EXAMPLE PROGRAM BASE e APPROXIMATION

1. Requirements

Produce an Ada program that outputs a `12×12` table such that the intersections of coordinates represent the product of the coordinates, i.e. a `12×12` multiplication table.

2. Design

A top down analysis of the problem is given in Table 1.

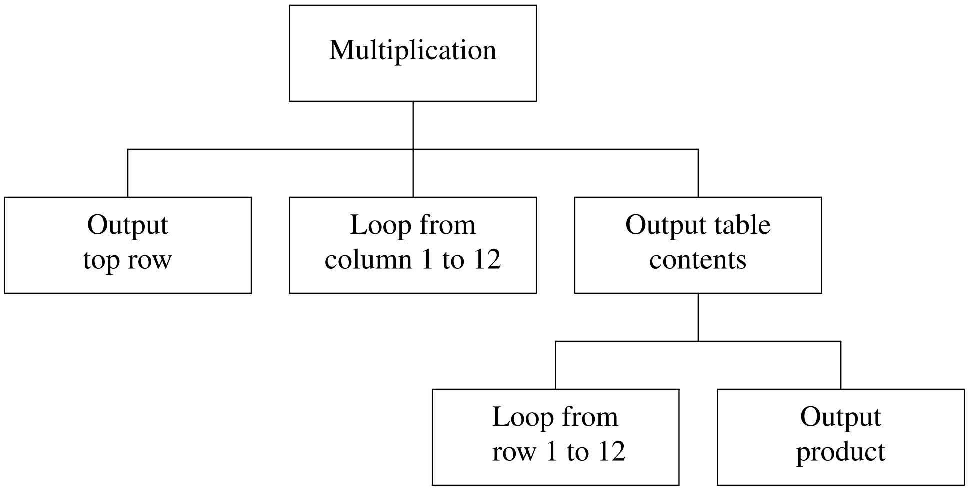


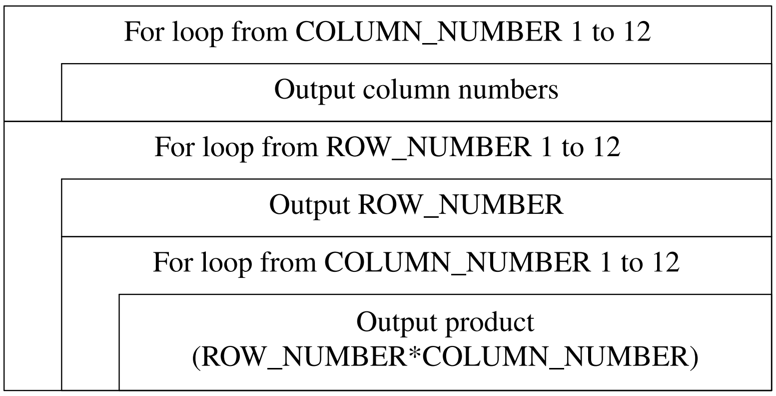
Table 1: *Top down analysis*

The operations given in Table 1 may be implemented as a single procedure incorporating a pair of nested for loops to output the table.

1) `MULTIPLICATION` (top level procedure): Two loops nested one within the other to calculate and output products.

Name	Description	Type	Range
START_VALUE	Global constant	INTEGER	1
END_VALUE	Global constant	INTEGER	12

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



MULTIPLICATION
(Top level procedure)

Table 2*: Nassi-Shneiderman chart*

3. Implementation

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

```
-- MULTIPLICATION
-- 16 September 1997
-- Frans Coenen
-- Dept Computer Science, University of Liverpool

with TEXT_IO;
use TEXT_IO;

procedure MULTIPLICATION is
    package INTEGER_INOUT is new INTEGER_IO(INTEGER);
    use INTEGER_INOUT;
    START_VALUE : constant := 1;
    END_VALUE   : constant := 12;
begin
-- Output top line of coordinates
    PUT("     |");
    for COLUMN_NUMBER in START_VALUE..END_VALUE loop
        PUT(COLUMN_NUMBER, WIDTH=>4);
    end loop;
    NEW_LINE;
    PUT_LINE("-----+------------------------------------------------");
-- Output table entries
    for ROW_NUMBER in START_VALUE..END_VALUE loop
        PUT(ROW_NUMBER, WIDTH=>4);
        PUT(" |");
        for COLUMN_NUMBER in START_VALUE..END_VALUE loop
            PUT(ROW_NUMBER*COLUMN_NUMBER, WIDTH=>4);
        end loop;
        NEW_LINE;
    end loop;
    NEW_LINE;
end MULTIPLICATION;
```

Table 3: *Implementation*

4. Testing

There are no input values or other means whereby the user can influence the operation of the program, therefore a single rum through will suffice to test the operation of the program (Table 4).

	1	2	3	4	5	6	7	8	9	10	11	12
1	1	2	3	4	5	6	7	8	9	10	11	12
2	2	4	6	8	10	12	14	16	18	20	22	24
3	3	6	9	12	15	18	21	24	27	30	33	36
4	4	8	12	16	20	24	28	32	36	40	44	48
5	5	10	15	20	25	30	35	40	45	50	55	60
6	6	12	18	24	30	36	42	48	54	60	66	72
7	7	14	21	28	35	42	49	56	63	70	77	84
8	8	16	24	32	40	48	56	64	72	80	88	96
9	9	18	27	36	45	54	63	72	81	90	99	108
10	10	20	30	40	50	60	70	80	90	100	110	120
11	11	22	33	44	55	66	77	88	99	110	121	132
12	12	24	36	48	60	72	84	96	108	120	132	144

Table 4: *Test output*.