文字列を検出した場合に、00NotSolved.list に問題番号を出力する操作も行います。

バージョン確認が終わると、tex のコンパイルに移ります。このときコマンドは ‘platex’ を利用しています。一度コンパイルすると、続いて索引の作成のためのコマンド ‘mendex’ が実行されます。このためのスタイルファイルは ‘00main.ist’ です。索引を正しく含めたコンパイルとなるように、‘platex’ がもう 1 回実行されます。最後に、pdf への出力のために ‘dvipdfmx’ が実行されます。

‘platex’ 実行時にエラーが出た場合は、対話モードに入るため、スクリプトは一時中断されます。対話モードの中でエラーを解決するよりは、‘x’ として中断することを勧めます。この場合、スクリプトも ‘platex 1st(2nd) ERROR’ と出力して停止します。なるべく、00debug.sh を利用して、tex のコンパイルエラーは全て解決してから 00main.sh を実行することを勧めます。‘mendex’, ‘dvipdfmx’ でエラーを検出した場合も同様の出力をして停止します。もちろんシェルスクリプトなので、ctrl+C で中断することも可能です (やむを得ない場合以外では勧めませんが)。

最後に、一時ファイル.aux, .dvi, .ind, .idx, .ilg を削除します。作成者がよく利用する emacs が (勝手に) 作成するバックアップ仮ファイルも削除します。

4.2 00main.tex について

この tex ファイルは、problmes ディレクトリ内で編集した、preamble や各問題ファイルを結合して取り扱うためのファイルです。

```
\input ファイル名{}
```

によって、別ファイルに書かれた tex 原稿を呼び出すことができることを利用しています。

はじめに preamble を呼び出します。ここでは ‘00main.sh’ によって、バージョン確認がなされたことを前提にしているので、後の問題ファイルも含めて、バージョンは除かれたファイル名で呼び出します。

続いてイントロの内容を記述します。2020 年 3 月 20 日現在、初稿の内容が反映されています。将来的に複数ページにわたっても問題ありません。内容の最後に、次のカラムへ移るように指示があります (\newpage)。

続いて、各問題ファイルを読み込みます。問題番号に ‘q’ というカウンタを当てて、これをループ処理させます。tex のカウンタは自動的には 0 埋めできないので、条件分岐の処理をいれてあります。

最後に、索引を出力します。2 カラムのままでは綺麗に出力されなかったので、一度 ‘\onecolumn’ し、改ページした上で ‘multicols’ 環境の中に索引を出力しています。

4.3 00debug.sh について

ある 1 問についてその問題と解答を pdf に出力するためのスクリプトです。実行に必要なファイルは、対象の問題ファイルとプリアンブルです。出力ファイルは ‘00debug.pdf’ と ‘00debug.tex’ です。

指定の問題について対応するように、問題番号を引数にとります。このとき 0 埋めした値を引数に与えないと、シェルの変数扱いの仕様上エラーが出ます。バージョン確認の挙動は 00main.sh と同様です。preamble と引数に与えられた問題についてのみバージョン確認を行う点と、‘Q_nnn.tex’, ‘preamble.tex’ の仮ファイルは作成しない点で異なります。

バージョン確認が済むと、ヒアドキュメントを利用して ‘00debug.tex’ を生成します。続いて ‘platex’, ‘dvipdfmx’ を用いてコンパイルと pdf の出力を行います。‘platex’ 実行時にエラーが出た場合は、対話モードに入るため、スクリプトは一時中断されます。スクリプト内で対話モードを実行することでどのような挙動となるか不明なため、‘x’ として中断することを勧めます。この場合、スクリプトも ‘platex ERROR’ と出力して停止します。

やはり 00main.sh と同様に、不要な一時ファイルはスクリプトが削除して終了します。

4.4 preamble.tex について

このファイルには、問題・解答作成のために必要な設定がまとめられています。ここを確認すればどのようなスタイルファイルが読み込まれているか・いないかが確認できるほか、ユーザー定義コマンド・環境の詳細も確認できます。

パッケージについては、あまり一般的でないもののみ簡単に記述します。

‘wrapfig’ は図の貼り付けに用いるパッケージで、これを利用することで図の周囲に回り込むように文章を記述できるようになります。ただし、パラメータを適切に設定しないとタイプが崩れがちで、その影響が次の問題にも及ぶことがあり、これは debug の出力では分かりづらいために注意が必要です。

‘itembkbx’ は問題を囲む角の丸い四角形を出力するパッケージです。LaTeX にプリセットされているものとは異なり、ページやカラムをまたいで出力できることが大きなメリットです。一方で、このボックスの中に ‘数式を除いて’1 行しか本文が存在しないと表示が崩れるバグ (?) が存在します。その場合には、問題文の末尾に改行コマンド ‘\\’ を付け足すと解消できます。

そのほか多数の設定が書き込まれていますが、コメントを参照して解読していただければと思います。

索引を出力するための環境調整は、作成者も明るくないので解説不可です。

enumerate 環境は、(1) などが中央そろえになるように調整しています。

図や数式周囲の余白はなるべく小さく、また長い行数の数式は常にページまたぎするような設定です。

問題のためのユーザー定義環境 ‘thm’ は、前の問題の解答からの余白・続く解答への余白の調整、ページのヘッダに問題番号を表示させるための設定、を含んでいます。

補題のためのユーザー定義環境 ‘subthm’ は、左端にグレーのハイライトを付けるようになっています。なお、作成者は引数に ‘問題番号. 補題番号’ をとることを想定しています。例えば、問題 123 の 1 つめの補題は、引数に ‘123.1’ を渡し、出力は ‘**補題 123.1**’ となるように設定しています。

小問の見出しを出力するユーザー定義コマンド ‘syoumon’ は、少し大きめの太字で引数の内容を (**引数**) のように出力し、改行します。そのため、引数に 1 をとれば (1) が、別解とすれば (別解) と出力できます。一方で、書式デザインとしては改良の余地がありそうです。

必要なコマンドや、書式の改良など、手を加えて頂くとよいと思います。変更の際にはバージョン管理をお願いします。あるいは作成者まで依頼していただいても構いません。