EXAMPLE PROGRAM POWER 4 REPORT

1. Requirements

Design and create an Ada program that takes a single positive integer as input. If the input is less than 50 adds 10 and returns the result raised to the power of 4, otherwise simply return the input raised to the power of 4.

2. Design

A top down analysis is presented in Table 1.

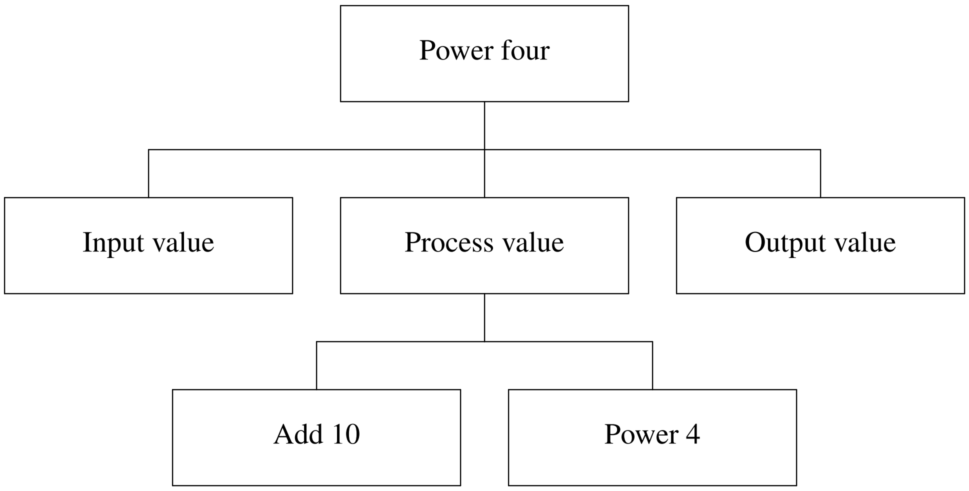

Table 1: *Top down analysis*

A single procedure:

1) `POWER_4` (top level procedure): Input data, process and output.

Name	Description	Type	Range
NUMBER	Input variable	POSITIVE	Default

The algorithm to implement this procedure is given in the Nassi-Shneiderman chart shown in Table 2. A flow chart indicating the flow of control through the proposed system is given in Table 3.

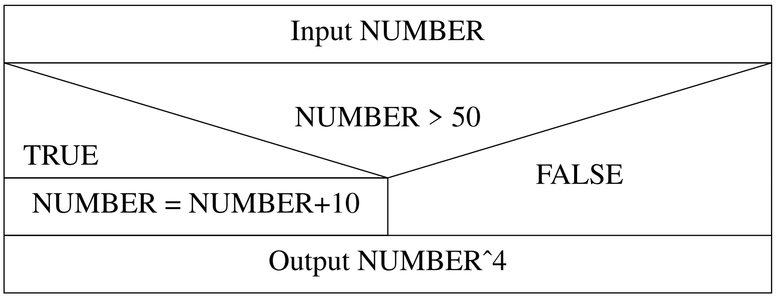

Table 2: *Nassi-Shneiderman Chart*

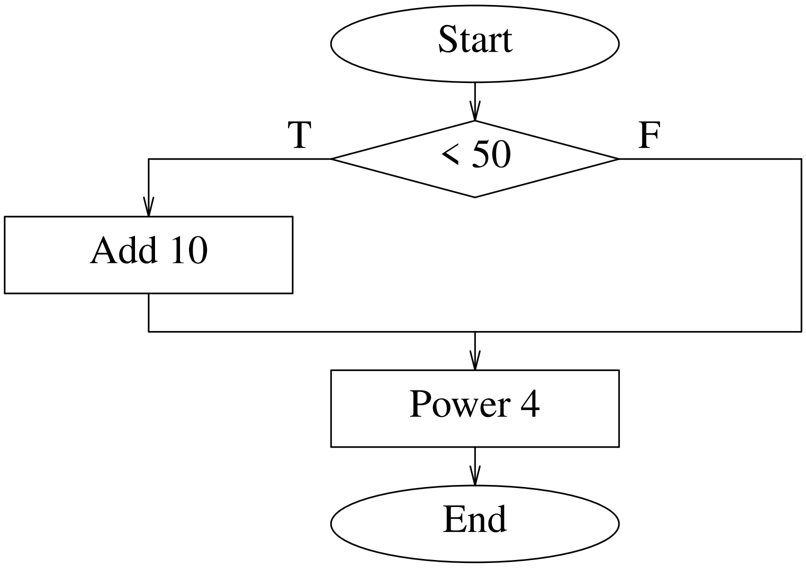

Table 3: *Flow chart*

3. Implementation

Implementation as shown in Table 4:

```
-- POWER 4
-- 6 August 1997
-- Frans Coenen
-- Dept Computer Science, University of Liverpool

with TEXT_IO;
use TEXT_IO;

procedure POWER_4 is
        package INTEGER_INOUT is new INTEGER_IO(INTEGER);
        use INTEGER_INOUT;
        NUMBER: POSITIVE;
begin
-- Read value A
        PUT_LINE("Input a value: ");
        GET(NUMBER);
-- Check for NUMBER < 50, if so add 10
        if NUMBER < 50 then
                NUMBER := NUMBER + 10;
        end if;
-- Output A^4
        PUT_LINE("Result is: ");
        PUT(NUMBER**4);
        NEW_LINE
end POWER_4;
```

Table 4: *Implementation*

4. Testing

4.1. Black box testing

4.1.1. BVA and limit testing

Use BVA and limit testing to analyse and test limits of input values. Suitable test cases are presented in Table 5. Results are presented in Table 6, note incorrect answers produced for last two tests - the maximum value for the integer type used has been exceeded. A return to the requirements is therefore indicated.

TEST CASE	EXPECTED RESULT
NUMBER	OUTPUT
0	CONSTRAINT ERROR
1	14641
2	20736
2147483646	2.126764785333e+37
2147483647	2.126764789294e+37

Table 5: *BVA and limit test cases*

```
kuban-214 $ power_4
Input value for A:
0
Ada-runtime: Exception CONSTRAINT_ERROR raised in
/tmp_mnt/home/staff2/ra/frans/public_html/2CS21/week5/power_4.ada
on line 11. Rangecheck error: 0 not in range 1..2147483647.

kuban-215 $ power_4
Input value for A:
1
Result is:
      14641

kuban-216 $ power_4
Input value for A:
2
Result is:
      20736

kuban-217 $ power_4
Input value for A:
2147483646
Result is:
         16

kuban-218 $ power_4
Input value for A:
2147483647
Result is:
          1
```

Table 6: *BVA and limit test case output*

4.1.2. Arithmetic testing

To test the arithmetic operation of the software we should include some arithmetic testing. This dictates a test case with a positive sample value (zero and negative values are precluded). We have already devised a test cases incorporating positive input values, however these were all at the limits of the input range. A test case which uses a "middle of the range" value might therefore be appropriate here (Table 7). (Result in Table 8).

TEST CASE NUMBER	EXPECTED RESULT OUTPUT
215	2136750625

Table 7: *Arithmetic test cases*

```
kuban-219 $ power_4
Input value for A:
215
Result is:
 2136750625
```

Table 8: *Arithmetic test case output*

4.2. White box testing

4.2.1. Path testing

Test both paths through the procedure one where input is less than 50 and one where it is equal to or greater than 50. Two appropriate test cases are given Table .Two appropriate test cases are given in the Table 9 (results in Table 10).

TEST CASE NUMBER	EXPECTED RESULT OUTPUT
49	12117361
50	6250000

Table 9: *Path test cases*

```
kuban-221 $ power_4
Input value for A:
49
Result is:
   12117361

kuban-222 $ power_4
Input value for A:
50
Result is:
    6250000
```

Table 10: *Path test case output*

4.3. Data validation testing

Table 11 shows the results produced when a number of data validation tests are run.

```
kuban-225 $ power_4
Input value for A:
23.5
Result is:
    1185921

kuban-226 $ 5
ksh: 5:  not found

kuban-228 $ power_4
Input value for A:
frans
Ada-runtime: Exception DATA_ERROR propagated out of main.

kuban-229 $ rans
ksh: rans:  not found

kuban-230 $ power_4
Input value for A:
2*4
Ada-runtime: Exception DATA_ERROR propagated out of main.

kuban-232 $ 4
ksh: 4:  not found

kuban-233 $ power_4
Input value for A:
2f5
Ada-runtime: Exception DATA_ERROR propagated out of main.

kuban-234 $ 5
ksh: 5:  not found
```

Table 11: *Data validation test case output*