EXAMPLE PROGRAM VERTICAL MOTION REPORT

1. Requirements

Given the start velocity of a ball which is thrown vertically upwards from ground level, design and implement an Ada program that determines:

1) The maximum height reached, and
2) The time taken to return to the ground.

Assume that the start velocity is an integer between 1 and 100m/s (inclusive) and that deceleration due to gravity is equivalent -10 m/s^2. Use the equation:

$$h = \frac{u^2}{-2 \times g}$$

where `u` = start velocity and `g` = deceleration due to gravity, to calculate the maximum height (`h`). Use the equation:

$$t = \frac{4 \times h}{u}$$

to calculate total time (`t`) taken to return to the ground. Output should be accurate to two decimal places.

2. Design

A top down analysis is presented in Table 1.

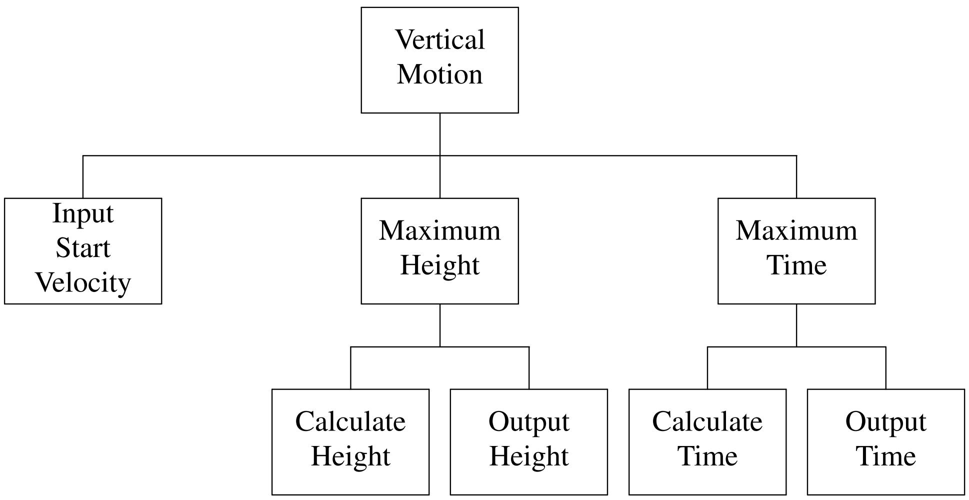

Table 1: *Top down analysis*

The operations identified in the above top level analysis can be combined into the following three procedures/functions:

1) **VERT_MOTION** (top level procedure).

NAME	DESCRIPTION	TYPE	VALUE/RANGE
START_VELOCITY	Input global variable	VELOCITY	1..100
DECELERATION	Global constant	FLOAT	-10.0
MAX_HEIGHT	Global variable	FLOAT	Default

2) **MAXIMUM_HEIGHT** (level 2 function): Calculate, output and return maximum height of projectile.

NAME	DESCRIPTION	TYPE	VALUE/RANGE
START_VELOCITY	Formal parameter	VELOCITY	1..100
MAX_HEIGHT	Local variable	FLOAT	Default

3) **TOTAL_TIME** (level 2 function): Calculate and output total time.

NAME	DESCRIPTION	TYPE	VALUE/RANGE
START_VELOCITY	Formal parameter	VELOCITY	1..100
MAX_HEIGHT	Formal parameter	FLOAT	Default

The design detail for these procedures/functions is expressed by the set of Nassi-Shneiderman charts given in Table 2.

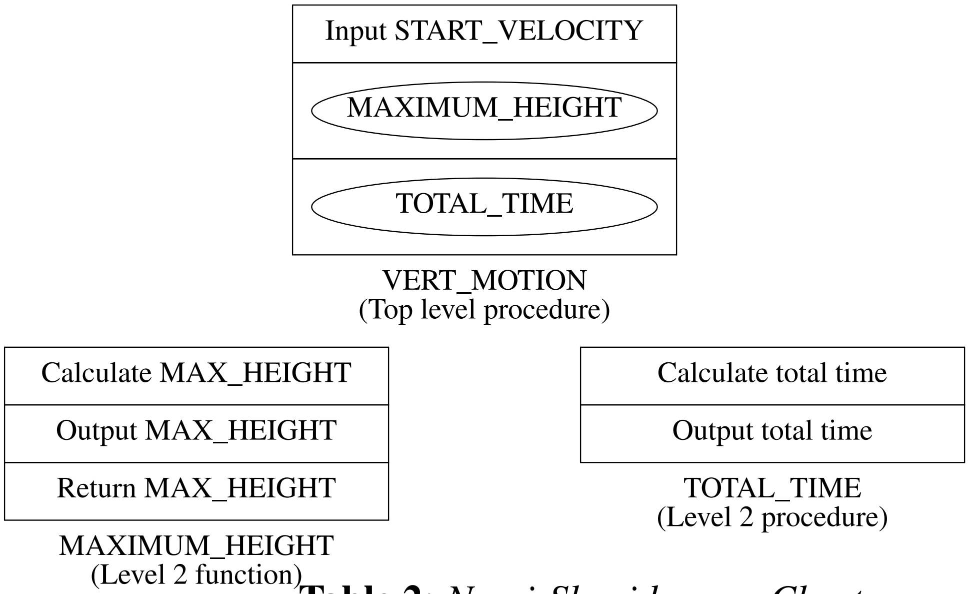

Table 2: *Nassi-Shneiderman Chart*

3. Implementation

Implementation is as shown in Table 3:

```
-- VERTICAL MOTION
-- 9 September 1997
-- Frans Coenen
-- Dept Computer Science, University of Liverpool

with TEXT_IO;
use TEXT_IO;

procedure VERT_MOTION is
    subtype VELOCITY is INTEGER range 1..100;
    package INTEGER_INOUT is new INTEGER_IO(INTEGER);
    use INTEGER_INOUT;
    package FLOAT_INOUT is new FLOAT_IO(FLOAT);
    use FLOAT_INOUT;
    START_VELOCITY: VELOCITY;
    DECELERATION: CONSTANT FLOAT := -10.0;
    MAX_HEIGHT: FLOAT;

    ------------------------------------------------------------------------
    -- CALCULATE MAXIMUM HEIGHT
    function MAXIMUM_HEIGHT(START_VELOCITY: VELOCITY) return FLOAT is
        MAX_HEIGHT: FLOAT;
    begin
        MAX_HEIGHT := (FLOAT(START_VELOCITY)**2)/(-2.0*DECELERATION);
        PUT("Maximum height reached = ");
        PUT(MAX_HEIGHT, FORE=>4, AFT=>2, EXP=>0);
        PUT(" m");
        NEW_LINE;
        RETURN(MAX_HEIGHT);
    end MAXIMUM_HEIGHT;
    ------------------------------------------------------------------------
    -- CALCULATE TOTAL TIME
    procedure TOTAL_TIME(START_VELOCITY: VELOCITY; MAX_HEIGHT: FLOAT) is
    begin
        PUT("Total time to return to ground = ");
        PUT(4.0*MAX_HEIGHT/FLOAT(START_VELOCITY), FORE=>4, AFT=>2, EXP=>0);
        PUT(" s");
        NEW_LINE;
    end TOTAL_TIME;
    ------------------------------------------------------------------------

-- TOP LEVEL
begin
-- Input start velocity
    PUT_LINE("Input start velocity (integer value between 1 and 100): ");
    GET(START_VELOCITY);
-- Calculate maximum height;
    MAX_HEIGHT := MAXIMUM_HEIGHT(START_VELOCITY);
-- Calculate total time;
    TOTAL_TIME(START_VELOCITY, MAX_HEIGHT);
end VERT_MOTION;
```

Table 3: *Implementation*

4. Testing

4.1. BVA and limit testing

Use BVA (Boundary Value Analysis) and limit testing to analyse and test limits of input variables. Suitable values for START_VELOCITY are 0, 1, 2 99, 100 and 101. An appropriate set of test cases is given in Table 4.

TEST CASE	EXPECTED RESULT	
START_VELOCITY	MAX_HEIGHT	Total time
0	CONSTRAINT_ERROR	
1	0.05	0.20
2	0.20	0.40
99	490.05	19.80
100	500.00	20.00
101	CONSTRAINT_ERROR	

Table 4: *BVA and limit test cases*

A screen dump illustrating the output produced when the above test cases are run is given in Table 5.

```
kuban-375 $ vert_motion
Input start velocity (integer value between 1 and 100):
0
Ada-runtime: Exception CONSTRAINT_ERROR raised in
/tmp_mnt/home/staff2/ra/frans/public_html/2CS21/week5/vert_motion.ada
on line 49. Rangecheck error: 0 not in range 1..100.

kuban-376 $ vert_motion
Input start velocity (integer value between 1 and 100):
1
Maximum height reached =    0.05 m
Total time to return to ground =    0.20 s

kuban-377 $ vert_motion
Input start velocity (integer value between 1 and 100):
2
Maximum height reached =    0.20 m
Total time to return to ground =    0.40 s

kuban-378 $ vert_motion
Input start velocity (integer value between 1 and 100):
99
Maximum height reached =  490.05 m
Total time to return to ground =   19.80 s
```

Table 5: *BVA and limit test output*

```
kuban-379 $ vert_motion
Input start velocity (integer value between 1 and 100):
100
Maximum height reached =  500.00 m
Total time to return to ground =   20.00 s

kuban-380 $ vert_motion
Input start velocity (integer value between 1 and 100):
101
Ada-runtime: Exception CONSTRAINT_ERROR raised in
/tmp_mnt/home/staff2/ra/frans/public_html/2CS21/week5/vert_motion.ada
on line 49. Rangecheck error: 101 not in range 1..100.
```

Table 5 continued

4.2. Arithmetic testing

Use arithmetic testing techniques to test the arithmetic expressions. Test using positive values only in this case (negative and zero values precluded by input type definition in this case). Positive value already included in BVA and limit test cases, however, we should include a test somewhere in the middle of the range (Table 6).

TEST CASE	EXPECTED RESULT	
START_VELOCITY	MAX_HEIGHT	Total time
50	125.00	10.00

Table 6: *Arithmetic test cases*

The result of running the above test is given in table 7.

```
kuban-375 $ vert_motion
Input start velocity (integer value between 1 and 100):
50
Maximum height reached =  125.00 m
Total time to return to ground =   10.00 s
```

Table 7: *Arithmetic test output*

4.3. Data validation testing

Results produced by a number of data validation test cases are given in Table 8.

```
kuban-245 $ vert_motion
Input start velocity (integer value between 1 and 100):
10.0
Maximum height reached =    5.00 m
Total time to return to ground =    2.00 s

kuban-246 $ 0
ksh: 0:  not found

kuban-247 $ vert_motion
Input start velocity (integer value between 1 and 100):
1e.8
Ada-runtime: Exception DATA_ERROR propagated out of main.

kuban-248 $ 8
ksh: 8:  not found
```

Table 8: *Data validation testing output*