# ユーザ定義手続きーprocー



## proc

複雑な手続きや，何度も繰り返すルーチンはprocで作る．

## 引数

procは以下のようにして作る．

| Mapleスクリプト | Maple文法 | C言語 |
|---|---|---|
| `test1:=proc(a)`<br>`  printf("%f\n",a)`<br>`;`<br>`end proc;` | ユーザ関数名:=proc(仮引数)<br>　動作<br>end proc; | void ユーザ関数名(仮引数){<br>　動作<br>} |

procの呼び出しは，以下のようになる．

```
> test1(13);
13.000000
```

引数としてはどんな型(変数や配列)でもよい．複数指定するときにはコンマで区切る．引数をprocの中で変更すると怒られる．下で示すglobalで取り込むか，local変数にコピーして使う．

## 戻り値

procの戻り値はreturnで指定される．return文がないときは，最後の動作結果が戻り値となる．

| Mapleスクリプト | Maple文法 | C言語 |
|---|---|---|
| `test1:=proc(a)`<br>`  printf("%f\n",a)`<br>`;`<br>`  return a+1;`<br>`end proc;` | ユーザ関数名:=proc(仮引数)<br>　動作<br>　return 戻り値;<br>end proc; | double ユーザ関数名(仮引数){<br>　動作;<br>　return 変数;<br>} |

## グローバル(大域)，ローカル(局所)変数

```
変数名:=proc(引数)
  local 変数;
  global 変数;
  動作の記述
end proc;
```

procの内部だけで使われるのがlocal, 外部を参照するのがglobal．global,localを省略してもMapleが適当に判断してくれる．

## 例題

リストを受け取ってその和を返すprocをつくれ．aというリストを受け取り，要素数を見て，和をとり，最後にその和を返す．

```
> total:=proc(a)
  local S,n,i;
  n:=nops(a);
  S:=0;
  for i from 1 to n do
    S:=S+a[i];
  end do;
  return(S);
  end proc:
```

以下のように実行する．(結果は未表示)

```
> aa:=[3,5,7];
  total(aa);
```

## 演習

次のMapleスクリプトを指示に従って書き換えよ．

```
> a:=35; b:=29;
  c:=a+b;
  printf("%3d + %3d =\n",a,b);
  printf("Answer =%3d\n",c);
```

1．a1, b1を仮引数として
　printf("%3d + %3d =\n",a1,b1);
で表示する手続きsum1を作成せよ．sum1(a,b);で呼び出せ．

2．a,bを仮引数とし，その和を返す手続きsum2を作成せよ．戻り値を受け取って，
　printf("Answer =%3d\n",c);
として表示せよ．

3．bだけを引数として，グローバル変数aとの和を局所変数ccにいれて，その2乗を返す手続きsum3を作成せよ．

do-loopの演習で作った素数判定プログラムをproc化せよ．