

# TRIP programming



M. Gastineau  
gastineau@imcce.fr  
Observatoire de Paris - IMCCE - CNRS  
Astronomie et Systèmes Dynamiques  
77, avenue Denfert Rochereau  
75014 PARIS

© IMCCE - CNRS

August 4, 2010

# Scripts

- TRIP provides a full programming language that enables you to write a series of TRIP statements into a file.
- TRIP has a case sensitive interpreter and can receive your commands from the keyboard.
- Or you write your program using your preferred text editor and save it with the extension `.t`.
- The script file are loaded and executed using the `include` statement.
- TRIP looks for the script files in the current directory and then in the directory specified by the variable `_path` if they are not found.
- The C comments `/* .... */` can be used in the script file.
- The single line C++ comments `// ....` can be used in the script file.
- The `@@` statement resumes the TRIP session, loads and executes the last script.

# Functions

- The `macro` statement defines a function.
- The arguments are given by value. The arguments could be modified in a function if they are given by reference using the brackets `[]` around the argument.
- A variable number of arguments is specified by `...`. They are retrieved using the array `macro_optargs`.
- The returned value is specified by the `return` statement.
- The functions are called using the operator `%`.
- Private variables could be defined inside the function using `private`.

## source

```
macro mymean[T, mysum, ...]
{
  private m;
  mysum = sum(T);
  m = mysum/size(T);
  if (size(macro_optargs)==1) then { m=m*macro_optargs [1]; };
  return m;
};
v=1,100;
m3 = %mymean[v,[s], 3];
```

# Flow control

- TRIP supports four flow control statements
  - ▶ **for** loops.
  - ▶ **while** loops.
  - ▶ **sum** loops.
  - ▶ **if-then-else** statements.
  - ▶ **switch-case-else** statements.
- The **stop** statement interrupts the execution of a loop statement.

# if-then-else

The **if-then-else** statement looks like source

```
if (condition) then
{

    statements

};
```

or

SOURCE

```
if (condition) then
{
    statements
}
else
{
    statements
};
```

- If the boolean condition is true, then the statement block following the **then** is executed. Otherwise, the execution continues in the optional **else** block.
- It is possible to combine several conditions by using **else if**. Only the statements following the first condition that is evaluated as true will be executed.

# Switch

The **switch** statements are used to perform one of several possible sets of operations. They are intended to replace nested **if** statements depending on the same operation.

## source

```
n = 5;
value1 = 1;
value2 = -1;
value3 = 2;
switch (n)
{
  case value1 : { msg "n_is_value1"; };
  case 0, value2, value3 : { msg "n_is_value2_or_value3"; };
  else { msg "n_isn't_value1/2/3_or_0"; }
};
```

## output

```
n = 5
value1 = 1
value2 = -1
value3 = 2
n isn't value1/2/3 or 0
```

Note: Unlike C, TRIP doesn't need 'break' in each case statement and it accepts strings as value of the switch-case statements.

# For

- The **for** statement repeats a definite number of times the statements.
- The **stop** statements lets you exit early from the loop. In nested loops, **stop** terminates from the innermost loop only.
- The default **step** is 1 .
- Private variables could be dened inside the loop using **private**.

## source

```
for j=1 to n {  
  for k=n to 1 step -1 {  
    A[j]= 1/(j+k)$  
  }:  
};
```

Note: Unlike C, TRIP can't use the variable *i* or *l* as a loop variable.  
In most cases, one can replace nested loops with efficient vector manipulation.

# Sum

- The **sum** statement repeats a definite number of times the statements and performs the summation of the return statement.
- The **s = sum j=... to .... { return ...\$ };** statement is equivalent to perform **s=0\$ for j=... to .... { s=s+...\$ };**
- The **stop** statements lets you exit early from the loop. In nested loops, **stop** terminates from the innermost loop only.
- The default **step** is 1 .
- Private variables could be dened inside the loop using **private**.

## source

```
s = sum k=1 to 4 {  
    n= mod(k,3)$  
    return (1+x)**n;  
};
```

## output

```
s(x) =  
4  
+ 4*x  
+ 1*x**2
```

# While

- The **while** statement repeats an indefinite number of times the statements.
- The boolean condition is evaluated before the block is executed if the condition is true. This repeats until the condition becomes false.
- The **stop** statements lets you exit early from the loop. In nested loops, **stop** terminates from the innermost loop only.
- Private variables could be dened inside the loop using **private**.

## source

```
n=5;
while (A[j]<=n) do {
  A[j] = n-A[j]$
};
```

Note: Unlike C, TRIP can't use the variable **i** or **l** as a loop variable.