コンピュータ概論 A/B (竹澤先生,栗野)

-- mathematica --

数学科 栗野 俊一

2010/10/12 コンピュータ概論

伝言

- 補習
  - 今日は、1221 室で、5 限で、補講を行います
    - Mathematica は本日中でインストールする事 !!
- 調べておきましょう
  - 次のキーワードを google で調べておきましょう
    - 数式処理
    - Mathematica プログラミング
    - 再帰的定義
  - 便利と思ったページの URL は、skype で交換しましょう

注意

- 講義前の注意
  - 講義開始前にすませておくこと
    - PC の電源を入れる/ネットワークに接続しておく/今日の資料に目を通す
  - 過去ログ(記録)を参照しましょう
    - 新しいメールが届いていないか確認しましょう
    - CST Portal のフォーラムの新着の記事に目を通しましょう
    - skype を起動し、新しいメッセージがないか確認しましょう
    - ついでに何か発言する習慣を身に付けましょう(挨拶も可)
- やる気のある方へ
  - 今日の資料は、すでに上っています
    - どんどん、先に進んでかまいません

先週の復習

□ 先週の内容

○ 講義

- ▹ Mathematica

○ 実習

- ▹ [演習 1] Mathematica のインストール
- ▹ [演習 2] Mathematica の操作
- ▹ [演習 3] 課題の作成

□ 講義内容

○ Mathematica

- ▹ 数式処理言語

○ 文字式が扱える高級電卓

- ▹ 多倍長(長い桁の数値/精度の高い数値)の計算ができる
- ▹ 文字式の計算ができる(式の展開や因数分解)
- ▹ 微分積分や極限の計算もできる
- ▹ ベクトルや行列の計算もできる
- ▹ 関数のグラフも描画できる

○ ポイント

- ▹ 大学 1/2 年で学ぶ「計算」はほとんど、Mathematica でできる
- ▹ ヘルプの見方を覚えれば、Mathematica の機能は自分で調べられる

本日の予定

- 講義
  - Mathematica によるプログラミング基礎
- 実習
  - [演習 1] Mathematica の変数の利用法
  - [演習 2] Mathematica の関数の作成方法
  - [演習 3] 課題の作成

本日の課題 (2010/10/12)

- 前回 (2010/10/05) の課題
  - なし
    - 前回の課題を今週の課題とする
- 今週 (2010/10/12) の課題
  - 次のファイルを提出しなさい
    - 表題 : Mathematica の課題
    - ファイル名 : 20101005-QQQQ.nb (QQQQ は学生番号)
    - 詳しくは、配布した sample-20101005.nb の内容を参照

構文(表現)と意味

- □ 構文規則
  - ○「文字の並び」を「何の表現か?」と「判断するための規則」
    - ▹「123 + a」は、「123」、「a」、「+」に分解される
    - ▹「123」は、「1」、「2」、「3」という「数字列」なので「数値」
    - ▹「a」は、「a」という「英文字から始まる文字列」なので「変数」
    - ▹「+」は、「足し算」を表す「演算子」
    - ▹「123 + a」全体は、「式」になる
  - ○「どんな文字から構成されるか」という基準で区別される
    - ▹「何か」を表す、「表現方法」の規則
    - ▹規則に従って、「文字列」をみると、幾つかの「構文要素」に分解される
- □ 意味規則
  - ○構文規則に対応した「意味」を決める規則
    - ▹「123」という「数字列」は、123 という「整数値」
    - ▹「a」という「シンボル」は、a という「変数」
    - ▹「123 + a」は、「二つの数を加える」という「文字式」
- □ 言語
  - ○構文規則 (いわゆる文法)と、意味規則(「辞書」ならびに、「意味」) から成る
    - ▹「1 + 2」は「構文的」には、「「1」と「2」の「足し算」」を表す「式」となる
    - ▹「1 + 2」は「意味的」には、「3」(1, 2 は数値を表し、+ は足し算を表すから)

Mathematica Notebook

- Mathematica Notebook (*.nb)
  - Mathematica の計算結果を保存する形式
  - 結果の保存方法
    - [ファイル]→[別名で保存] : 好きな名前で保存できる
    - [ファイル]→[保存] (Ctrl-S) : 今の名前で内容を更新する(古い内容は失われる)
  - 結果の利用方法
    - [ファイル]→[開く] (Ctrl-O) : 以前に保存した内容を読み込む
- 電卓としての利用
  - 「式」を入れて [Shift]+[Enter] (以下、[SE]) すると計算
    - 入力した式は In[番号] の形で表示される
    - 計算した結果は Out[番号] の形で表示される
    - 番号は、[SE] の順番につけられる
  - [SE] の効果
    - 「式」の評価が行われる
    - 「番号」が増える
    - In[番号]/Out[番号]が、それぞれ「定義」される

変数とその値

□ シンボル (構文的な要素)

○ シンボルとは

- ▹「英字」から始まり「英数字」からなる文字列の事 (a, abc, a12z)
- ▹(注意) 「英字」には、「ギリシャ文字」も含まれる (π , α)
- ▹cf. 「数字」から始まる文字列はシンボルではない (2a は 2 × a となる)

○ Mathematica は、大文字で始まるシンボルの幾つかを事前に定義している

- ▹(注意) 自分が利用する場合は、小文字で始まるシンボルを利用する事

□ 変数 (意味的な要素)

○ 変数とは

- ▹名前として「シンボル」をもち、「値を持つ事ができるもの」
- ▹変数名は、変数の名前で、それは「シンボル」である

○ (即)値とは

- ▹(現時点で)変数に対応している「式」の事
- ▹「環境」は、「変数名」と「変数の値」の対応表をもっている

○ 変数の値の参照

- ▹「シンボル」の形を示すを入れると、「その値」が評価され、その結果が表示される
- ▹「値」が設定されていない場合は、その「変数自身」が「変数の値」となる
- ▹?変数名で、「変数の値」である「式」を見る事ができる

変数への代入と定義

- 代入と定義
  - 共に変数に値を「割り当て」る
  - 代入：「変数 = 式」
    - 「式」は「即時評価」される。その評価結果が「変数の値」となる
  - 定義：「変数 := 式」
    - 「式」は「遅延評価」される。その「式」そのものが「変数の値」となる
  - 変数の値は何度でも変更できる
- 環境
  - 変数名とその値の対応表をもっている
    - 代入も定義も、その対応表を書換えている
    - 書き換える値が違うだけ
- 「式」の評価(変数の場合)
  - 式の中に変数名が含まれていた場合に、それを変数の値に書き換える

関数の利用

- □ 関数
  - ○ 関数とは
    - ▹ 「シンボル[引数列]」の形で表現され、値を持つ事ができる「もの」
    - ▹ cf. Sin[Pi] / f[3] / g[x,y^4]
  - ○ 関数の評価
    - ▹ 「シンボル[引数列]」の形を、その値に置き換える
- □ 引数のパターンマッチ
  - ○ 「_」(アンダースコア)を利用して「式」の「抽象パターン」が表現できる
  - ○ 例1
    - ▹ next[0] := 1
    - ▹ next[1] := 2
    - ▹ next[_] := 0
  - ○ 例2
    - ▹ fib[1]=1
    - ▹ fib[2]=1
    - ▹ fib[x_]:=fib[x-1]+fib[x-2]

自然数

- ペアノの公理
  - 自然数を定義する公理
    - 0 は自然数である
    - n が自然数なら n+1 も自然数である
    - 上記の二つ以外に自然数はない
- Mathematica でペアノの公理の自然数を考える
  - 自然数
    - 0 は 0 で表現
    - n + 1 は s[n] で表現
  - 自然数の和
    - padd[0,x_] := x
    - padd[s[x_],y_] := s[padd[x,y]]
  - これを繰り返すと、分数まで表現できる