EXAMPLE PROGRAM EULER'S NUMBER APPROXIMATION

1. Requirements

Design, develop and produce an Ada program that calculates an approximation, accurate to five decimal places, of Euler's number *e* (as used in the exponential function, natural logarithms, Etc.), using the infinite series:

$$e = 1 + \frac{1}{1!} + \frac{1}{2!} + \frac{1}{3!} + \frac{1}{4!} + \cdots$$

Thus the first element in the series is 1 after which each subsequent value is equivalent to the previous value divided by N-1, where N is the current element number.

Output should comprise: (a) a list of terms, (b) the number of terms and (c) the approximation.

2. Design

The series has an infinite number of terms. However, the approximation is only required to be accurate to five decimal places therefore we can ignore any term with a value of less than 0.000001. A top down analysis of the problem is given in Table 1.

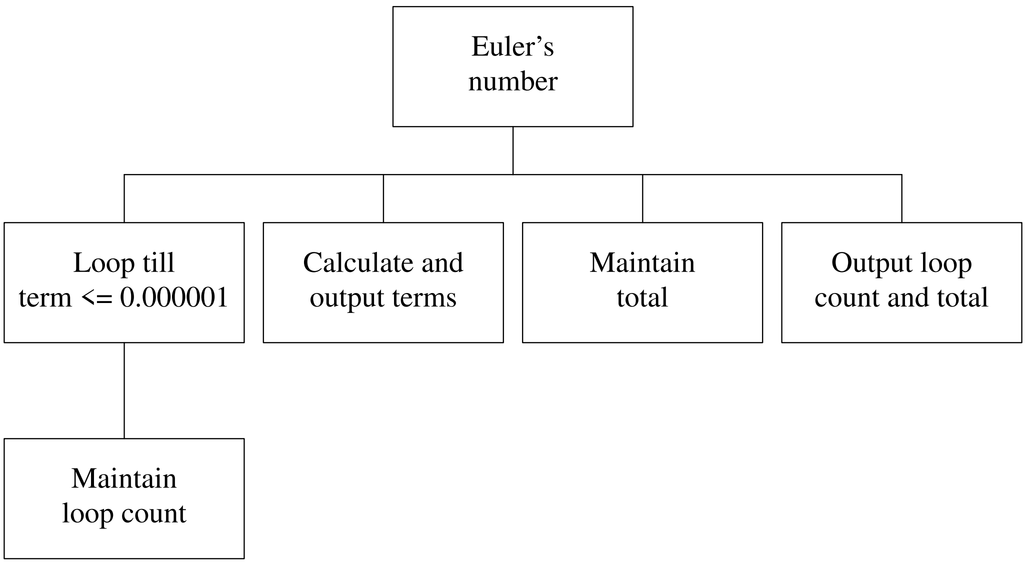

Table 1: *Top down analysis*

The operations given in Table 1 are trivial enough to be implemented as a single procedure incorporating a single fixed count loop.

1) `EULERS_NUMBER` (top level procedure): Calculate terms, maintain and output number of terms and maintain and output total.

Name	Description	Type	Range
LIMIT	Global constant	FLOAT	0.000001
TERM	Global variable	FLOAT	Default
TOTAL	Global variable	FLOAT	Default
COUNTER	Global variable	INTEGER	Defualt

An appropriate Nassi-Shneiderman chart is given in Table 2.

Set TERM and TOTAL variables to 1.0	
Set COUNTER to 1	
Loop while TERM > 0.000001	
	output TERM
	TERM := TERM/COUNTER
	Update Total (TOTAL := TOTAL+TERM)
	Increment COUNTER (COUNTER := COUNTER+1)
Output number of terms - COUNTER	
Output approximation - TOTAL	

Table 2: *Nassi-Shneiderman chart*

3. Implementation

The encoding of the above design is given in Table 3.

```
-- EULER'S NUMBER APPROXIMATION
-- 16 August 1997
-- Frans Coenen
-- Dept Computer Science, University of Liverpool

with TEXT_IO;
use TEXT_IO;

procedure EULERS_NUMBER is
    package INTEGER_INOUT is new INTEGER_IO(INTEGER);
    use INTEGER_INOUT;
    package FLOAT_INOUT is new FLOAT_IO(FLOAT);
    use FLOAT_INOUT;
    LIMIT   : constant := 0.000001;
    TERM    : FLOAT    := 1.0;
    TOTAL   : FLOAT   := 1.0;
    COUNTER : INTEGER := 1;
begin
    while TERM > LIMIT loop
        PUT(TERM, FORE=>1, AFT=>5, EXP=>0);
        TERM := TERM/FLOAT(COUNTER);
        PUT(", ");
        TOTAL   := TOTAL+TERM;
        COUNTER := COUNTER+1;
    end loop;
    NEW_LINE;
-- Output result
    PUT("Number of terms = ");
    PUT(COUNTER);
    NEW_LINE;
    PUT("Approximation = ");
    PUT(TOTAL, FORE=>1, AFT=>5, EXP=>0);
    NEW_LINE;
end EULERS_NUMBER;
```

Table 3: *Implementation*

4. Testing

The code cannot be influenced by a user therefore a simple run through will be appropriate. The result is given in Table 4.

```
kuban-158 $ eulers_number
1.00000, 0.50000, 0.16667, 0.04167, 0.00833, 0.00139,
0.00020, 0.00002, 0.00000, 0.00000,
Number of terms = 11
Approximation = 2.71828
```

Table 4: *Result of test run*