

Introduction to GP programming

B. Allombert

IMB
CNRS/Université de Bordeaux

16/04/2018



This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement N° 676541

Start by creating a text file `prog.gp` with the following content :

```
dist(a,b) = sqrt(a^2+b^2);
```

and save it. Under GP do (without hitting return) :

```
\r
```

Under Windows drag the file icon to the windows to complete the name :

```
\r .../prog.gp
```

On Linux you can just type the name. Call the function :

```
? dist(1,2)
```

```
%1 = 2.2360679774997896964091736687312762354
```

In a GP file, line ending terminates the line unless they are preceded by = or \ or are in a section delimited by braces :

```
cmpz(a,b) = if (abs(a)>abs(b), print(a), print(b))
cmpz(a,b) =
  if (abs(a)>abs(b), print(a), print(b))
cmpz(a,b) = if (abs(a)>abs(b), \
               print(a), \
               print(b))
```

```
cmpz(a,b) =
{
  if (abs(a)>abs(b),
      print(a)
      ,print(b))
}
```

Example of function

Add to the file `prog.gp`

```
fibonacci(n) =  
{  
  my(u0=0, u1=1);  
  for(i=2, n,  
    [u0, u1]=[u1, u0+u1]);  
  u1;  
}
```

and try

```
? \r  
? fibonacci(100)
```

- ▶ Put the opening brace on the line after the = sign.
- ▶ End the function by a semicolon.
- ▶ Declare any local variables with `my ()`.
- ▶ Do not declare the loop index : it is local to the loop anyway.
- ▶ The function return value is the last computed value.
- ▶ Indent you code following the example.

While loop

Add

```
rho (n) =  
{  
  my (x=2, y=5) ;  
  while (gcd (y-x, n) == 1,  
    x = (x^2+1) % n ;  
    y = (y^2+1) % n ; y = (y^2+1) % n  
  ) ;  
  gcd (n, y-x) ;  
}
```

and do

```
\r  
rho (2^64+1)  
%1 = 274177
```

control flow

```
wieferich(n) =  
{  
  forprime(p=2, n,  
    if(Mod(2, p^2)^(p-1)==1,  
      return(p)));  
}  
wieferich2(n) =  
{  
  my(r);  
  forprime(p=2, n,  
    if(Mod(2, p^2)^(p-1)==1, r=p; break));  
  r;  
}  
? wieferich(10000)  
%4 = 1093
```

Constructors

```
? V=vector(10,i,1/i)
%1 = [1,1/2,1/3,1/4,1/5,1/6,1/7,1/8,1/9,1/10]
? [1/i|i<-[1..10]]
%2 = [1,1/2,1/3,1/4,1/5,1/6,1/7,1/8,1/9,1/10]
? M=matrix(4,4,i,j,i*j)
%3 = [1,2,3,4;2,4,6,8;3,6,9,12;4,8,12,16]
```

The variables i and j are local and must not be declared.

forvec

Instead of

```
s3(n) =  
{  
  my(m=sqrtint(n));  
  for(i=1,m,  
    for(j=1,m,  
      for(k=1,m,  
        if(i^2+j^2+k^2==n,  
          return([i,j,k]))));  
}
```

? s3(12345)
%2 = [4, 77, 80]

forvec

use forvec

```
s3(n) =  
{  
  my(m=sqrtint(n));  
  forvec(v=vector(3,i,[1,m]),  
    if(v*v~==n,  
      return(v)));  
}  
? s3(12345)  
%2 = [4,77,80]
```

For a better algorithm, see `qfsolve`.

associative array

```
birthday(n) =  
{  
  my(M = Map());  
  for(i=1, oo,  
    my(x=random(n), j);  
    if(mapisdefined(M, x, &j),  
      return([i, j]));  
    mapput(M, x, i));  
}  
? birthday(2^30)  
%2 = [417, 383]
```