KeTCindy/KeTCindyJSの利用

高遠節夫（東邦大）

明治大教材作成セミナー

2019.09.03

CindyScript

ヘルプ

- Cinderellaのトップメニューで
  ヘルプ ＞マニュアルを読む
- CindyScriptを選ぶ
- help

幾何点とリスト点（1）

- 幾何点は，画面で「点を加える」で作る点
  - マウスで動かすことができる
  - いろいろな属性をもつ
    - A.xy(座標)，A.x（x 座標），A.y（y 座標）
    - A.size（点の大きさ），A.color（色 RGB）
      - 「インスペクタ」で設定可能
- 消すには，画面の消しゴムを用いる
  - CindyScript では消せない

幾何点とリスト点（2）

- KETCindy は幾何点を追加するコマンドをもつ

```
Putpoint("B",[1,2],B.xy);
Putoncurve("B","sg1");
Slider("S",[0,-1],[5,-1]);
```

- リスト点は，長さが 2 のリストで定義される点

  pt=[2,3];　pt_1（x 座標），pt_2（y 座標）

- リスト点は CindyScript で簡単に消せる

- 動きは CindyScript で制御

幾何点とリスト点（3）

- リスト点を画面に表示するには

  Pointdata(”1”,pt,[”Size=3”,”Color=red”]);

- 幾何点は TEX に出力されない.
- リスト点は Pointdata で TEX に出力される

  オプションに “notex” を追加 =>TEX に出ない

- その他の幾何要素(線分,円,2 次曲線など)は,KETCindy では余り使わない

リスト（1）

- $[1, 5, 3]$, $1..5 = [1, 2, 3, 4, 5]$
- **リストの長さ**　length(list)

  注意）1..length(list) **は** CindyJS **でエラー**
- n **番目の要素**　list_n
- **リストの追加**

  list2=append(list1,"a");**（前方追加は** prepend**）**
- **リストの結合**

  list3=concat(list1,list2);

条件分岐

```
if(条件,
    真のときの実行文;
,
    偽のときの実行文;
);
```

- **条件**

  islist(a)（**リストか**）, isstring(a)（**文字列か**）

  contains(list,a)（**list が a を含むか**）

繰り返し

```
forall(1..10,
  println(#); z
);
forall([2,5,4], k,
  println(k);
);
```

- 他の繰り返しとして　repeat, while
  while は要注意

リスト（2）

- 要素の削除　remove(リスト, 要素)
- 逆順リスト　reverse(リスト)
- ソーティング　sort(リスト)
- 式の適用　apply

```
list2=apply(list1,2*#);
```

- 要素の選択　select

```
list2=select(list1, #<10);
```

文字列処理（１）

- 連結　”abc”+”de”
- 長さ　length(文字列)
- 部分文字列　substring(文字列, 始め-1, 終り)
- 文字列から数（リスト）へ　parse(文字列)
- 数（リスト）から文字列へ

  text(数), format(数, 桁数)

  Textformat(数リスト, 桁数), Sprintf(数, 桁数)

文字列処理（２）

- **検索**　indexof(str)

  KeTCindy **に追加**　Indexall(str)

- **分解**　tokenize(str,expr)

  KeTCindy **に追加**　Strsplit(str,expr)

  tokenize(”a,b,c”,”,”); ＝＞ [”a”,”b”,”c”];

- **置換**　replace(str,str1,str2);

関数

- 関数定義（例）

```
Dotprod(v1,v2):=(
  regional(nn, out,tmp);
  nn=length(v1);
  tmp=apply(1..nn,v1_#*v2_#);
  out=sum(tmp);
  out;
);
```

- regional にない変数はグローバル


KETCindy

ヘルプ

- setwork.bat(.command) を実行する

  KeTCindyGuideJ.pdf（概要）

  KeTCindyReferenceJ.pdf（関数一覧）

- CindyScript で

  Help(”関数名”); ＝＞ヘルプが表示される

- KeTCindy の関数は大文字で始まる.

手順

- KeTCindy のファイルの 1 つをダブルクリック
- CindyScript を開き，コマンドを記述
- ギヤマークを押す
- 画面に戻り，Figure ボタンを押す

  R, TEX, pdf ビューアが順に起動する
- 図ファイル (.tex) は fig フォルダにできるので，使いたいフォルダに移動する

Script の基本形（１）

- **通常は，draw/figures に記述**
  initialization/ketlib はライブラリ読み込み
- Ketinit();//カレントディレクトリはDircdy/iig
  p1=[1,2]; p2=[3,1];
  Listplot("1",[p1,p2]); // sg1に線分データが入る

  Plotdata("1","x^2","x"); //gr1にグラフデータが入る
  Windispg();

標準設定の変更

- Ketinit("");//カレントディレクトリをDircdyにする

  Setfiles(Cdyname()+"2");// ファイル名を変更

  Usegraphics("pict2e");//デフォルトはTpic

  Addpackage(["emath"]);//Packageを追加

  Addax(0);//座標軸を描かない

  Setax(["l","x","e","y","n","O","sw","Size=1"]);


KETCindyJS

手順

- トップメニューから
  ファイル＞ CindyJS に書き出す
- 通常は Dircdy/fig に作られるが，他のファイルを同時に開いていたりすると，カレントディレクトリが変わっている可能性があるから，要注意
- KeTJS オンラインバージョンの HTML を作成
- KeTJSoff オフラインバージョンの HTML を作成

設定

```
Setketcindyjs(["Nolabel=all","Color=offwhite"]);
Setketcindyjs(["Scale=1","Grid=0.5"]);
Setketcindyjs(["","Fig=y","Axe=n"]);
```

アニメーション

数式処理の利用