2011 年度 前期 電子計算機 1

更新日時 2011-08-07 17:15:04　担当 和地 輝仁

目 次

1　シラバス抜粋

到達目標　表計算ソフトウェアで統計処理ができる。基本的なスクリプトの作成方法を学び、作業の効率化を図れる。文書組版システムを用いて、論理構造をもつ文書の作成ができる。コンピュータを用いた学習教材の作成ができる。

1. 表計算ソフトウェアで統計処理ができる。
2. 基本的なスクリプトの作成方法を学び、作業の効率化を図れる。
3. 文書組版システムを用いて、論理構造をもつ文書の作成ができる。
4. コンピュータを用いた学習教材の作成ができる。

授業計画　順序を交換する場合もあるので注意すること。

1. 表計算ソフトウェアによる統計処理
2. マクロとスクリプト
3. 変数・演算
4. 関数とプロシージャ
5. 制御構造 1
6. 制御構造 2
7. ワークシートの取り扱い
8. 文書組版システム TEX の導入と使用方法
9. 文書の構造
10. 表・箇条書き
11. 数式
12. 相互参照・目次・索引
13. 画像
14. マクロ
15. 期末試験

成績評価　期末試験では、表計算ソフトウェアでのスクリプトの作成と、文書組版システムを用いた文書の作成ができるかどうかを評価する (50%)。各回の授業では演習問題を出題し、その提出状況と内容も評価する (50%)。原則として全ての時間の出席を求めるが、やむを得ない理由で欠席をする (した) 場合はできるだけ速やかに申し出て、指示を受けること。

2　授業のノート (VBA 編)

§1　表計算ソフトウェアによる統計処理

(1.1)　課題 01 – 課題提出の練習

テーマ: 課題の提出方法、俳句に親しむ

以下のことを実行して下さい。

(1) 小手調べに「メモ帳」を起動して、適当な俳句を詠んで入力する。
(2) 「0000 和地-01」のような「学生番号名前-課題番号」というファイル名[1]で保存する。数字は半角。
(3) 「提出用ページ」[2] をウェブブラウザで開き、「課題提出」の所に配布したパスワードを入力し、提出するファイルを選択する。そして「提出」ボタンを押すと提出が完了する。
(4) 「提出用ページ」の「提出状況閲覧」の所に、パスワードと学生番号を入力し、「閲覧」ボタンを押すとこれまでのすべての提出状況が見られる。

また、提出した課題に修正を求められたときなどは、修正後のファイルを 1 回目と同様に提出して下さい。同一ファイル名でも区別して管理されるため、「提出状況閲覧」から提出状況を見ると、2 回提出したこともわかります。

(1.2)　**課題の合否と再提出**　「提出状況閲覧」では、提出したファイルが課題番号ごとに一覧できます。複数回提出した課題では、新しいファイルほど下に表示されます。ファイル名の右に「合格」とあれば、その提出課題は合格です。「再提出」とあれば、修正して再提出して下さい。

(1.3)　課題 02 – 無意味に数を埋める訓練

テーマ: オートフィル、形式を指定した貼り付け、式、条件付き書式、簡単な関数、書式のコピー

[1] テキストファイルの拡張子「.txt」も含めれば、「0000 和地-01.txt」となる
[2] URL は板書します

Excel の基本操作を思い出してみます。次の内容を記載した Excel のファイルを提出して下さい。

(1) A 列の A1 から A100 までのセルすべてに、あなたの誕生月を埋める。埋める方法は問いません。以下も同じです。
(2) B 列の B1 から B100 までのセルに、上から順に $0, 1, 2, \ldots, 99$ を埋める。ただし、あなたの学生番号の下 2 桁が埋まるべき所には、代わりに 100 を埋める。
(3) C 列の C1 から C100 までのセルに、上から順に $0, 2, 4, \ldots$ と偶数を埋める。
(4) D 列の D1 から D100 までのセルに、上から順に $0, 3, 6, \ldots$ と 3 の倍数を埋める。ただし、5 の倍数のときはセルの背景を赤にする。
(5) A1 から D100 の範囲の、和と平均を、E 列の適当な所に記入する。
(6) F 列の F1 から F100 までのセルに、上から順に $0, 1, 2, \ldots, 99$ を埋める。ただし、素数のセルの文字色を青にする。

(1.4)　課題 03 – 九九表

テーマ: 相対参照、絶対参照、罫線

次の内容を記載した Excel のファイルを提出して下さい。

(1) A 列に A2 から下へ $1, 2, \ldots, 9$ を埋める。
(2) 1 行目に B1 から右へ $1, 2, \ldots, 9$ を埋める。
(3) これらを九九表の被乗数と乗数だと思って、B2 から J10 の範囲の九九表を埋める。
(4) A 列や第 1 行の背景を変えたり、罫線のようなものを追加するなどして体裁を整える。

(1.5)　課題 04 – ソート・フィルタ

テーマ: テキストファイルのインポート、オートフィルタ、共有フォルダ

以下の作業結果の Excel ファイルを提出して下さい。

(1) 共有フォルダ[3]に「課題-04」というテキストのファイルがあるので、自分のコンピュータにコピーする。
(2) このテキストファイルの学生番号、得点のデータを、Excel のファイルの A 列、B 列にそれぞれ埋める。
(3) 「フィルタ」を使って、学生番号や得点でソートできるようにする (ここは実演する予定)。
(4) 得点の右の列 (つまり C 列) に以下の要領で成績を埋める。

- 60 点未満は F
- 60 点以上 70 点未満は D
- 70 点以上 80 点未満は C
- 80 点以上 90 点未満は B
- 90 点以上 100 点以下は A

(5) 提出するファイルでは、学生番号の昇順にソートしておくこと。

(1.6) **分析ツールの導入 (課題 05 の準備)** 今使っている Excel には、おそらく「分析ツール」が導入されていないので、以下のように導入します。[4]

(1) Excel のメニューバーから「ツール」–「アドイン...」と選択する。
(2) 「有効なアドイン」というリストが現れるが、その中の「分析ツール」にチェックが付いていたら導入済み。付いていなかったらクリックしてチェックを付け、OK をクリックして導入する。

(1.7) **課題 05 – 度数分布表とヒストグラム**

テーマ: 分析ツール

1981 年から 2010 年までの 30 年間の釧路の降水量の度数分布表とヒストグラムを作成します。以下の作業結果の Excel ファイルを提出して下さい。

(1) 気象庁のトップページ `http://www.jma.go.jp/` から、「気象統計情報」–「過去の気象データ検索」とたどり、「都道府県を選択」から釧路を選び、「年ごとの値」をクリックする。
(2) このページのデータのうち、1981 年から 2010 年までの西暦を Excel のシートの A 列に、その間の降水量の合計の値を B 列に転記する。方法は問わない。
(3) C 列に階級の下限を順に記入する。例えば、一番最初の階級を 800mm 以上 900mm 未満にするならば、`C2` に 800 を、以下 100 ずつ増やしながら必要な所まで C 列を埋める。仮に、`C2` から `C10` に埋めたとして説明を進める。
(4) Excel のメニューから「ツール」–「分析ツール...」と選択し、[5]リストから「ヒストグラム」を選び OK をクリックするとヒストグラムのダイアログが開く。

- 入力範囲には B 列の降水量のデータの範囲を指定
- データ区間には C 列の階級の下限が埋められている範囲を指定
- 出力先をートの適当な位置に指定
- グラフ作成をチェック

そして OK をクリックすると度数分布表とヒストグラムが得られる。

この方法では、降水量データを変更すると、もう一度分析ツールを使用しないと度数分布表とヒストグラムに変更が反映されないので注意が必要です。

§2 マクロとスクリプト

(2.1) **マクロ** Excel での**マクロ**とはユーザが一連の操作を記録したものです。記録したマクロをユーザが実行すると、記録された操作が順に実行されます。**同じ操作の反復にマクロを利用すると作業が効率化できます**。

[3]デスクトップにある「共有フォルダへのショートカット」というアイコンをクリックして、その中の「和地」というフォルダ

[4]本文の記述は Excel 2003 のもの。Excel 2007 では、「Office ボタン (ウィンドウ左上隅の丸いボタン)」–「Excel のオプション」–「アドイン」– プルダウンメニューの「Excel のアドイン」を選んで「設定」–「分析ツール」–「ヒストグラム」をチェックして「OK」

[5]本文の記述は Excel 2003 のもの。Excel 2007 では、「データ」タブの「分析」から「分析ツール」を選ぶ。

しかし、Excel のマクロは非常に高機能なので、Excel のマクロで作られたコンピュータウィルスが存在します。作成者を信用できないマクロは不用意に実行してはいけません。このため、Excel のセキュリティレベルによっては、マクロを含むファイルを開いてもマクロを実行できないよう設定されています。その場合は、ファイルを開いた時に表示されるダイアログのヘルプに従って、マクロのセキュリティレベルを「中」に変更して下さい。

(2.2) 課題 06 – マクロ

テーマ: マクロの記録

簡単なマクロを記録してみます。以下の作業結果の Excel ファイルを提出して下さい。

(1) Excel のシートの適当なセルを選択する。
(2) メニューから「ツール」–「マクロ」–「新しいマクロの記録...」と選択し、現れるマクロの記録のダイアログで、OK をクリックする。ここで小さいウィンドウが出現する[6]はず。以降の操作はすべて記録されるため、不用意なクリックなども意図せず記録されるので注意して下さい。
(3) セルの選択はそのままにして、あなたの学生番号をタイプし、ツールバーの下の入力欄の左のチェックマークをクリックしてタイプを完了する。
(4) ツールバーにある **B**、*I*、U を用いて文字装飾を変更したり、背景色、文字色の変更を好きなだけ行う。
(5) 先ほど出現したウィンドウの四角い停止ボタンを押す。これでマクロの記録が完了する。
(6) セルを選択して、メニューから「ツール」–「マクロ」–「マクロ...」と選択し、今記録したマクロを実行する。自分の学生番号が装飾付きで選択したセルに書かれるはず。
(7) 適当なセル 20 箇所程度に、同様にして装飾付き学生番号を埋める。余裕があればマクロにショートカットキーを設定すれば作業がさらに楽である。

[6]出現しない人は相談して下さい。

(2.3) 課題 07 – スクリプト

テーマ: スクリプトの編集

マクロは Excel の内部では **VBA** (Visual Basic for Applications) というプログラミング言語で記録されています。このような、Java や C のように本格的ではないが、小回りのきくプログラミング言語を (明確な基準はないものの) **スクリプト言語**と呼び、スクリプト言語で書かれたプログラムのことを**スクリプト**と呼びます。

メニューから「ツール」–「マクロ」–「マクロ...」と選択し、記録されているマクロを選んで編集をクリックすると、記録されたマクロのスクリプトを見ることもできます。

さらに、操作を記録することによってだけではなく、直接 VBA スクリプトを作成することもでき、Excel の操作の補助という役割を大きく超えたプログラミングが可能です。

以下の作業結果の Excel ファイルを提出して下さい。

(1) 課題 06 の Excel ファイルをコピーして、ファイル名を課題 07 に合わせる。
(2) 課題 07 のファイルを開くと、セキュリティレベルが中になっていれば、ダイアログに「マクロを有効にする」のボタンがあるのでクリックする。セキュリティレベルが高のままならば、ヘルプボタンを押して、セキュリティレベルの変更方法を読み、セキュリティレベルを中にする。
(3) メニューから「ツール」–「マクロ」–「マクロ...」と選択し、課題 06 で作成したマクロを選んで「編集」をクリックする。
(4) 記録されているスクリプトを読めば、何となくわかるので、学生番号の数値に何か追加してみたり、色などの装飾を変更して保存する。
(5) 今変更したマクロを用いて、数箇所のセルに埋めてみる。

長いマクロの記録中にわずかなタイプミスが起こったとしても、そのまま記録を完了して、スクリプトを修正すればタイプミスは回復できます。つまり、長いマクロを再び最初から記録し直す必要はありません。

§3　変数・演算

(3.1)　課題 08 – 最初のプログラム

テーマ: Visual Basic エディタの利用、モジュール、Sub、MsgBox、プログラムの実行

Excel のマクロは **VBA** (Visual Basic for Applications) というプログラミング言語で記録されていますので、操作をマクロとして記録することで、VBA のプログラムが作成されていることになります。これからは、操作を記録してプログラムを作成するのではなく、1 からプログラムを入力して作成することを学びます。

- VBA のエディタである Visual Basic エディタ (**VBE**) を起動するには、「Alt-F11」を押します。
- (Excel ではなく) VBE のメニューから「挿入」–「標準モジュール」を選択すると、**標準モジュール** が作成されます。この授業で扱う VBA プログラムは標準モジュールに書きますが、ワークシート上のイベントを取得したいときなどは、別のモジュールを利用します。
- プログラムを実行するには、VBE のツールバーの再生ボタンをクリックするか、Excel のメニューから「ツール」–「マクロ」–「マクロ...」と選択し、マクロの実行と同じようにプログラムを実行します。

今後は、プログラムを作成すると言えば、上のようにして作成した標準モジュールにプログラムを書くことを指します。

以下の作業結果の Excel ファイルを提出して下さい。正しく作業できれば、俳句を表示したウィンドウが表示され、ボタンを押すとそれが閉じる、というプログラムが出来ます。

(1) 新規に作成した標準モジュールに、次の 3 行を書く。ただし、俳句は自作すること。

課題 08

```
1 Sub MyFirstScript()
2   MsgBox "雨蛙そこのけそこのけお家へ帰る"
3 End Sub
```

(2) 実行して動作を確認する。

- 「Sub プロシージャ名 () ～ End Sub」は、**Sub プロシージャ**の定義です。プロシージャとは、例えば Sum(B2:B5) や Average(C3:D8) のような Excel の組み込み関数と同じプログラムの 1 実行単位です。
- MsgBox は組み込みの VBA 関数で、上のプログラムでは与えられた文字列をダイアログに表示します。ここで重要なのは Sum 関数の場合と異なり、関数名の後にかっこを置かないことです。これは VBA では基本的ですが、誤った理解や解説を載せたウェブサイトが多くあります。この件に限らず、ネット上の情報は誤りも多いので、VBA のことであれば 1 次情報である Microsoft のサイトやヘルプ以外は鵜飲みにしない位の慎重さが必要です (つまりこの資料も鵜呑みにするのは危険)。
- 文字列は 2 重引用符で囲みます。

(3.2)　課題 09 – 変数・型

テーマ: 変数宣言、型、代入

変数とはデータを格納しておく器のことです。Excel のセルと似ています。変数には格納できるデータの種類を指定した**データ型**があります。主な型を以下にあげます。

データ型	説明
Integer	−32768 から 32767 までの整数値
Long	−2147483648 から 2147483647 までの整数値
Double	浮動小数点数
Date	日付
String	文字列
Variant	任意の型の値を格納できる

この授業では Variant 型の使用は避けます。**変数宣言**は例えば Long 型ならば、`Dim a As Long` とします。

Integer 型の変数 `a` に 1 を**代入**し、Integer 型の変数 `b` に 2 を代入し、Integer 型の変数 `c` に `a+b` の結果を代入し、変数 `c` を MsgBox で表示するプログラムは、

演算と `MsgBox` の例

```
1  Sub MyFirstCalculation()
2    Dim a As Integer
3    Dim b As Integer
4    Dim c As Integer
5    a = 1
6    b = 2
7    c = a+b
8    MsgBox c
9  End Sub
```

となります。これにならって、以下の 3 つのことを続けて行う 1 つの Sub プロシージャ(プロシージャ名は `kadai09`) を作成し、その Excel ファイルを提出して下さい。

(1) Long 型の変数 `d1` に 11 を代入し、Long 型の変数 `d2` に 23 を代入し、Long 型の変数 `d3` に `d1+d2` の結果を代入し、変数 `d3` を MsgBox で表示する。
(2) Double 型の変数 `e1` に 1.1 を代入し、Double 型の変数 `e2` に 2.3 を代入し、Double 型の変数 `e3` に `e1+e2` の結果を代入し、変数 `e3` を MsgBox で表示する。
(3) String 型の変数 `f1` に`"abc"`を代入し、String 型の変数 `f2` に`"xyz"`を代入し、String 型の変数 `f3` に `f1` と `f2` を連結した結果を代入し、変数 `f3` を MsgBox で表示する。ただし、文字列の連結の演算子は`&`である。

(3.3) 課題 10 − 演算

テーマ: 算術演算子

数値の演算に用いる演算子には次のようなものがあります。

記号	意味	例	結果
`+`	和	`a = 9+4`	13
`-`	差	`a = 9-4`	5
`*`	積	`a = 9*4`	36
`/`	商	`a = 9/4`	2.25
`^`	累乗	`a = 9^4`	6561
`¥`	整数の商	`a = 9¥4`	2
`Mod`	整数の余り	`a = 9 Mod 4`	1

自分で、Long の範囲に収まる大きな数 2 つを適当に選んで変数に代入し、この変数を用いて以下のことを続けて行う 1 つのプログラム (Sub プロシージャ `kadai10`) を作成し、その Excel ファイルを提出して下さい。以下では 2 数を仮に 12345 と 56789 としていますが、オーバーフロー (計算結果が Long に収まらないこと) が発生するので、大きすぎないように選ぶ必要はあります。

(1) 「12345 と 56789 の和は 69134 です」と MsgBox で表示する。
(2) 「12345 と 56789 の差は −44444 です」と MsgBox で表示する。
(3) 以下、上の表の残りの演算についても、同様に順に結果を表示する。

文字列結合の演算子`&`は、数値に適用すると、その数値を暗黙的に文字列に変換するので、上の MsgBox で表示する文字列の作成に用いることができます。

(3.4) 課題 11 − セルの値の操作

テーマ: セルの値の取得、セルの値の変更

セル (やセル範囲) の値は次のように取得できます。

セルの値の取得

```
1  ThisWorkBook.ActiveSheet.Cells(1,2)
2  ThisWorkBook.WorkSheets(1).Cells(1,2)
3  ThisWorkBook.WorkSheets("Sheet1").Range("A2")
```

これを、変数に代入したり、数式に用いたりできます。

上のように、セルがどのワークシートにあるかをいちいち指定するのは繁雑ですが、

セルの値の取得の簡略化

```
1 Dim sheet As WorkSheet
2 Set sheet = ThisWorkBook.ActiveSheet
3 sheet.Cells(1,2)
```

のような記述が可能です。WorkSheet 型のような**オブジェクト変数**への代入では、Set が必要です。

アクティブなワークシートのセル A1 と A2 の値の和を A3 に書き込むプログラムは、次のようになります。

セルの操作の例

```
1 Sub MyFirstCellOperation()
2   Dim sheet As WorkSheet
3   Dim a As Integer
4   Dim b As Integer
5   Set sheet = ThisWorkBook.ActiveSheet
6   a = sheet.Cells(1,1)
7   b = sheet.Cells(2,1)
8   sheet.Cells(3,1) = a + b
9 End Sub
```

sheet.Cells(2,1) は sheet の 2 行 1 列、つまり A2 のセルを表します。(2,1) の順が、A2 の順と逆であるので注意が必要です。これにならって、以下のようなプログラムを作成し、その Excel ファイルを提出して下さい。

(1) アクティブなワークシートのセル A1 と A2 の差を A3 に書き込む
(2) セル B1 と B2 の積を B3 に書き込む
(3) セル C1 を C2 で割った整数の商を C3 に、余りを C4 に書き込む

(3.5) With 文

テーマ: With

課題 11 では、オブジェクト変数 sheet を導入して記述の簡略化を図りましたが、With オブジェクト変数 ··· End With を用いると、オブジェクト変数の記述も省略できます。この方法を用いると、アクティブなワークシートのセル A1 と A2 の値の和を A3 に書き込むプログラムは、次のようにも書けます。

With 文

```
1 Sub MyFirstWith()
2   Dim sheet As WorkSheet
3   Set sheet = ThisWorkBook.ActiveSheet
4   With sheet
5     .Cells(3,1) = .Cells(1,1) + .Cell(2,1)
6   End With
7 End Sub
```

変数 a,b の使用も抑制しました。課題 11 での例と、記述の単純さを比べてみて下さい。

§4 関数とプロシージャ

(4.1) 課題 12 – 最初の Function プロシージャ

テーマ: 引数、戻り値、Function

合計を計算する Sum のような Excel の関数は、入力に対して出力を返します。例えば、セル B1 に、=Sum(A1:A10) と数式が書き込まれていたら、関数 Sum の入力はセル範囲 A1:A10、出力はその範囲の合計です。関数への入力を**引数**(ひきすう)、出力を関数の**戻り値**と呼びます。

今まで作ってきたプログラムのような戻り値のないプロシージャを **Sub プロシージャ**、戻り値を戻すプロシージャを **Function プロシージャ**と呼びます。引数を 2 乗して戻す Function プロシージャは次のように定義します。

数の平方を返す Function プロシージャ

```
1 Function HeyHoe(a As Integer) As Integer
2   HeyHoe = a * a
3 End Function
```

a が引数で、プロシージャ名の後のかっこの中に宣言します。戻り値の型を引数リストの後に指定します。戻り値は、プロシージャ名を変数のように見立てて HeyHoe に代入します。また、引数が複数ある場合は、次のようにカンマ区切りで列挙します。

複数の引数をとる Function プロシージャ

```
1 Function Wa(a As Integer, b As Integer) As Integer
2   Wa = a + b
3 End Function
```

ただし、これらのプロシージャはこのままでは実行できません (試してみて下さい)。普通に実行しても引数を指定できないからです。

Function プロシージャの使用例

```
1 Sub HeyHoeWa()
2   Dim a As Integer
3   Dim b As Integer
4   a = HeyHoe(3)
5   b = Wa(4, 5)
6   MsgBox a & b
7 End Sub
```

のように、実行の主体となる (引数のない) Sub プロシージャを作りこれを実行すると、ダイアログに「99」と表示されます。

上では 3 つのプロシージャを作成しましたが、1 つの (例えば Module1 という名前の) 標準モジュールに、プロシージャの定義を 3 つ続けて書きます。複数のモジュールに分けることも出来ますが、それはより大規模なプログラムで、意味でプロシージャたちをグループ分けしたいときや、後述のスコープを限定したいときに行います。

上の例を参考にして、以下の 2 つのプロシージャを (1 つのモジュールに) 作成し、その Excel ファイルを提出して下さい。また、**型の宣言は省略してもプログラムが実行可能な場合が多くありますが、この授業では必ず型宣言を省略せずに書いて下さい**。

(1) 1 つの Integer 型の引数を持ち、それを 3 乗して戻り値とする、Rippo という名前の Function プロシージャを作る。

(2) それを用いて、1 の 3 乗から 10 の 3 乗までを足した結果を MsgBox で表示する Sub プロシージャRippoWa を作る (このプロシージャを実行すると、正しい結果が表示されていることを確認すること)。

もちろんこの課題もプログラムの練習が目的です。答を出すだけなら、1^3 + 2^3 ... と書いた方がよほど早いし、エクセルのシートで計算すればもっと早いです。

(4.2) 課題 13 – スコープ

テーマ: 変数のスコープ

課題 12 の解説の部分での 2 つの Function プロシージャHeyHoe、Wa と Sub プロシージャHeyHoeWa において、引数や変数に同じ a を用いましたが計算に矛盾はありません。それは、プロシージャの引数や、プロシージャの中で宣言された変数は、そのプロシージャの中からしか見えないからです。このことを変数の**変数のスコープ**がプロシージャ内だけであると言います。

以下の 2 つのプロシージャを作成し、その Excel ファイルを提出して下さい。

(1) 2 つの Double 型の引数 w と t を持ち、Double 型の $w \div t^2$ を戻り値とする関数 BMI を作る (体重 (kg)÷(身長 (m))^2 を BMI 指数と言います)。

(2) 次のことを順に行う。Main という名前の Sub プロシージャを作る。

- Double 型の変数 w と t を宣言する。
- w に 65.8、t に 1.72 を代入する (w と t の値は何でもいいです)。
- Function プロシージャBMI を用いて、BMI(52.2, 1.59) を計算し、MsgBox で表示する (これらの値は、先程の w と t と異なる値なら何でもいいです)。
- w と t を MsgBox で表示する。(実行してみて最初の値と同じであることを確認せよ)。

(4.3) 課題 14 – Function プロシージャの利用

テーマ: セルの値の Function プロシージャを用いた変更

以下のことを行った Excel ファイルを提出して下さい。

(1) セル A1 に勝手な体重 (kg), セル B1 に勝手な身長 (m) を書き込む。
(2) 課題 13 で作成した Function プロシージャ`BMI` を、コピーアンドペーストするなどして再利用する。
(3) 以下のことを行う `calcBMI` という名前の Sub プロシージャを、Function プロシージャ`BMI` を利用して作成する。

- セル A1 を体重、B1 を身長とみて、C1 に BMI 指数を書き込む。

§5 制御構造 (条件分岐)

(5.1) 課題 15 – 条件分岐 If (1)

テーマ: If ∼ Then ∼ End If

ある条件を満たすときにのみ何かの処理をしたい場合 **If 文**を用います。`pt` が 60 未満ならば「不合格」とダイアログに表示したいならば次のようになります。

`If` 文の例

```
1 If pt < 60 Then
2   MsgBox "不合格"
3 End If
```

数値の比較演算子は他にも以下のようなものがあります。演算子の式の結果は、条件が満たされるとき True、そうでないとき False となります。

記号	意味	例			結果
`=`	等しい	`9`	`=`	`4`	False
`<>`	等しくない	`9`	`<>`	`4`	True
`<`	左辺が小さい	`9`	`<`	`4`	False
`>`	左辺が大きい	`9`	`>`	`4`	True
`<=`	左辺が小さいか等しい	`9`	`<=`	`4`	False
`>=`	左辺が大きいか等しい	`9`	`>=`	`4`	True

次のことを行う Sub プロシージャ`kadai15` を作り、比較対象になっているセルに適当な値を埋めて実行し結果を確認して下さい。その Excel ファイルを提出して下さい。

(1) セル A1 と A2 を比較し、等しいときに限り A3 に自分の学生番号を書き込む。
(2) セル B1 と B2 を比較し、B1 が大きいとき自分の学生番号を `MsgBox` で表示する。

(5.2) 課題 16 – 条件分岐 If (2)

テーマ: If ∼ Then ∼ Else ∼ End If

ある条件を満たすときに処理 1、満たさないときに処理 2 をしたい場合 If 文の中で **Else** を用います。`pt` が 60 以上ならば「合格」、そうでないなら「不合格」とダイアログに表示したいならば次のようになります。

`Else` の例

```
1 If pt >= 60 Then
2   MsgBox "合格"
3 Else
4   MsgBox "不合格"
5 End If
```

次のことを行う Sub プロシージャ`kadai16` を作り、比較対象になっているセルに適当な値を埋めて実行し結果を確認して下さい。その Excel ファイルを提出して下さい。

(1) セル A1 と A2 を比較し、等しいときは A3 に自分の学生番号を書き込み、そうでないなら自分の学生番号の −1 倍を書き込む。
(2) セル B1 と B2 を比較し、大きい方の値を `MsgBox` で表示する。

(5.3) 課題 17 – 条件分岐 If (3)

テーマ: If ∼ Then ∼ ElseIf ∼ End If

条件 1 を満たすときに処理 1 をし、条件 1 は満たさないが条件 2 を満たす場合は処理 2 をし…、というように、複数の場合分けをしたいときは、If 文の中で **ElseIf** を用います。pt が 90 以上ならば「A」、60 以上 90 未満ならば「合格」、そうでないなら「不合格」と MsgBox に表示したいならば次のようになります。

ElseIf の例

```
1 If pt >= 90 Then
2   MsgBox "A"
3 ElseIf pt >= 60 Then '90以上の時ここへ到達しない
4   MsgBox "合格"
5 Else
6   MsgBox "不合格"
7 End If
```

ElseIf はいくつでも連ねることもできます。Else はなくても構いません。

次のことを行う Sub プロシージャkadai17 を作り、比較対象になっているセルに適当な値を埋めて実行し結果を確認して下さい。その Excel ファイルを提出して下さい。

(1) セル A1 の値によって、次のような表示を MsgBox で行う。

- 60 以上なら「合格」
- 45 未満なら「不合格」
- それ以外なら「F*」

(2) セル B1 の値によって、次のような処理を行う。

- 90 以上ならばセル B2 に文字列「A」を埋める。
- 60 以上 (60 以上 90 未満という意味**ではない**) ならばセル B3 に文字列「合格」を埋める。
- 60 未満ならばセル B3 に文字列「不合格」を埋める。

(5.4) 課題 18 – 論理演算子

テーマ: And, Or, Not

複雑な条件を表すのに次のような論理演算子が使えます。

記号	意味	例	結果
And	かつ	`1 = 1 And 2 > 1`	True
Or	または	`1 <> 1 Or 2 < 1`	False
Not	否定	`Not (1 > 3 Or 2 <> 2)`	True

次のことを行う Sub プロシージャkadai18 を作り、比較対象になっているセルに適当な値を埋めて実行し結果を確認して下さい。その Excel ファイルを提出して下さい。

(1) セル A1 が A2 より小さく、セル B1 も B2 より小さいとき「どちらも小さい」と MsgBox で表示する。
(2) セル A1 が A2 より大きく、セル B1 も B2 より大きいとき「どちらも大きい」と MsgBox で表示する。

§6 制御構造 (繰り返し)

(6.1) 課題 19 – 繰り返し For

テーマ: For ∼ Next

同じような処理を繰り返す場合は **For 文**を用います。同じ俳句を MsgBox で 18 回繰り返して表示したいときは、MsgBox "俳句"を 18 回続けて書いてもよいですが、For 文を用いて次のように書けます。

For の例

```
1 Sub Haiku18Ban()
2   For i = 1 To 18
3     MsgBox "茨豌豆むく背をひとり思い見る"
4   Next i
5 End Sub
```

i を**カウンタ変数**と呼び、この変数が**開始値** 1 から**終了値** 18 まで変わりながら、For と Next の間が繰り返されます。したがって、開始値と終了値も変更

した次のようなプログラムだと、MsgBox が 8 回表示されますが、最初は「11 回目」、次は「12 回目」と順に表示され、最後は「15 回目」と表示されます。

開始が 1 ではない For の例

```
1 For j = 11 To 15
2   MsgBox j & "回目"
3 Next j
```

次のことを行う Sub プロシージャkadai19 を作り、その Excel ファイルを提出して下さい。

(1) MsgBox を次のように合計 10 回表示する。

- 1 回目: 「1 の 4 乗は 1 です」
- 2 回目: 「2 の 4 乗は 16 です」
- 以下、10 回目: 「10 の 4 乗は 10000 です」まで同様

(6.2) 課題 20 − For と Step

テーマ: For ∼ Step ∼ Next

For 文のカウンタ変数の増分を 1 以外にしたい場合は、次のように **Step** を使えます。

Step の例

```
1 Dim a As Integer
2 a = 0
3 For i = 1 To 10 Step 2
4   a = a + i^3 'aとi^3の和を、改めてaに代入する
5 Next i
6 MsgBox a
```

このプログラム断片は、変数 a に $1^3 + 3^3 + 5^3 + 7^3 + 9^3$ の計算結果を代入します。数式で書けば、$\sum_{k=1}^{5}(2k-1)^3$ なので、次のように Step を使わない For 文でも書けます。

Step を使わない例

```
1 For k = 1 To 5
2   a = a + (2*k-1)^3
3 Next k
```

次のことを行う Sub プロシージャkadai20 を作り、その Excel ファイルを提出して下さい。

(1) 100 から 200 までの整数のうち 3 の倍数の 2 乗の和を MsgBox で表示する。

(6.3) 課題 21 − For 文によるより複雑な処理

テーマ: より複雑な繰り返し

For 文のカウンタ変数は自由に式に用いることができます。

複雑な繰り返しの例

```
1 Dim sheet As WorkSheet
2 Set sheet = ThisWorkBook.ActiveSheet
3 For i = 1 To 10
4   sheet.Cells(i,2) = sheet.Cells(i,1) + 1
5 Next i
```

上のようにすると、セル A1 に 1 加えた値をセル B1 に設定し、以下同様にセル A10 に 1 加えた値をセル B10 に設定するまで行います。

For 文を用いて、次のことを行う Sub プロシージャkadai21 を作り、その Excel ファイルを提出して下さい。セル A1 から A10 までに、0 点から 100 点を無作為に設定して、実際にプログラムを実行してみたものを提出して下さい。

(1) セル A1 が 60 以上ならばセル B1 に「合格」、そうでないなら「不合格」と設定する
(2) 以下同様にして、セル A10 が 60 以上ならばセル B10 に「合格」、そうでないなら「不合格」と設定する、まで行う。

(6.4)　課題 22 – For 文によるさらに複雑な処理

テーマ: For 文中での関数の利用

課題 22.txt というテキストファイル (共有フォルダにあります) の架空の体重、身長データをワークシートの A 列と B 列に埋め、次のことを行う Sub プロシージャ`kadai22` を作り、その Excel ファイルを提出して下さい。実際にプログラムを実行してみたものを提出して下さい。また、課題 13 で作成した Function プロシージャ`BMI` をコピーアンドペーストするなどしてそのまま使うこと。

(1) セル A1 を体重、B1 を身長とみて、C1 に BMI 指数を書き込み、以下順に 100 行目まで C 列に BMI 指数を書き込む。
(2) 各行に対して、C 列の BMI 指数が 25 以上ならば「肥満」、そうでないなら「肥満ではない」と D 列に書き込む。

(6.5)　課題 23 – For と Exit (提出は義務とはしない)

テーマ: Exit 文

For 文の繰り返しがすべて終わる前に For 文を終了したいときは、**Exit 文**を用います。

`Exit` **の例**

```
1 For i = 1 To 5
2   MsgBox i & "前"
3   If i = 3 Then
4     Exit For
5   End If
6   MsgBox i & "後"
7 Next i
```

上のプログラムでは、カウンタ変数が 3 になると、If 文の条件が成立して`Exit For` が実行されて For 文を抜けます。したがって、`MsgBox` で表示されるのは、「1 前」、「1 後」、「2 前」、「2 後」、「3 前」の 5 回です。

A 列のセル A1 から A20 までに 0 から 100 までの整数を無作為に書き込み、それから、次のことを行う Sub プロシージャ`kadai23` を作り、その Excel ファイルを提出して下さい。

(1) セル A1 を 2 乗してセル B1 に書き込み、以下の行でも同様に A 列の値を 2 乗して B 列に書き込んでいく。
(2) ただし、途中で 0 点のセルがあったら、B 列には書き込まず、そこで処理を中断する。

(6.6)　課題 24 – 2 重ループ (提出は義務とはしない)

テーマ: 2 重ループ

For 文は**入れ子**にできます。

(1)

```
1 For i = 1 To 3
2   For j = 1 To 4
3     MsgBox i & "," & j
4   Next j
5 Next i
```

全体としてはカウンタ変数が `i` である For 文ですが、繰り返される処理 (内側の 3 行) は、カウンタ変数が `j` である For 文です。`For` と `Next` で入れ子状にカウンタ変数の対応がついていることにも注意して下さい。カウンタ変数が `j` である For 文を、「内側 3 行」と言うことにすると、上のプログラムの実行過程は次のようになります。

(1) `i=1` のとき内側 3 行を実行する
(2) `i=2` のとき内側 3 行を実行する
(3) `i=3` のとき内側 3 行を実行する

「内側 3 行」は `j` の値が 1 から 4 までの繰り返しなので、結局、`MsgBox` で表示されるのは次の順番となります。

(1) `i=1` のとき 「1,1」、「1,2」、「1,3」、「1,4」

(2) i=2 のとき 「2,1」、「2,2」、「2,3」、「2,4」
(3) i=3 のとき 「3,1」、「3,2」、「3,3」、「3,4」

次のことを行う Sub プロシージャkadai24 を作り、その Excel ファイルを提出して下さい。

(1) ワークシートの 1 行 1 列から 20 行 20 列の範囲について、i 行 j 列には $i \times j$ を書き込み、九九ならぬ二十二十の表を作る。

(6.7) 課題 25 – 回数が不定の繰り返し (提出は義務とはしない)

テーマ: Do ∼ Loop

For 文はあらかじめ決まった回数を、カウンタ変数が変化しながら繰り返しを行いました。あらかじめ回数が決まってはいないけれど、ある条件が満たされたときに繰り返しを終了するためには、**Do 文**を用います。例えば次のプログラムでは、i=1 から開始して、i の 3 乗が 100 未満ならば繰り返し、この条件が満たされなくなった時に繰り返しを終了します。したがって、繰り返しが終了したときに、i は 5 です。

Do の例

```
1 i = 1
2 Do While i^3 < 100
3   i = i + 2
4 Loop
```

Do 文を用いて、次のことを行う Sub プロシージャkadai25 を作り、その Excel ファイルを提出して下さい。まず、準備として A 列の 1 行目から大体 20 行目くらいまで、途中に空きがないように、勝手な数値をセルに書き込んでおいて下さい。

(1) A 列の数値の 2 乗を隣の B 列のセルに書き込む。

sheet.Cells(i,1) <> "" のように空文字列""との比較をすると、セルが空かどうか判定できます。

(6.8) Do 文

Do 文には 4 種類あります。

(1)
```
Do While 条件式
  処理
Loop
```

(2)
```
Do Until 条件式
  処理
Loop
```

(3)
```
Do
  処理
Loop While 条件式
```

(4)
```
Do
  処理
Loop Until 条件式
```

(1) はループ先頭で条件式を計算し、条件が成立する場合のみ繰り返します。
(2) はループ先頭で条件式を計算し、条件が不成立の場合のみ繰り返します。
(3) と (4) はそれぞれ (1) と (2) に対応しますが、条件のチェックがループの末尾になります。したがって、(1) と (2) ではループ内部の処理は、1 回も実行されない可能性がありますが、(3) と (4) ではループ内部の処理は、必ず 1 回は実行されます。

§7 ここまでの Q&A

Excel におけるプログラミングが、まだなんとなくわからないという人向け。

プログラムとは何ですか
コンピュータへの一連の指示です。VBA ではいくつかのプロシージャがプログラムを構成します。

プロシージャとは何ですか
意味のある処理をまとめたものです。同じような処理を何度もする場合、プロシージャにして呼び出すと記述が簡潔になります。

プロシージャはどう呼び出すのですか
Function プロシージャは、a = BMI(65, 1.70) のようにかっこの中に引数を列挙して呼びます。Sub プロシージャは、MsgBox a のようにかっこなしで引数を列挙します。

Sub プロシージャと Function プロシージャの違いは何ですか
戻り値を戻すのが Function プロシージャ、戻さないのが Sub プロシー

ジャです。Function プロシージャは、`a = BMI(65, 1.70)` のように変数に代入するなどして戻り値を利用しますが、Sub プロシージャは利用すべき値を持たないので、`MsgBox a` のように、値は利用しません。

プログラムを実行するとは何をすることですか

ある、引数のない Sub プロシージャを実行することです。1 つのプログラムが複数のプロシージャを持つ場合も、主と従があります。主プロシージャの内部で、いくつかの従プロシージャを呼び出しているはずです。この場合、主の Sub プロシージャを実行することがプログラムを実行することです。

主と従の関係にないのであれば、それは 1 つではなく、複数のプログラムを書いたとも言えます。

変数とは何ですか

値を入れておく器のようなものです。

代入とは何ですか

「変数 = 式」という文で、変数へ式の値を代入します。変数は値を入れる器ですが、その値を書き換えることを意味します。

「`a = a + 1`」は数学的にありえないですがどういう意味ですか

「変数 = 式」は変数へ式の値を代入します。本当は「変数 <- 式」などと書く方が本来の意味と合っていますが、多くの言語ではイコールを用います。上の代入文では右辺の式の値 `a+1` を計算して、それを、左辺の変数 `a` に代入します。結果として `a` を 1 増加させます。

型とは何ですか

値には、整数、文字列、日付などの種類があり、この種類のことを**型**と呼びます。変数の型は、格納する値の型を表し、プロシージャの引数の型は、呼び出すときの引数の型を指定します。Function プロシージャの戻り値も型が指定されます。

Function プロシージャの戻り値は必ず変数に代入すべきですか

いいえ。しかし、同じ値を何度も使うときや、変数に代入した方がその後の式がかえって簡潔になる場合は、進んで代入すべきです。

変数は 1 度使ったらその後は使えなくなりますか

プロシージャの中で宣言した変数はそのプロシージャ内でずっと有効です。したがって、次の処理をするために新たに変数を宣言しなくても、既にある変数を使い回すこともできます。しかし、まったく意味の違う目的に同じ変数を使うのは、例えば変数の名が体を表さなくなるなど弊害もありますので避けるべきです。

プログラムを実行するとオーバーフローと言われましたが何のことですか

計算結果が、`Integer` 型や `Long` 型の値の範囲を超えることをオーバーフローと呼びます。より広い範囲の値を格納できる型を用いたり、計算結果がその型の範囲に収まるようにすると、このエラーは除去できます。

プログラムが停止してデバッガというボタンが出ましたが何のことですか

デバッガとはエラーを修正するためのツールです。便利ですがこの講義では使用法を学べません。単にプログラムを終了させてよいです。

「ツール」-「マクロ」-「マクロ...」のマクロ一覧が繁雑なのはなぜ

不要なマクロがあると一覧に多く現れます。内容を確認してから削除しましょう。また、マクロ名に `Sheet1` のようなものが付加されている場合は、プログラムを標準モジュール以外に保存しています。

なぜプログラムを編集しようとすると多くのウィンドウが現れるのですか

プロシージャごとに新しいモジュールを作ったからです。複数のプロシージャを 1 つのモジュールに書いて下さい。

型を書かなくてもプログラムが動きましたが型はなぜ書くのですか

ミスを防ぐためです。式をタイプミスしたときなど、型の指定があれば誤りを指摘してくれる可能性があります。また、プログラムの書き手としても、その変数が何を格納しているかが明瞭になり、効率が上がる可能性があります。ただ、文法上型指定できない言語も多くあり、問題なく使われているので、これは宗教論争になる話題でもあります。

§8 オブジェクトの取り扱い

(8.1) オブジェクト

テーマ: いろいろなオブジェクト

Dim sheet As Worksheet と宣言するワークシートのオブジェクト変数は既出のオブジェクト変数ですが、他にもいろいろあります。sheet.Cells(1,2) で得られる値の型は今までは気にしていませんでしたが、この値は Range 型のオブジェクトです。オブジェクトと Long などの型との主な違いは、次の 2 点です。

- オブジェクトは Excel のシートなどに実体があるが Long 型などにはない
- オブジェクトは**メソッド**と**プロパティ**を持つ

メソッドはオブジェクトを操作するもので、プロパティはオブジェクトの情報 (属性) を表すものです。

- オブジェクト . メソッド
- オブジェクト . プロパティ

のようにピリオドでつないで用います。また、メソッドやプロパティによっては、.Cells プロパティのように引数を必要とする場合もあります。

例として、以下に、Range オブジェクトの持つメソッドとプロパティを抜粋します。

Range オブジェクトのメソッド抜粋

Clear	オブジェクト全体をクリアする
ClearContents	指定した範囲から数式と文字を削除する
FillDown	指定した範囲の上端のセルを下方向に複写する
Sort	指定した範囲内の値を並べ替える

Range オブジェクトのプロパティ抜粋

Count	指定した範囲内の要素の数を返す
Column	指定した範囲の左端の列番号を返す
Interior	指定した範囲の塗り潰し属性を返す
Row	指定した範囲の上端の行番号を返す

例えば、sheet.Cells(1,2).Clear とすれば、B1 のセル (と色などの書式) が削除され、sheet.Cells(1,2).Count とすれば、セルに値があるかないかに応じて 1 または 0 が値として返ります (Range オブジェクトが複数のセル範囲であるときに、Conut プロパティはより役に立つでしょう)。さらに、sheet.Cells(1,2).Interior.Color = vbRed のように代入できるプロパティもあり、この場合だとセルの背景色が黒になります。

Excel には多数のオブジェクトがあり、それらは多数のメソッドとプロパティを持っています。また、vbRed のような組み込みの定数も多くあります。詳細は参考書や、Excel 内蔵のヘルプを参照して下さい。

(8.2) 課題 26 – オブジェクトのプロパティの利用 (提出は義務とはしない)

テーマ: Selection, Interior, Color, Column, Row の使用

Window オブジェクトには、Selection プロパティがあり、これは、選択範囲を表す Range オブジェクトを返します。本来ならば、window.Selection のようにピリオドでつないで使用するべきですが、この場合は、Selection と省略しても同じ意味になるよう定義されています。従って、例えば、Selection.ClearContents とすると、選択範囲の値が消去されます (書式は消去されない)。

また、既に使い慣れている WorkSheet オブジェクトの Cells プロパティも、単に Cells(1,2) と書くと、実は、ThisWorkbook.ActiveSheet.Cells(1,2) の意味になるよう定義されています。

次のことを行う Sub プロシージャkadai26 を作り、その Excel ファイルを提出して下さい。

(1) 選択されたセルと、その右と下と右下のセル、合計 4 つのセルの背景を赤にする。

(8.3) 課題 27 – ボタン (提出は義務とはしない)

テーマ: ボタンを押すとプログラムを実行する

Excel のメニューから「表示」–「ツールバー」–「フォーム」を選択すると、小さいウィンドウが現れます。そこで「ボタン」をクリックし、配置したい場

所でシート上をクリックすると、ボタンが作成され、「マクロの登録」というウィンドウが現れます。ここで、実行させたいマクロ (Sub プロシージャ) を選択すれば、ボタンを押すとそのマクロが実行されます。ボタンを右クリックすると名前が変更できます。

次の Sub プロシージャ`kadai27` とボタンを作り、その Excel ファイルを提出して下さい。

(1) `kadai27` は、実行すると `MsgBox` に俳句を表示する。
(2) ボタンを押すと `kadai27` を実行する。

(8.4) 課題 28 – InputBox

テーマ: InputBox メッセージ, ウィンドウのタイトル, 初期値

組み込みの `InputBox` プロシージャを用いると対話的な入力が可能になります。`a = InputBox("季語を入力して下さい", "季語の入力", "蝉丸忌")` とすると、「季語の入力」というタイトルのダイアログが現れて、入力を促されます。入力欄には初期値として「蝉丸忌」が設定されており、入力結果は、`InputBox` プロシージャの戻り値として `a` に代入されます。

次のことを行う Sub プロシージャ`kadai28` を作り、その Excel ファイルを提出して下さい。

(1) `InputBox` で整数を入力させる。
(2) `MsgBox` でその整数の 2 乗を表示する。

§9 自由課題

以下はすべて提出を義務とはしない課題です。余裕のある人は提出して下さい。

(9.1) 課題 29 – 文字列操作 (提出は義務とはしない)

テーマ: Mid、Left、Right

`Mid(文字列, beg, len)` は文字列の `beg` 番目から `len` 文字を返し、`Left(文字列, len)` と `Right(文字列, len)` は、それぞれ、文字列の左と右から `len` 文字を返します。

課題 `29.txt` に、`tp0123mab` のような英字 2 文字 + 数字 4 文字 + 英字 3 文字の形式の架空の学籍番号が 100 件あるので、まずそれをシートの A 列にコピーして下さい。それから、次のことを行う Sub プロシージャ`kadai29` を作り、その Excel ファイルを提出して下さい。

(1) A 列にある各学籍番号について、英字 2 文字、数字 4 文字、英字 3 文字に分けて、それぞれ B, C, D 列に書き込む。

(9.2) 課題 30 – フィボナッチ数 (提出は義務とはしない)

$$F_1 = F_2 = 1, \qquad F_{n+2} = F_{n+1} + F_n \quad (n = 1, 2, \ldots)$$

で定まる数列をフィボナッチ数列と言う。次のことを行う Sub プロシージャ`kadai30` を作り、その Excel ファイルを提出して下さい。

(1) `InputBox` で数値 n を入力すると、F_n を `MsgBox` に表示する

(9.3) **課題 31 – 互除法 (提出は義務とはしない)** 次のことを行う Sub プロシージャ`kadai31` を作り、その Excel ファイルを提出して下さい。

(1) セル A1 と A2 の最大公約数を、ユークリッドの互除法で求め、A3 に書き込む

3　授業のノート (TEX 編)

§1　文書組版システム TEX の導入

TEX(テフ, テック) とは「組版(くみはん)ソフトウェア」です。組版とは活字を組んで版を作ることなので、これは文書を作成するソフトウェアを意味しています。以下に TEX の大事な特徴をあげます。

- フリーソフトウェアであり、かつ、あらゆる OS で動作する。
- 数式が美しい。
- 文書の論理構造を明確にでき、相互参照も強力。

これらほど大事ではない特徴も脚注[7] や傍注 にあげます。

- 高精度
- 目次が作れる
- 索引が作れる
- 1980 年頃に誕生
- 自動で hyphenation
- office ではなく office になる

大学のコンピュータ室では、TEX が使えるように設定されていますが、自宅のコンピュータでも TEX を利用したい場合、以下のような方法があります。

(1) TEX の書籍の付録 CD-ROM などからインストールする。
(2) Linux の場合 apt-get (synaptic) などを利用した簡単なインストール方法が用意されている。始めからインストールされていることも多い。
(3) 阿部紀行氏の Windows 用 TeX インストーラを利用する。
`http://www.math.sci.hokudai.ac.jp/~abenori/`
(4) 角藤亮氏のサイト
`http://www.fsci.fuk.kindai.ac.jp/kakuto/win32-ptex/` の Windows 用簡易インストーラを利用する。
(5) Macintosh の場合、桐木紳氏のサイト
`http://math.kyokyo-u.ac.jp/~kiriki/ptex/` や、小川弘和氏のサイト `http://www2.kumagaku.ac.jp/teacher/herogw/` から入手する。

Windows の場合、特に阿部氏のインストーラがおすすめです。

[7] ● どの OS や、どのコンピュータでも出力が同一。
● TEX 方式の数式入力方法を採用するソフトウェアが多い。
● TEX で書かれた本も多い。
● WYSIWYG (what you see is what you get. 画面のままの印刷が得られる) ではなく「バッチ式」であり、この点は HTML に近い。

TEX には、拡張パッケージの追加の仕方でいろいろな亜種がありますが、以後、最も広く使われている LATEX2ε (ラテフツーイー、ラテックツーイー) を解説します。大学のコンピュータ室では LATEX2ε が利用できるようになっており、前述のインストーラによっても LATEX2ε がインストールされます。

§2　TEX の使用方法

TEX は Microsoft Word などとは異なり、入力から印刷までを単一のソフトウェアで行うわけではありません。

「メモ帳」などで tex ファイルを作成
`foo.tex`
↓　LATEX を実行し dvi ファイルに変換 (**コンパイル**)
`foo.dvi`
↓　dvout や dvipdfmx を実行
画面に表示、印刷、`foo.pdf`

しかし、これらを統合するソフトウェアがありますので、複数のプログラムが動いていることは意識せずとも TEX を利用できるようにはなっています。

(2.1)　**課題 51 – 最初の TEX 文書**　LATEX2ε の書式に従った最小の tex ファイルは次のようになります。バックスラッシュ (\) はキーボードの円記号 (¥) のキーを押すと入力でき、コンピュータの環境によっては画面上でも円記号です。

0000 **和地**-51.tex

```
1  \documentclass{jarticle}
2  \begin{document}
3  夏空と駆ける先には蝸牛
4  \end{document}
```

`\documentclass{jarticle}` は、「jarticle」という文書クラスで文書を作成するという命令です。用途に応じて次のような文書クラスがあります。

用途	欧文	和文横書き	和文縦書き	新和文横書き
論文・レポート	article	jarticle	tarticle	jsarticle
長い報告書	report	jreport	treport	(ない)
本	book	jbook	tbook	jsbook

また、`\documentclass[a4j,12pt]{jarticle}` のように、**オプション**を 1 つ、またはカンマ区切りで複数個与えることができます。[8]

`\begin{document}` ～ `\end{document}` は **document 環境**で、この中に書かれたものが出力されます。

次の作業の結果得られた dvi ファイルを提出して下さい。

(1) 上の例の俳句を他のものに替えて `0000和地-51.tex` のようなファイル名で保存する。
(2) コンパイルして dvi ファイルを作成する (これを提出)。
(3) dvi ファイルがきちんと作成できているか、ビューアで見て確認すること。

(2.2) **コメント** tex ファイル中で、パーセント記号 (%) から行末までは無視されます (改行も無視)。これは**コメント**と呼ばれ、出力はしたくないけれど、tex ファイル中に残しておきたいことなどを書けます。

入力

```
1 \documentclass{jarticle}
2 \begin{document}
3 夏空と駆ける先には蝸牛 % 字余り
4 \end{document}
```

出力

夏空と駆ける先には蝸牛

[8] 例えば次のようなオプションが利用可能です。`a4j`, `b5j`: 用紙サイズの指定。無指定の時はやや余白の広い a4paper を指定したことになる。`landscape`: 用紙を横置きで使う指定。`10pt`, `11pt`, `12pt`: フォントのサイズの指定。`twocolumn`: 2 段組にする指定。

(2.3) **改行や空白文字の扱い** Microsoft Word では、改行したりスペースを入力すれば、その通りの画面表示と印刷結果になります。しかし、TEX での扱いはこれとは異なり次のようになります。

- 行末が和文文字 (全角) の時、改行は無視される。
- 行末が欧文文字 (半角) の時、改行は空白 (半角スペース) と解釈される。
- 連続する空白 (半角スペース) は 1 つの空白と同等。
- 行頭や行末の空白 (半角スペース) は何個あっても無視される。
- 空白だけの行 (空行) は、段落の切れ目になる。

このように、半角スペースは良い組版をするため特別の扱いを受けますので、全角スペースで空白を空けることは必要でなければしない方がよいです。

入力

```
1 \documentclass{jarticle}
2 \begin{document}
3 tex ファイルを書くときは、
4     適宜改行を入れて見易く習慣になっています。
5
6 She wore              a yellow ribbon.
7 \end{document}
```

出力

tex ファイルを書くときは、適宜改行を入れて見易く整えるのが習慣になっています。
She wore a yellow ribbon.

(2.4) **初めての命令** `\documentclass{jarticle}`のように\ (バックスラッシュ。環境によっては ¥) の付いたものが TEX の命令です。例えば、`\TeX` という命令は、TEX を出力します。末尾が英字や和字である命令の後の空白については、次の例のように注意が必要です。

入力

```
\documentclass{jarticle}
\begin{document}
\TeX is joyful. % 命令の後の空白は無視される
\TeX{} is joyful. % こうすればよい
{\TeX} is joyful. % あるいはこう
\end{document}
```

出力

TEXis joyful. TEX is joyful. TEX is joyful.

反対に、TEX の直後に空白なしに続けたいときに、

\TeX の楽しみ % エラー

と入力すると「\TeX の楽しみ」までで 1 つの命令とみなされてエラーになります。

(2.5) **特殊文字** パーセント記号を出力したいとき、単に tex ファイルに書いてもコメントの始まりとみなされて出力されません。バックスラッシュ (円記号) なども同様で、このような**特殊文字**には以下ものがあります。

```
# $ % & _ { } \ ^ ~ < > |
```

これらのうちいくつかは、バックスラッシュを付加すると次のように出力できます。

入力

```
\documentclass{jarticle}
\begin{document}
\# \$ \% \& \_ \{ \}
\end{document}
```

出力

$ % & _ { }

また別の方法として、入力をそのまま出力する\verb 命令を用いることもできます。\verb|foo|のように、任意に選んだ同じ文字 (ここでは |) で挟まれた部分をそのまま出力します。

入力

```
\documentclass{jarticle}
\begin{document}
\verb@\@ \verb|$29 > $28|
\end{document}
```

出力

\ $29 > $28

また、< > | は後述の数式モードでは出力できます。

(2.6) **課題 52 – 特殊文字の出力** 次のような出力を与える tex ファイル (ファイル名は 0000 和地-52.tex のように) を作り提出して下さい。ただし、わずかな (無害の) スペースの有無などにより、作成したファイルの出力結果と下の出力例を比べたときに、TEX が自動的に決定する行の折り返し位置が異なる場合がありますが、それは構いません。逆に、改行位置を下の出力例に合わせるために、無理な操作をするのは良くありません。全角スペースの利用は特に有害です。

出力

TEX $G%$r=PNO9k$?$aKO\%$HF~NO$7$^$9!#\verb L?Na$r;H$C$?>19g$N=PNO$O%$G9,!"=gBN$,0[$Jj^$9!#

上の出力では、1 つ目の%と他の%は、tex ファイルでの記述が異なるため、書体が違うことにも注意して下さい。最初の TEX は、単に TeX と入力するわけではありません (少し前を見て下さい)。また、上の出力では掲載を省略していますが、用紙の下部にページ番号も出力されています。

§3　文書の構造

文書には、章があり、その中にいくつかの節 (セクション) があり、さらに、小節 (サブセクション) や段落を含むといった**論理構造**があります。他方、章の見出しは文字サイズを大きくし、節の見出しはそれより少し小さくし、また、これらの見出しは太字にするといった**物理構造 (レイアウト)** もあります。TEX では、論理構造の指定のみ行い、物理構造の指定は陽には行わないことが推奨されます。つまり、「文字サイズを大きくする」と指定するのではなく、「ここから節を始める」と指定します。どのような物理構造になるかは jarticle などの文書クラスによって決まります。

(3.1)　**論理構造を指定する命令**　jarticle クラスで論理構造を指定するには、主に次のような命令があります。ただし、見出しの必要がない段落は、\paragraph 命令を用いずとも、単に空行を作れば段落の区切りになります。

命令	意味	使用方法
\section	節	\section{節の名前}
\subsection	小節	\subsection{小節の名前}
\subsubsection	小小節	\subsubsection{小小節の名前}
\paragraph	段落	\paragraph{段落の名前}

入力

```
\documentclass{jarticle}
\begin{document}
\section{論理構造を指定する命令}
\subsection{節と小節}
\verb@\section@命令で節、
\verb@\subsection@命令で小節の開始を指定します。
\subsection{段落}
単に空行を作るだけでも段落を開始できますが、
\verb@\paragraph@命令ならば
見出し付きの段落を開始します。

段落先頭のインデントは自動で付くため、
入力の必要はありません。
見出しの節番号も自動で付きます。
\end{document}
```

出力

1　O@M}9=B$$r;XDj9kL?Na

1.1　@a$H>.@a

\section L?Na$G@a!"\subsection L?Na$G@a$N3+;O$r;XDj7^$9!#

1.2　CJMn

C1$K6u9T$r:nk@1G$bCJMn$r3+;OG-$^$9$,!"\paragraph L?Na$JiP8+=P$7IU$-$NCJMn$r3+;O7^$9!#

CJMn@hF,$N%$%s%G%s%H$O<+F0$GIU$/$?$a!"F~NO$NI,MWO"j^$;$s!#8+=P7N@aHV9f$b<+F0$GIU$-$^$9!#

(3.2)　**物理構造を指定する命令 (下線)**　TEX では物理構造を陽に指定しないことが推奨されるものの、実際には文字サイズなどの物理構造を指定したいこともあります。文章の一部に下線を引くことも物理構造ですが、次の命令で下線を引けます。

入力	出力
\underline{下線}	下線

(3.3)　**物理構造を指定する命令 (改ページ)**　ページを改めることは物理構造ですが、次の命令で改ページできます。

入力	意味
\newpage	改ページする

(3.4) **物理構造を指定する命令 (改行)** TEX は自動的に改行位置を決めますが、それと無関係に改行したいこともあります。次の命令で (地の文の) 任意の位置で改行できます。

入力	意味
\\	改行する

ただし、\\[1cm] のように後ろにブラケットがあると、囲まれた寸法だけ行を送ります。\\の後にブラケットを書きたい場合は、こう解釈されないように、何もしない命令\relax を用いて、\\ \relax [] とするなどの注意が必要です。

(3.5) **物理構造を指定する命令 (文字サイズ、書体)** 文字サイズも物理構造です。次の命令で文字サイズが変更できます。

命令	見本
\tiny	サイズ Sample
\scriptsize	サイズ Sample
\footnotesize	サイズ Sample
\small	サイズ Sample
\normalsize	サイズ Sample
\large	サイズ Sample
\Large	サイズ Sample
\LARGE	サイズ Sample
\huge	サイズ Sample
\Huge	サイズ Sample

サイズ変更の影響範囲を限定するためには、

```
{\small 小さい} 普通
```

のようにブレース ({ }) で囲みます。

入力

```
1 {\small ここだけ小さい、}
2 普通、
3 \large ここだけ大きい、
4
5 つもりだったが以降ずっと大きい。
```

出力

ここだけ小さい、普通、ここだけ大きい、
つもりだったが以降ずっと大きい。

また、次の命令で文字の書体が変更できます。上の\small などの命令とは、ブレースの付き方が違うことに注意が必要です。

入力	意味	出力
\textrm{Roman}	ローマン	Roman
\textbf{Bold}	ボールド	**Bold**
\textit{Italic}	イタリック	*Italic*
\textsl{Slant}	斜体	*Slant*
\textsf{Sans Serif}	サンセリフ	Sans Serif
\texttt{Typewriter}	タイプライター	Typewriter
\textsc{Small Caps}	スモールキャップス	Small Caps
\textgt{ゴシック体}	ゴシック	**ゴシック体**

(3.6) **課題 53 – 文書の論理構造と物理構造** 次のような出力を与える tex ファイル (ファイル名は 0000 和地-53.tex のように) を作り提出して下さい。ただし、TEX が自動的に決定する行の折り返し位置は異なっても構いません。

出力

1 @aJI

1.1 @a!&>.@a!&CJMn

@a$r;O$akK$O\section L?Na$r;H$$!">.@a$r;O$akK$O\subsection L?Na$r;H$$^$9!#

3Nh&$K?7$7$CJMn$r;Oak$?$aKO!"C1$K6u9T$r:nlP$h$$G$9!#$^$?!"9T$NESCf$G2~9T$bG-$^$9!#

1.2 8+=P$7IU$-CJMn

CJMn$K8+=P$7rIUl$?$$$H$-$O!"\paragraph L?Na$r;H$$^$9!#

8+=P$7IU$-CJMn$NNc 3NCJMn$N>eJ}D>A0$N6u$-$O!"\paragraph L?Na$r;H$&$H<+F0E*KIU/bNG9!#

2 J8;z%5%$%:$d=qBN

2.1 J8;z%5%$%:

Nc$($P\footnotesize L?Na$r;H$&$H!">.$5$$J8;zKJj^$9!#

2.2 =qBN

Nc$($P\textsf L?Na$r;H$&$H!"read me!$Nh&$J=qBN$KJj$^$9!#B@;zGBg-$$$J8;z%5%$%:$K7?$$$H$-$O!"\textbf $H\Large rAH_9go;kH!"**like this** Nh$&$K=PNO5l$^$9!#

§4 箇条書き

(4.1) **環境** \begin{document} ··· \end{document} のように\begin と\end で囲まれた構造を**環境**と呼びます。document 環境の他には、

環境名	意味
quote 環境	引用を行う。段落の先頭はインデントしない。
quotation 環境	引用を行う。段落の先頭をインデントする。
flushright 環境	右寄せする。
flushleft 環境	左寄せする。
center 環境	中央揃えする。

などもあります

入力

```
\documentclass{jarticle}
\begin{document}
\begin{quote}
  引用します
\end{quote}
\begin{quotation}
  引用します。インデントもします。
\end{quotation}
\begin{flushleft}
  \Large 大きい字で左寄せ
\end{flushleft}
\begin{flushright}
  右寄せ
\end{flushright}
\begin{center}
中央揃え
\end{center}
\end{document}
```

出力

引用します

> 引用します。インデントもします。

大きい字で左寄せ

右寄せ

中央揃え

上の例でわかるように、環境は{ ··· }と同じ効果があり、\Large 命令の影響範囲を限定しています。

(4.2) **箇条書き環境** 次のような、意味の異なる箇条書き環境があります。

環境名	意味
itemize	番号の付かない箇条書き
enumerate	番号の付く箇条書き
description	見出しの付く箇条書き

これらの環境の本体では、\item 命令に続いて箇条書きする項目を書きます。また、description 環境では、\item[見出し] として見出しを指定します。itemize と enumerate 環境でもこうすると見出しを変更できます。

入力

```
1  \documentclass{jarticle}
2  \begin{document}
3  \begin{itemize}
4    \item かき氷ともに頬ばる日を思う
5    \item 向日葵を見上げる君の大き顔
6  \end{itemize}
7  \begin{enumerate}
8    \item 母の持つ砂場の蚯蚓に後ずさる
9    \item 炎天下滑りて昇りまた滑り
10 \end{enumerate}
11 \begin{description}
12   \item [仲夏] 逃げ惑い泣き疲れ蝿を夢に見る
13   \item [晩夏] 水風船親の心を子は知らず
14 \end{description}
15 \end{document}
```

出力

- かき氷ともに頬ばる日を思う
- 向日葵を見上げる君の大き顔

1. 母の持つ砂場の蚯蚓に後ずさる
2. 炎天下滑りて昇りまた滑り

仲夏 逃げ惑い泣き疲れ蝿を夢に見る

晩夏 水風船親の心を子は知らず

また、例えば、document 環境の中に itemize 環境を入れるなど、多くの場合環境は入れ子にできます。

(4.3) **課題 54 – 箇条書きなどの環境** 次のような出力を与える tex ファイル (ファイル名は 0000 和地-54.tex のように) を作り提出して下さい。ただし、TEX が自動的に決定する行の折り返し位置は異なっても構いません。また、下の出力では掲載を省略していますが、用紙の下部にページ番号も出力されています。

出力

K\F|N^Ha

0000 OBCO51?N

>e$N%?%$%H%k$O\Large NBg-5G$9!#:#F|3Xs@4D6-r^HakH0J2<$Nh&KJj^$9!#

0zMQ quote, quotation

4s$;!&B7$(flushleft, flushright, center

2U>r=q$-itemize, enumerate, description

3N$&$A!"HV9fIU$-$N2U>r=q$-$NNc$O2<$Nh&KJj^$9!#

1. itemize 4D6-$OHV9f$J$7
2. enumerate 4D6-$OHV9fIU$-
3. description 4D6-$O8+=P$7IU$-

§5 表組み

(5.1) **tabular 環境** 表組みは tabular 環境で行います。

tabular 環境の文法

```
\begin{tabular}{列指定}
  表本体
\end{tabular}
```

列指定には、1 列につき、l (左寄せ)、c (中央揃え)、r (右寄せ) のどれかを 1 文字指定します。これらの各文字の前後に| (縦罫線) が指定できます。

表本体において、列の区切りは&、行の終わりは\\で表します。行の先頭に\hline を書くと横罫線が引かれ、\cline{2-3}のように書くと部分的な横罫線 (この場合 2 から 3 列目まで) が引かれます。

横方向にコマを連結したいときは、\multicolumn{連結するコマ数}{列指定}{コマの内容} を用います。

入力

```
1  \documentclass{jarticle}
2  \begin{document}
3  \begin{tabular}{|l|c|rr|}
4  \hline       \multicolumn{4}{|c|}{りんごの表} \\
5  \hline\hline 品名   & サイズ & 単価 & 数量 \\
6  \hline       りんご & 大きい & 180 & 15 \\
7  \cline{2-4}         & ミニ   &  90 & 8 \\
8  \hline
9  \end{tabular}
10 \end{document}
```

出力

りんごの表			
品名	サイズ	単価	数量
りんご	大きい	180	15
	ミニ	90	8

(5.2) **課題 55 – 表組み** 次のような出力を与える tex ファイル (ファイル名は 0000 和地-55.tex のように) を作り提出して下さい。下の出力では掲載を省略していますが、用紙の下部にページ番号も出力されています。

出力

tabular 環境のまとめ		
列指定	揃え	l, c, r
	縦罫線	|
表本体	列区切り	&
	改行	\\
	横罫線	\hline, \cline
	コマの連結	\multicolumn

§6　数式

入力例に毎回書いてきた\documentclass{jarticle}や、document 環境は、そろそろ慣れてきたと思いますので、今後省略することにします。

(6.1)　**インライン数式と別行立て数式**　$で囲まれた部分は数式モードになります。地の文に数式が混じるためインライン数式とも呼ばれます。

入力

```
1 数式モードだと、$a = 1 - 2x$ ですが、
2 通常のモードだと、a = 1 - 2x となり、
3 書体や空白の空き方が異なります。
```

出力

数式モードだと、$a = 1 - 2x$ ですが、通常のモードだと、a = 1 - 2x となり、書体や空白の空き方が異なります。

複雑な数式などを別行立てにしたいときは、\[··· \] を用います。

入力

```
1 インライン数式だと、$a = 1 - 2x$ ですが、
2 別行立てだと、\[ a = 1 - 2x \]
3 となります。
```

出力

インライン数式だと、$a = 1 - 2x$ ですが、別行立てだと、

$$a = 1 - 2x$$

となります。

後述の amsmath パッケージには、他にも便利な別行立て数式の環境があります。[9]

(6.2)　**累乗、添字**　上付きの累乗は「^」、下付きの添字には「_」を用います。累乗や添字が 1 文字でない場合は、{ ··· } でグループ化します。累乗や添字も入れ子にできます。

入力	出力
e^x + e^{-x}	$e^x + e^{-x}$
a_1^{b_1} + a_2^{b_2}	$a_1^{b_1} + a_2^{b_2}$

(6.3)　**点**　1 つの点や複数の点を出力する以下のような命令があります。すべて数式モードでのみ使えます。これらを用いると前後の空白を自動調節してくれるので、全角の「・」や「…」で代用すべきではありません。

\ldots と\cdots の使い分けは、下の表にあるものが正当とする考えが普通ですが、日本の高校までの教科書に見られるように、すべて\cdots で済ませる場合もあります。

[9] amsmath パッケージを使う方がおすすめです。複数行に渡る数式や、それらをイコールの位置で揃えたいだとか、数式に番号を付けたいときは、多くの改善が施された amsmath パッケージを必ず使って下さい。

入力	意味	出力	使用例
\cdot	掛け算記号の点など	$\cdot$	$a \cdot b$
\cdots	長い演算の途中の省略	$\cdots$	$a + b + \cdots + z$
\ldots	長い列挙の途中の省略	$\ldots$	$a, b, \ldots, z$
\vdots	縦方向の省略	$\vdots$	
\ddots	斜め方向の省略	$\ddots$	

(6.4) **根号** 「\sqrt{根号の中身}」でルートの記号を出力できます。3 乗根などを出力する場合は、「\sqrt[3]{根号の中身}」のようにします。

入力	出力
\sqrt{x^2+1}	$\sqrt{x^2+1}$
\sqrt[3]{a_1 + a_2}	$\sqrt[3]{a_1 + a_2}$

(6.5) **分数** 「\frac{分子}{分母}」で分数を出力できます。入れ子にすれば繁分数も可能です。この命令は、インライン数式と別行立て数式で文字の大きさなどが異なります。

入力	出力 (インライン)	出力 (別行立て)
\frac{1}{2}	$\frac{1}{2}$	$\frac{1}{2}$
\frac{1+x}{\sqrt{1+x^2}}	$\frac{1+x}{\sqrt{1+x^2}}$	$\frac{1+x}{\sqrt{1+x^2}}$
1+\frac{2}{3+\frac{4}{5}}	$1+\frac{2}{3+\frac{4}{5}}$	$1+\frac{2}{3+\frac{4}{5}}$

(6.6) **シグマ記号** 和のシグマ記号は「\sum」で出力します。「_」と「^」で、パラメータの動く範囲を書くことができます。この命令は、インライン数式と別行立て数式で記号の大きさや、範囲の付く場所が異なります。

入力	出力 (インライン)	出力 (別行立て)
\sum_{k=1}^n k^2	$\sum_{k=1}^{n} k^2$	$\sum_{k=1}^{n} k^2$

またシグマ記号と同様の振舞いをする大型演算子には以下のようなものもあります。

入力	出力	入力	出力	入力	出力
\prod	$\prod$	\bigcap	$\bigcap$	\bigcup	$\bigcup$

(6.7) **積分記号** 積分記号は「\int」です。積分の上端・下端は「^」と「_」で指定します。この命令は、インライン数式と別行立て数式で記号の大きさや、範囲の付く場所が異なります。

また、そのままだと被積分関数と「dx」などの間が詰まりすぎるので、少しの空白を空ける命令「\,」を挿入するのが適当です。

入力	出力 (インライン)	出力 (別行立て)
\int x dx % 詰まりすぎ	$\int x dx$	$\int x dx$
\int x \, dx	$\int x \, dx$	$\int x \, dx$
\int_0^{10} \sqrt{x} \, dx	$\int_0^{10} \sqrt{x} \, dx$	$\int_0^{10} \sqrt{x} \, dx$

(6.8) **課題 56 – 数式その 1** 次のような出力を与える tex ファイル (ファイル名は 0000 和地-56.tex のように) を作り提出して下さい。

出力

r が 1 ではないとき、等比数列の和の公式は、

$$a + ar + ar^2 + \cdots + ar^{n-1} = \sum_{k=0}^{n-1} ar^k = \frac{a(1-r^n)}{1-r}$$

である。また、$\frac{1}{\sqrt{x}}$ の不定積分は次のようになる。

$$\int \frac{1}{\sqrt{x}}\,dx = \int x^{-\frac{1}{2}}\,dx = -2x^{\frac{1}{2}} + C = -2\sqrt{x} + C$$

(6.9) log **型関数** 対数関数を「`$log x$`」と入力してしまうと、「$logx$」となります。しかし、数学の習慣では log 等のできあいの関数名は斜体ではなくローマン体で書きます。また、x の前に適切な空きもありませんから、決してこうしないで下さい。

正しくは「`\log`」命令を使い、「`$\log x$`」とします。主な log 型の関数を以下に記します。

入力	出力	入力	出力	入力	出力
`\log x`	$\log x$	`\arg x`	$\arg x$	`\exp x`	$\exp x$
`\sin x`	$\sin x$	`\deg x`	$\deg x$	`\lim x`	$\lim x$
`\cos x`	$\cos x$	`\det x`	$\det x$	`\max x`	$\max x$
`\tan x`	$\tan x$	`\dim x`	$\dim x$	`\min x`	$\min x$

シグマ記号と同様に「`^`」や「`_`」を用いると、上限、下限の数式を添えられますが、関数によって、あるいは、インラインか別行立てかによって上限、下限の付く場所が異なります。

入力	出力 (インライン)	出力 (別行立て)
`\log_a x`	$\log_a x$	$\log_a x$
`\lim_{n\to\infty} a_n`	$\lim_{n\to\infty} a_n$	$\lim_{n\to\infty} a_n$

`\to` と `\infty` は、それぞれ矢印と無限大の記号を出力する命令です。

(6.10) **mod** mod には、使い方に合わせて 2 通りの命令があります。

入力	出力
`x \bmod 3`	$x \bmod 3$
`x \equiv 4 \pmod{3}`	$x \equiv 4 \pmod 3$

(6.11) **賢いかっこ** 分数など縦に大きな数式に合わせて、かっこを自動的に伸縮できます。開きかっこは「`\left(`」、閉じかっこは「`\right)`」とすると、その間の数式の高さに合わせてかっこが伸縮します。「`{ }`」や「`[ ]`」など他のかっこも「`\left`, `\right`」をつけると伸縮します。また、左右の片方だけにかっこを出力したいときは、「`\left.`」のように、出力しないかっこの代わりにピリオドを用います。以下では別行立て数式の出力のみ掲げます。

入力	出力 (別行立て)
`\{ \frac{1}{2}, \frac{3}{4} \} % 不釣合い`	$\{\frac{1}{2}, \frac{3}{4}\}$
`\left\{ \frac{1}{2}, \frac{3}{4} \right\}`	$\left\{\frac{1}{2}, \frac{3}{4}\right\}$
`\left( \frac{1}{2} \right)`	$\left(\frac{1}{2}\right)$
`\left[ \frac{1}{2} \right.`	$\left[\frac{1}{2}\right.$

(6.12) **数式の表組み** 地の文では tabular 環境を用いて表組みしましたが、数式モードでは array 環境を用います。列指定などは tabular 環境のときと同じです。

入力

```
1 \[
2   \left(
3     \begin{array}{cc}
```

```
4       1 & 2 \\
5       3 & 4
6     \end{array}
7   \right),
8   \left\{
9     \begin{array}{l}
10      2x + 3y = 5 \\
11      x - 6y = -5
12    \end{array}
13  \right.
14 \]
```

出力

$$\left(\begin{array}{cc} 1 & 2 \\ 3 & 4 \end{array} \right), \left\{ \begin{array}{l} 2x + 3y = 5 \\ x - 6y = -5 \end{array} \right.$$

ただし、行列や連立方程式は、後述の amsmath パッケージを使う方がすっきりと書けます。[10]

(6.13) **上下に付けるもの** 数式モードだけで使えるアクセント記号には、おもに以下のものがあります。

入力	出力	入力	出力
`\hat{a}`	$\hat{a}$	`\tilde{a}`	$\tilde{a}$
`\bar{a}`	$\bar{a}$	`\vec{a}`	$\vec{a}$

以下のものは、数式の長さに応じて伸縮します。

[10] 必ず amsmath パッケージを使って下さい。

入力	出力	入力	出力
`\overline{a+b}`	$\overline{a+b}$	`\underline{a+b}`	$\underline{a+b}$
`\widehat{a+b}`	$\widehat{a+b}$	`\widetilde{a+b}`	$\widetilde{a+b}$
`\overbrace{a+b}`	$\overbrace{a+b}$	`\underbrace{a+b}`	$\underbrace{a+b}$
`\overleftarrow{a+b}`	$\overleftarrow{a+b}$	`\overrightarrow{a+b}`	$\overrightarrow{a+b}$

`\overbrace` と `underbrace` は、上限・下限が真上・真下に付きます。

入力	出力
`\overbrace{a+\cdots+z}^{26}`	$\overbrace{a+\cdots+z}^{26}$
`\underbrace{a+\cdots+z}_{26}`	$\underbrace{a+\cdots+z}_{26}$

(6.14) **その他の記号** これまでに紹介したもの以外にも、記号を出力する命令が大量に定義されています。おもなものを以下に記します。

矢印

入力	出力	入力	出力
`\leftarrow`	$\leftarrow$	`\rightarrow`	$\rightarrow$
`\uparrow`	$\uparrow$	`\downarrow`	$\downarrow$
`\updownarrow`	$\updownarrow$	`\leftrightarrow`	$\leftrightarrow$
`\Leftarrow`	$\Leftarrow$	`\Rightarrow`	$\Rightarrow$
`\Uparrow`	$\Uparrow$	`\Downarrow`	$\Downarrow$
`\Updownarrow`	$\Updownarrow$	`\Leftrightarrow`	$\Leftrightarrow$
`\nearrow`	$\nearrow$	`\swarrow`	$\swarrow$
`\searrow`	$\searrow$	`\nwarrow`	$\nwarrow$

ギリシア文字

入力	出力	入力	出力	入力	出力
\alpha	α	\beta	β	\gamma	γ
\delta	δ	\epsilon	ϵ	\zeta	ζ
\eta	η	\theta	θ	\iota	ι
\kappa	κ	\lambda	λ	\mu	μ
\nu	ν	\xi	ξ	o	o
\pi	π	\rho	ρ	\sigma	σ
\tau	τ	\upsilon	υ	\phi	ϕ
\chi	χ	\psi	ψ	\omega	ω
\varepsilon	ε	\varphi	φ		
\Gamma	Γ	\Delta	Δ	\Theta	Θ
\Lambda	Λ	\Xi	Ξ	\Pi	Π
\Sigma	Σ	\Upsilon	Υ	\Phi	Φ
\Psi	Ψ	\Omega	Ω		

二項演算子

入力	出力	入力	出力	入力	出力	入力	出力
\pm	$\pm$	\mp	$\mp$	\times	$\times$	\div	$\div$
\circ	$\circ$	\cap	$\cap$	\cup	$\cup$		

関係演算子 [11]

入力	出力	入力	出力	入力	出力	入力	出力
<	$<$	\subset	$\subset$	\in	$\in$	\equiv	$\equiv$
>	$>$	\supset	$\supset$	\ni	$\ni$	\cong	$\cong$
\le	$\le$	\subseteq	$\subseteq$	\notin	$\notin$	\sim	$\sim$
\ge	$\ge$	\supseteq	$\supseteq$	\neq	$\neq$	\perp	$\perp$

[11] le や ge といった命令の名前は、less than や greater than から来ている。

その他

入力	出力	入力	出力	入力	出力
\infty	∞	\partial	∂	\ell	ℓ

(6.15) **課題 57 – 数式その 2** 次のような出力を与える tex ファイル (ファイル名は 0000 和地-57.tex のように) を作り提出して下さい。

出力

$0^\circ \le \theta < 180^\circ$ とすると $\sin\theta \ge 0$ だから、$\sin\theta = \sqrt{1-\cos^2\theta}$ である。また、$a, b \in \{3n+1 \,|\, n = 1, 2, \ldots\}$ のとき、$a + b \equiv 2 \pmod 3$ である。

$$\lim_{n\to\infty} \left(1 + \frac{1}{n}\right)^n = e.$$

(6.16) **行列、行列式** 行列を記述する、より簡単な方法があります。\left(, \right) と array 環境の組合せによる記述と比較すると、はるかに簡潔であることがわかります。

入力

```
\documentclass{jarticle}
\usepackage{amsmath}
\begin{document}
\[
  \begin{pmatrix}
    1 & 2 & 3 \\
    4 & 5 & 6
  \end{pmatrix},
  \begin{vmatrix}
    a & b \\
    c & d
  \end{vmatrix}
\]
\end{document}
```

出力

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix}, \begin{vmatrix} a & b \\ c & d \end{vmatrix}$$

pmatrix 環境は上のように使用しますが、\begin{document}の前に\usepackage{amsmath}と記述して、amsmath パッケージを読み込む必要があります。後述のように amsmath パッケージには、他にも便利な数式の命令が多数定義されています。

また、vmatrix 環境は行列式のための環境です。

(6.17) **場合分けのある数式** 連立方程式や場合分けを記述するための cases 環境が、amsmath パッケージに定義されています。\left\{, \right. と array 環境の組み合せによる記述と比較して下さい。

入力

```
\documentclass{jarticle}
\usepackage{amsmath}
\begin{document}
\[
\begin{cases}
  2x + 3y = 5 \\
  x - 6y = -5
\end{cases}
\]
\end{document}
```

出力

$$\begin{cases} 2x + 3y = 5 \\ x - 6y = -5 \end{cases}$$

(6.18) **複数行の別行立て数式** 長い式変形や複数行の別行立て数式のための環境が、いくつか amsmath パッケージに定義されています。

入力

```
\documentclass{jarticle}
\usepackage{amsmath}
\begin{document}
\begin{gather}
  x^2+x+1 = 1 \\
  y = 2
\end{gather}
\begin{align*}
  f(x) &= x^2+x+1 \\
  y &= 2
\end{align*}
\begin{align}
  (\sqrt{x})'
  &= (x^{\frac{1}{2}})' \\
  &= \frac{1}{2} x^{-\frac{1}{2}} \\
  &= \frac{1}{2\sqrt{x}}
\end{align}
\end{document}
```

出力

$$x^2 + x + 1 = 1 \tag{1}$$
$$y = 2 \tag{2}$$

$$f(x) = x^2 + x + 1$$
$$y = 2$$

$$(\sqrt{x})' = (x^{\frac{1}{2}})' \tag{3}$$
$$= \frac{1}{2}x^{-\frac{1}{2}} \tag{4}$$
$$= \frac{1}{2\sqrt{x}} \tag{5}$$

gather 環境は中央揃え、align 環境は&の位置で揃えられます。これらの機能は需要が高いですが、amsmath パッケージに頼らないと満足いく出力は得られません。gather, align 環境とも数式番号が表示されます。これは*が付加されると抑制されます。

(6.19) **課題 58 – 数式その 3** 次のような出力を与える tex ファイル (ファイル名は 0000 和地-58.tex のように) を作り提出して下さい。

出力

$$\begin{vmatrix} 1 & 3 & 2 \\ 3 & 4 & 5 \\ 1 & -1 & 3 \end{vmatrix} = \begin{vmatrix} 1 & 3 & 2 \\ 0 & -5 & -1 \\ 0 & -4 & 1 \end{vmatrix}$$
$$= \begin{vmatrix} -5 & -1 \\ -4 & 1 \end{vmatrix}$$
$$= -5 \cdot 1 - (-1) \cdot (-4)$$
$$= -9$$

§7 相互参照・目次 (・索引)

(7.1) **相互参照** 「第 7 節を参照のこと」とか、「33 ページにある式 (2) により」のように節などの番号を文中に記述したいとき、じかに「32 ページ」のように書くと、原稿の修正でページがずれたときや、節を新たに挿入したときに手作業ですべて直す必要があります。\label, \ref, \pageref の命令を使うと、TEX が自動的に番号を適切に決定してくれます。

入力

```
\documentclass{jarticle}
\begin{document}
\section{目次と索引}
\label{sec:sogo-sansho}
\begin{equation}
  \label{eq:parab}
  y = x^2
\end{equation}
\section{相互参照}
\subsection{節番号やページ番号}
\label{subsec:setsu-page}
ここは、第\ref{sec:sogo-sansho}節の
\ref{subsec:setsu-page}です。
式(\ref{eq:parab})は
```

```
15 \pageref{subsec:setsu-page}ページにあります。
16 \end{document}
```

出力

1　目次と索引

$$y = x^2 \tag{6}$$

2　相互参照

2.1 節番号やページ番号

ここは、第 2 節の 2.1 です。式 (6) は 32 ページにあります。

この中で、equation 環境は数式番号が付く別行立て数式です。また、align や gather 環境でも相互参照が可能です。

(7.2)　**課題 59 – 相互参照**　TEX の相互参照の機能を用いて、次のような出力を与える tex ファイル (ファイル名は 0000 和地-59.tex のように) を作り提出して下さい。

出力

1　2r$H78?t$N4X78

$x^2 - x + 1 = 0$ $N2 DN2rrrα, β H9kH!"

$$\alpha\beta = 1, \quad \alpha + \beta = 1 \tag{1}$$

G"k+$i!"

$$\alpha^2 + \beta^2 = (\alpha + \beta)^2 - 2\alpha\beta \tag{2}$$
$$= 1 - 2 \cdot 1 \tag{3}$$
$$= -1 \tag{4}$$

G"$k!#

2　JdB-

Bh??@a$N(??) <0$+$i(??) <0$X$NJQ7A$G$O(??) <0$rMQ$$?!#

2 つ目の別行立て数式では、align 環境を用います。また 1 つ目の別行立て数式の、2 つの式の間の空白は、\quad 命令で空けます。これは以前用いた\, 命令よりも広い空きを作ります。

(7.3)　**目次**　\tableofcontents 命令を書くと、そこに目次が作成されます。

入力

```
1 \documentclass{jarticle}
2 \begin{document}
3 \tableofcontents
4 \section{目次と索引}
5 最初の節
6 \section{相互参照}
7 次の節
8 \subsection{節番号やページ番号}
9 いろいろ書く。
```

```
10 \end{document}
```

出力

L\<!

1 L\<!$H:w0z

:G=i$N@a

2 Aj8_;2>H

<!$N@a

2.1 @aHV9f$d%Z!<%8HV9f

$$$m$$$m=o$/!#

§8 画像

(8.1) **画像の挿入** Windows の dviout で BMP 画像ファイル表示させることを中心に説明します。他の OS、出力方法 (pdf に変換するなど)、画像形式 (JPEG, PNG, EPS, PDF など) でも基本は同じですが、別途プログラムをインストールしなくてはならないこともあります。[12]

[12] BMP 形式ではない画像を BMP 形式に変換したいならば、例えば、画像を Windows の「ペイント」で開き、ファイルメニューの「名前を付けて保存」で BMP 形式を選んで保存すると、新しく BMP ファイルが出来ます。

TEX の文書に画像を挿入するには、graphicx パッケージを利用します。dvioout で出力するためには、dvioout オプションが必要です。挿入したい所で、\includegraphics 命令を用いますが、BMP などの画像形式では、bb オプション (バウンディングボックス。画像のサイズ) を指定する必要があります。

includegraphics **命令の文法**

```
\includegraphics{画像ファイル名}
\includegraphics[オプション]{画像ファイル名}
```

includegraphics 命令のオプションの例

オプション	意味
scale=1.2	画像を 1.2 倍に拡大して出力する
bb=0 0 100 200	サイズが横 100 縦 200 ピクセルである指定

入力

```
1 \documentclass{jarticle}
2 \usepackage[dviout]{graphicx}
3 \begin{document}
4 0.3倍の大きさ。
5 \includegraphics[bb=0 0 240 320,scale=0.3]{t.bmp}
6 \end{document}
```

出力

0.3 G\$NBg$-$5!#

§9　マクロ

更新日時 2011-08-07 17:15:04

http://alg.kus.hokkyodai.ac.jp/にこの pdf が置いてあります。