MODULE 8: CONDITIONS

On completion of this module you will be able to teach the program to make decisions and which options to select from, best for the program and output.

Subject Outcome 1: Introduction

Subject Outcome 2: If ... Then ...

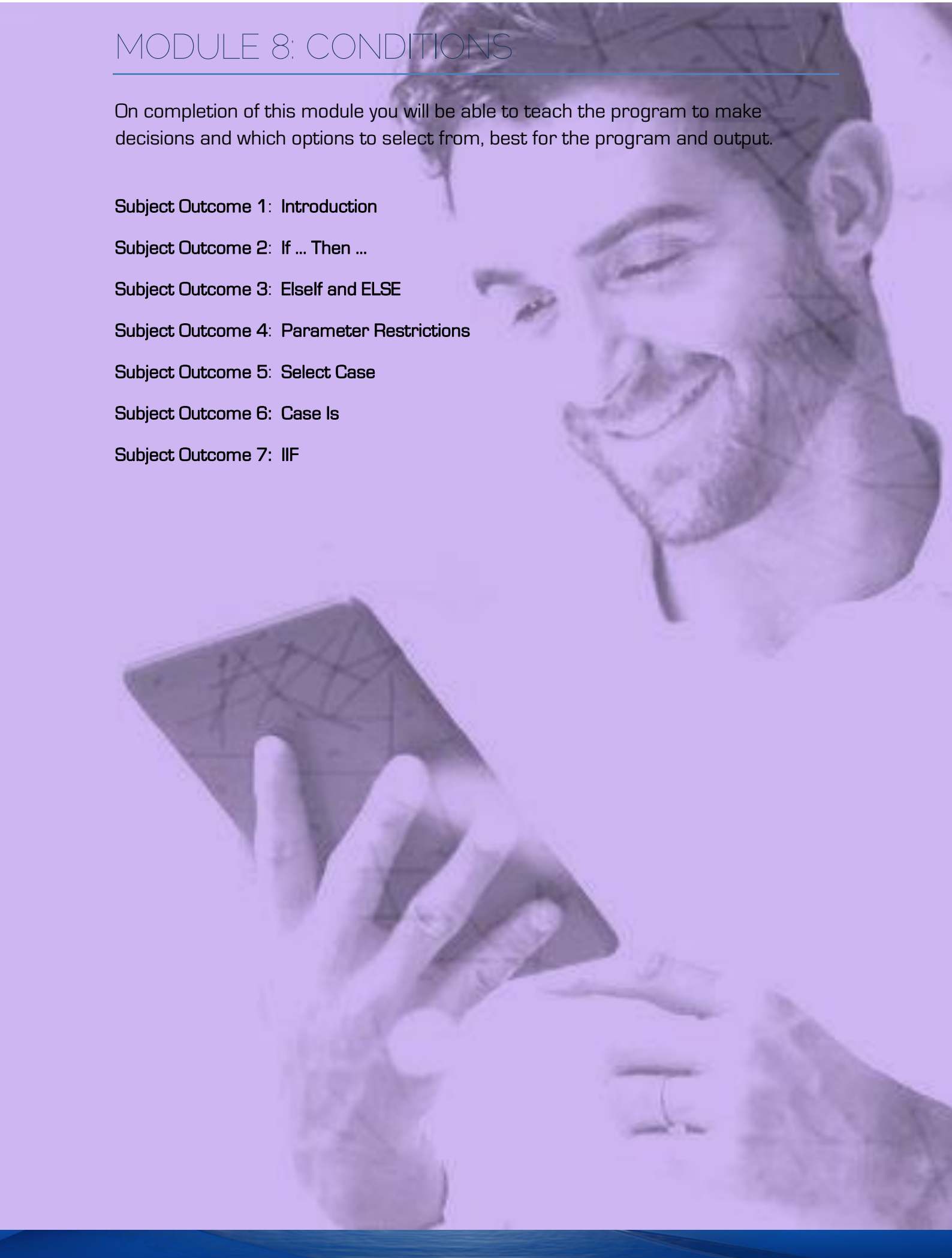
Subject Outcome 3: ElseIf and ELSE

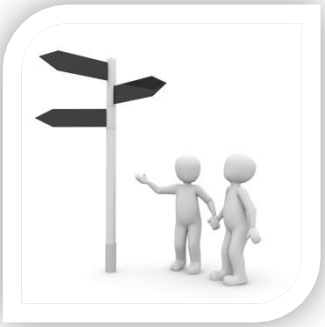
Subject Outcome 4: Parameter Restrictions

Subject Outcome 5: Select Case

Subject Outcome 6: Case Is

Subject Outcome 7: IIF





8.1 CONDITIONS

8.1.1 Introduction

Conditional statements offer the ability to execute code for which the condition is satisfied; may it be true or false; an amount less, equal or more; a specific string or portion of a string. **Conditions guide the program in a direction depending** on **data being processed** or **user input**. We have already touched condition numerous times, remember all the **IF** .. **THEN** commands, that is one type of condition.

8.1.2 IF ... THEN

This command verifies two or more values. You will use logical and multiple condition "checkers" – *logical* being; **= < > <= >=** and *multiple conditions checkers*; **or, and.**

Once the condition has been evaluated, you may code the executional part of the condition within a single line or multi line. If you use a multi-line then you need to end with **END IF**.

Syntax (single line executional):

Commands	Description
if edit1.text="andre" then showmessage "hallo"	This will **display a message** if the text entered within **EDIT1 edit** field was **andre**.

Syntax (multi-line executional):

Commands	Description

If edit1.text="andre" then
Showmessage "hallo"
Edit1.color=255
Edit1.text=""
End if
	Confirm if the condition field is "andre" and if so ...
	Display a message
	Alter the color of the EDIT1 field.
	Clear the EDIT1 field.
	End the condition.

Syntax (multi-line executional – condition numerical values):

Commands	Description

If val(edit1.text) < 90 then
Showmessage "hallo"
Edit1.color=255
Edit1.text=""
End if
	Confirm if the condition field is less than 90.
	Display a message
	Alter the color of the EDIT1 field.
	Clear the EDIT1 field.
	End the condition.

Syntax (multi-line executional – condition numerical values with multiple conditions):

Commands	Description

If val(edit1.text) < 90 and val(edit1.text) > 40 then
 	
Showmessage "hallo"
Edit1.color=255
Edit1.text=""
End if
	Confirm if the condition field is less than 90, but larger than 40, then ...
	Display a message
	Alter the color of the EDIT1 field.
	Clear the EDIT1 field.
	End the condition.

8.1.3 ELSEIF and ELSE

This command is used as alternatives to the first possible answer.

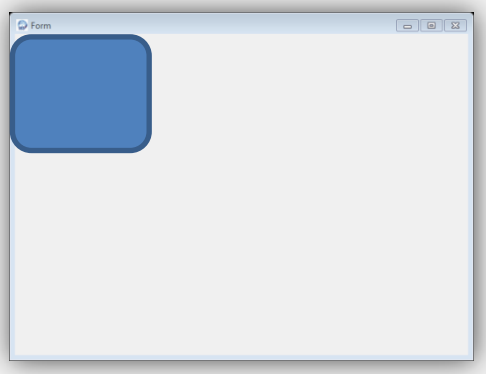
Commands	Description
$TYPECHECK ON $INCLUDE <RapidQ2.inc> declare sub logonnow CREATE Form AS QFORM Caption = "Form":Width = 640 Height = 480:Center create nameinput as qedit left=100:top=50:width=200 showhint=1:hint="enter password or GUEST to visit ..." end create create logonbtn as qbutton left=100:top=80:caption="LOG ON" onclick=logonnow end create END CREATE SetWindowLong(Form.Handle, -8, 0) SetWindowLong(Application.Handle, -8, Form.Handle) Form.ShowModal	

sub logonnow
if nameinput.text="hallo" then
showmessage "correct password"
elseif lcase$(nameinput.text)="guest" then
showmessage "view only"
else
showmessage "wrong password"
end if
end sub
	Condition 1; if edit field's content is **hallo** then correct.
	
	Condition 2; if edit field's content is **guest**, then ...
	
	Condition 3; all other **content** is wrong.

8.1.4 Parameter Restrictions

This is when a result must be between values, etc. During the next example we will call a sub program when the mouse cursor is within a certain position related to the window.

Commands	Description

$TYPECHECK ON
$INCLUDE <RapidQ2.inc>
declare sub checkmousepos(x%,y%,shift%)
CREATE Form AS QFORM
Caption = "Form":Width = 640
Height = 480:Center
onmousemove=checkmousepos
END CREATE
SetWindowLong(Form.Handle, -8, 0)
SetWindowLong(Application.Handle, -8, Form.Handle)
Form.ShowModal
 	
sub checkmousepos(x%,y%,shift%)
if x%>20 and x%<150 then
if y%>20 and y%<150 then
showmessage "got there."
end
end if
end if
end sub
	Move the mouse cursor as indicated below onto the blue boxed area.
	 
	If the cursor is within block 20-150 (horizontally) and 20-150 Vertical, then.

8.1.5 Select Case

Select Case is usually implemented when you have a lot of conditions that you want to test for (mostly numerical tests).

Commands	Description

$TYPECHECK ON
$INCLUDE <RapidQ2.inc>
declare sub ageinputnow(key as byte)
CREATE Form AS QFORM
Caption = "Form":Width = 640: Height = 480:Center
create ageinput as qedit
left=50:top=50
showhint=1:hint="enter age ... press enter"
onkeyup=ageinputnow
end create
END CREATE
SetWindowLong(Form.Handle, -8, 0)
SetWindowLong(Application.Handle, -8, Form.Handle)
Form.ShowModal
sub ageinputnow(key as byte)
if key=13 then
select case val(ageinput.text)
case 1 to 10
showmessage "child"
case 11 to 30
showmessage "youngster: "+ageinput.text
case 31 to 40
showmessage "parent: "+ageinput.text
case 41 to 100
showmessage "grandpa: "+ageinput.text
case 101
showmessage "really 101?
end select
end if
end sub
	Create sub program for when the user presses ENTER to confirm age.
	
	Create edit field for user input (age).
	
	Associate the numerical value of the edit field to SELECT CASE.
	If the value is between 1 and 10 then ...
	If the value is between 11 and 30 then...
	
	If the value is between 31and 40 then...
	
	If the value is between 41 and 100 then ...
	
	If the value is 101 then ...

8.1.6 Case Is

This is **part** of **Select Case**. It will open more logistical evaluations such as **>= <=**

Commands	Description

$TYPECHECK ON
$INCLUDE <RapidQ2.inc>
declare sub ageinputnow(key as byte)
CREATE Form AS QFORM
  Caption = "Form":Width = 640
  Height = 480:Center
  create ageinput as qedit
  left=50:top=50
  showhint=1:hint="enter age ... press enter"
  onkeyup=ageinputnow
  end create
END CREATE
SetWindowLong(Form.Handle, -8, 0)
SetWindowLong(Application.Handle, -8, Form.Handle)
Form.ShowModal
 	
sub ageinputnow(key as byte)
if key=13 then
  select case val(ageinput.text)
  case 1 to 10
  showmessage "child"
  case 11 to 30
  showmessage "youngster: "+ageinput.text
  case 31 to 40
  showmessage "parent: "+ageinput.text
  case 41 to 100
  showmessage "grandpa: "+ageinput.text
  case is >= 101
  showmessage "you are blessed"
  end select
end if
end sub
	If the result is greater or equal to 101 then ...

8.1.7 IIF

If you are familiar with MS Excel ® then you will know this format. It is taken from the same format as **Visual Basic ® coding**. **The command line is**:

Result$= iff(?=??,"#","##")

It reads as follow:

- The result will be holding the value of the possible outcome from the **IIF**.
- The **?** represents the first value to be confirmed.
- The **??** represents the second value to be confirmed.
- **#** represents the answer if the condition is true.
- **##** represents the answer if the condition is false.

Commands	Description
$TYPECHECK ON $INCLUDE <RapidQ2.inc> dim result$ as string declare sub ageinputnow(key as byte) CREATE Form AS QFORM Caption = "Form":Width = 640 Height = 480:Center create ageinput as qedit left=50:top=50 showhint=1:hint="enter age ... press enter" onkeyup=ageinputnow end create END CREATE SetWindowLong(Form.Handle, -8, 0) SetWindowLong(Application.Handle, -8, Form.Handle) Form.ShowModal	Establish the result$ string to capture the result from the condition.

sub ageinputnow(key as byte)
if key=13 then
result$=iif(val(ageinput.text)>10,"more","less")
showmessage result$
end if
end sub
	If the ENTER key (13) has been pressed, then ...
	If AGEINPUT is more than 10, then result$="more"
	If AGEINPUT is less than 10, then result$-="Less"
	Display the result.