e) Produce a table which labels columns by heights from 1.55m to 1.95m in steps of 5 cm and rows by weights from 50kg to 95kg in steps of 5kg. At each intersection compute the corresponding body mass index. Write then a UDF which uses this table as a Vlookup table to determine the body mass index from a given height and weight.

- The table should look like:

(Use the autofill function to produce it. Only type row 5.)

	A	B	C	D	E	F	G	H	I	J
2										
3										
4		1.55	1.6	1.65	1.7	1.75	1.8	1.85	1.9	1.95
5	50	20.8	19.5	18.4	17.3	16.3	15.4	14.6	13.9	13.1
6	55	22.9	21.5	20.2	19	18	17	16.1	15.2	14.5
7	60	25	23.4	22	20.8	19.6	18.5	17.5	16.6	15.8
8	65	27.1	25.4	23.9	22.5	21.2	20.1	19	18	17.1
9	70	29.1	27.3	25.7	24.2	22.9	21.6	20.5	19.4	18.4
10	75	31.2	29.3	27.5	26	24.5	23.1	21.9	20.8	19.7
11	80	33.3	31.2	29.4	27.7	26.1	24.7	23.4	22.2	21
12	85	35.4	33.2	31.2	29.4	27.8	26.2	24.8	23.5	22.4
13	90	37.5	35.2	33.1	31.1	29.4	27.8	26.3	24.9	23.7
14	95	39.5	37.1	34.9	32.9	31	29.3	27.8	26.3	25

e.g. D6 contains =bmi($A6,$D$4)

77

```
Function bmitable(weight As Single, height As Single) As Single
   Dim x As Integer
   If height >= 1.55 Then x = 2
   If height >= 1.6 Then x = 3
   If height >= 1.65 Then x = 4
   If height >= 1.7 Then x = 5
   If height >= 1.75 Then x = 6
   If height >= 1.8 Then x = 7
   If height >= 1.85 Then x = 8
   If height >= 1.9 Then x = 9
   If height >= 1.95 Then x = 10
   bmitable = WorksheetFunction.VLookup(weight, [a5:j14], x)
End Function
```

This function gives wrong results for heights >2.00 and heights <1.55.

78

f) Produce two tables which label columns and rows in the same way as in e). At each intersection compute the meaning for the body mass index for male and female in the tables. Write then a UDF which uses either of these tables as a Vlookup table to determine the meaning of the body mass index from a gives height, weight and gender.

- The tables (part of them) should look like:

29	female	1.55	1.6	1.65	1.7	1.75	
30	50	normal weight	normal weight	underweight	underweight	underweight	u
31	55	normal weight	normal weight	normal weight	normal weight	underweight	u
32	60	overweight	normal weight	normal weight	normal weight	normal weight	u
33	65	overweight	overweight	normal weight	normal weight	normal weight	n
34	70	obese	overweight	overweight	overweight	normal weight	n
35	75	obese	obese	overweight	overweight	overweight	n
36	80	obese	obese	obese	overweight	overweight	o

You can either compute the body mass index or use the table from e) to look up the values.

In the latter case D32 contains =bmimean($D7,"female")

79

```
Function BT(weight As Single, height As Single, mf As String) As String
   Dim x As Integer
   If height >= 1.55 Then x = 2
   If height >= 1.6 Then x = 3
   If height >= 1.65 Then x = 4
   If height >= 1.7 Then x = 5
   .....
   If height >= 1.95 Then x = 10
   If mf = "female" Then
     BT = WorksheetFunction.VLookup(weight, [a30:j39], x)
   ElseIf mf = "male" Then
     BT = WorksheetFunction.VLookup(weight, [a18:j27], x)
   Else
     BT = "Specify gender!"
  End If
End Function
```

Decide here which table to take:

80

▸ The SELECT CASE-structure

• The SELECT CASE structure is another branching structure provided by VBA. It is a more elegant and transparent version of an IF-structure, which tests always the same variable.

For instance (determine the sign of a number):

```
Function sig(x As Single) As String
    If x > 0 Then
      sig = "positive"
    ElseIf x < 0 Then
      sig = "negative"
    Else
      sig = "zero"
    End If
End Function
```

=

```
Select Case x
    Case Is > 0: sig = "positive"
    Case Is < 0: sig = "negative"
    Case Else:   sig = "zero"
End Select
```

81

Syntax: **Select Case** testvariable

[**Case** expressionlist
[statements]]... } can be repeated many times

[**Case Else**
[elsestatements]]

End Select

- testvariable ≡ a numeric or string expression
- expressionlist ≡ list of one or more expressions separated by a comma
  - · expression
  - · expression **To** expression
  - · **Is** comparisonoperator expression
- statements ≡ executed when one condition from expressionlist is true
- elsestatements ≡ executed when no previous condition is true

82

• Examples (Select case):

a) Function si(x)

```
Select Case x
  Case 0:      si = 1
  Case Else: si = Sin(x) / x
End Select
End Function
```

$$Si(x) = \begin{cases} \frac{\sin x}{x} & \text{for } x \in \mathbb{R}\backslash 0 \\ 1 & \text{for } x = 0 \end{cases}$$

b) Function F(x As Single) As Single

```
Select Case x
  Case Is < 0:   F = 0
  Case 0 To 4:   F = 3 * x
  Case Else:     F = 12
End Select
End Function
```

$$F(x) = \begin{cases} 0 & \text{for } x < 0 \\ 3x & \text{for } 0 \le x \le 4 \\ 12 & \text{for } x > 4 \end{cases}$$

· Note that "a To b" means "a ≤ x ≤ b"

83

c) Function G(x As Single) As Single

```
Select Case x
  Case -4 To 4:   G = 1
  Case Else:      G = 0
End Select
End Function
```

$$G(x) = \begin{cases} 1 & \text{for } -4 \le x \le 4 \\ 0 & \text{otherwise} \end{cases}$$

· Note that "a To b" means "a ≤ x ≤ b"

d) Function entry(age As Integer) As Variant

```
Select Case age
  Case 0 To 5, Is > 65:  entry = 0
  Case 6 To 15:          entry = 2
  Case 15 To 65:         entry = 5
  Case Else:             entry = "Age not valid!"
End Select
End Function
```

84

e) Function price(product As String) As Variant

```
    Select Case product
      Case "Mangoes":          price = 2.5
      Case "Bananas":          price = 1.8
      Case "Pears", "Apples":  price = 0.9
      Case Else:               price = "Fruit not in price list!"
    End Select
End Function
```

· Note that the test variable can also be of string type

· Note that price is of type Variant, as it could be a number or a string

· Note that the test is case sensitive, e.g.
=price("mangoes") → "Fruit not in price list!"

· Note that when the "Case Else" line is dropped
=price("Papayas") → 0

85

f) Function pricec(product As String, country As String) As Variant

```
    Select Case country
      Case "Brasil"
        Select Case product
          Case "Mangoes" , "Papayas":  pricec = 2.5
          Case "Bananas":  pricec = 1.3
          Case Else: pricec = "Fruit not in the list!"
        End Select
      Case "Thailand"
        Select Case product
          Case "Mangoes": pricec = 2.2
          Case "Papayas":  pricec = 2.8
          Case Else:  pricec = "Fruit not in the list!"
        End Select
      Case Else: pricec = "Country not the list!"
    End Select
End Function
```

· One can also nest the SELECT structure similar to the IF-structure

86