

KETCindy 体験授業

NIT, Numazu College

このガイドの画像は、macOS 13.5 Ventura, Cinderella 3.0beta build2074 を使用して作成しています。

1 KETCindy の準備

KETCindy の作業フォルダを開き、**templates** フォルダを開いてください。作業フォルダは既定では C:\ketcindyx.x.xx\doc\work にあります。

00start.cdy を複製してファイル名を変更し、このファイルをベースとします。

複製した 00start.cdy を開いてください。

2 グラフを描く

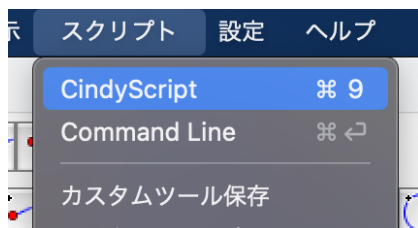


図1 CindyScript の選択メニュー

メニューから**スクリプト**を選択し、CindyScript を選択してください。

出てきたウィンドウで、CindyScript やKETCindy を書くことができます。

グラフを書くには、Plotdata 関数を利用します。

関数とは

プログラムにおける**関数**とは、毎回与えられたデータをもとに、毎回決まった処理を実行するというものです。

Plotdata 関数は、以下のような書き方です。

```
Plotdata(グラフの ID, "数式", "変数と定義域", "グラフの設定");
```

Plotdata 関数の注意

グラフの ID を同じにして Plotdata 関数を繰り返して実行することでグラフを書き換えることができます。

$y = ax$ のグラフを書く場合、 $a * x$ のようにして $*$ を書く必要があります。

実際に $y = 3x$ のグラフを表示してみます。

CindyScript の入力画面の Ketinit(); と Windispg(); の間に

```
Plotdata("1", "3*x", "x");
```

と入力してください。また、最後のコロン (;) を忘れないようにしてください。

```
1 Ketinit();
2 //Setparent(Cdynam("p"));
3 //Setunitlen("1cm");
4 //FontSize("n");
5 Plotdata("1", "3*x", "x");
6
7 //Figpdf();
8 //BBdata();
9 Windispg();
```

図2 入力後のプログラム

歯車マークを押して実行すると、以下のグラフが生成されれば成功です。

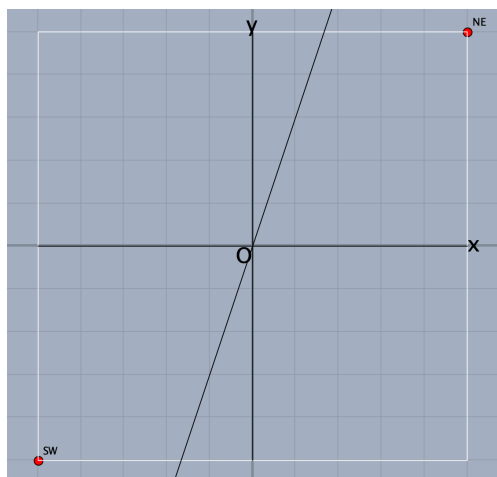


図3 生成されたグラフ

Plotdata 関数の式を変更して、いろいろなグラフを作ってみてください。

3 グラフを動かそう

スライダーを使ってグラフの傾きと切片を変更してみましょう。

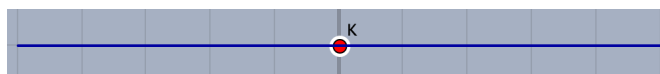


図4 実際に作るスライダー

スライダーを作るには Slider 関数を利用します。Slider 関数は以下のような書き方をします。
`Slider(" スライダー ID", [x 座標 1,y 座標 1], [x 座標 2,y 座標 2]);`

実際に傾きを変更してみます。最初にした Plotdata 関数より上の行に

`Slider("K", [-5, -6], [5, -6]);`

と入力してください。これでスライダーと動かすための点を作成しました。

また、スライダー ID を "A" とした時、そのスライダーの点の x 座標は `A.x`、y 座標は `A.y` と書くことでプログラム側から取得できます。

Tips

今まで Plotdata 関数中の関数の切片は固定でしたが、この固定の値ををスライダーの点の値にすることができます。

Plotdata のグラフの関数の 3 を `K.x` に置き換えてください。

```
1 Ketinit();  
2 //Setparent(Cdynam()+"p");  
3 //Setunitlen("1cm");  
4 //FontSize("n");  
5  
6 Slider("K", [-5, -6], [5, -6]);  
7 Plotdata("1", "K.x*x", "x");  
8  
9 //Figpdf();  
10 //BBdata();  
11 Windispg();
```

図5 入力後のプログラム

歯車マークを押して実行すると、以下のグラフが生成され、スライダーを動かすことでグラフの傾きが変われば成功です。

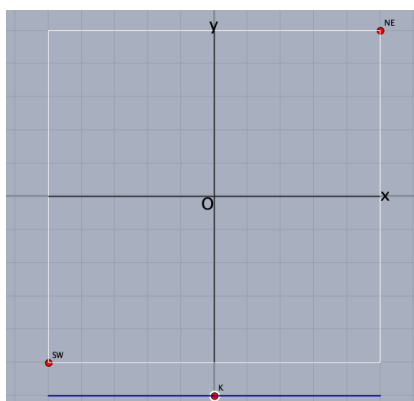


図6 生成されたグラフ

このような操作で新しくスライダーを作ることができます。

先ほどのような操作で新しくスライダーをつくり、切片もスライダーで動かせるグラフを作ってください。

4 発展 二次関数のグラフ

ここまで完成した方は数式の中に²と書くことで2乗を実現できます。

$y = x^2$ のグラフは、`Plotdata("1", "x^2", "x");` となります。

まずは $y = ax^2$ の a の値をスライダーで変更できるプログラムを書いてみましょう。

5 発展2 二次関数を並行移動

二次関数 $y = ax^2$ を x 方向に p , y 方向に q 移動した時の式は、 $y = a(x - p)^2 + q$ です。高専1年次でこの内容を学習します。例えば、 $y = 2(x - 2)^2 + 3$ のグラフは図7の緑のグラフのようになります。

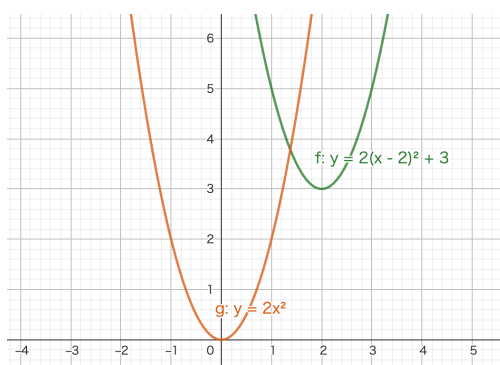


図7 $y = 2x^2$ のグラフ (g) と $y = 2(x - 2)^2 + 3$ のグラフ (f)

図7のように、 $y = a(x - p)^2 + q$ のグラフは、 $y = 2x^2$ のグラフを x 軸方向に2、 y 軸方向に3移動したグラフになることがわかります。

$y = a(x - p)^2 + q$ の a, p, q をスライダーで変更し、二次関数を並行移動するプログラムを書いてみましょう。また、Plotdata 関数の数式部分では、括弧もそのまま入力することができます。

6 発展 二次関数のグラフ 解説

まずは値を動かせるように Slider 関数を追加します。

```
Slider("A", [-5, -6], [5, -6]);
```

また、Plotdata 関数はこのようになります。

```
Plotdata("1", "A.x*x^2", "x");
```

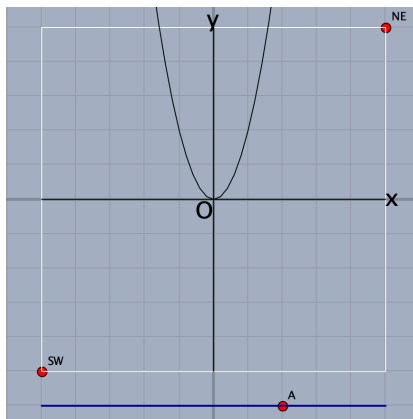


図8 実行結果

7 発展2 二次関数を並行移動 解説

まずスライダーで動かせるようにする値は3つなので、Slider 関数を3つ書きます。また被らないように位置やスライダーのIDを割り当てます。

```
Slider("A", [-5, -6], [5, -6]);
```

```
Slider("P", [-5, -7], [5, -7]);
```

```
Slider("Q", [-5, -8], [5, -8]);
```

ここでは、 a の値を動かすスライダーを"A"、 p の値を動かすスライダーを"P"、 q の値を動かすスライダーを"Q"としています。

また、Plotdata 関数はこのようになります。

```
Plotdata("1", "A.x*(x-P.x)^2+Q.x", "x");
```

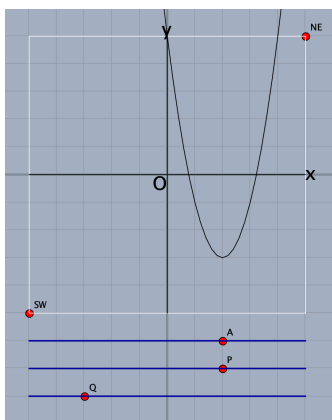


図9 実行結果