Chapter 4 Recap - continued

===========================

Please note that the following are mainly points with some brief explanations to add to the notes you are compiling. The recap is an opportunity to revisit the work we have spoken about during class.
* Please note that the following recap and associated notes are compiled from multiple sources.

==

Overview

==

Chapter highlights:
-- the case structure

==

----

still if..then statements

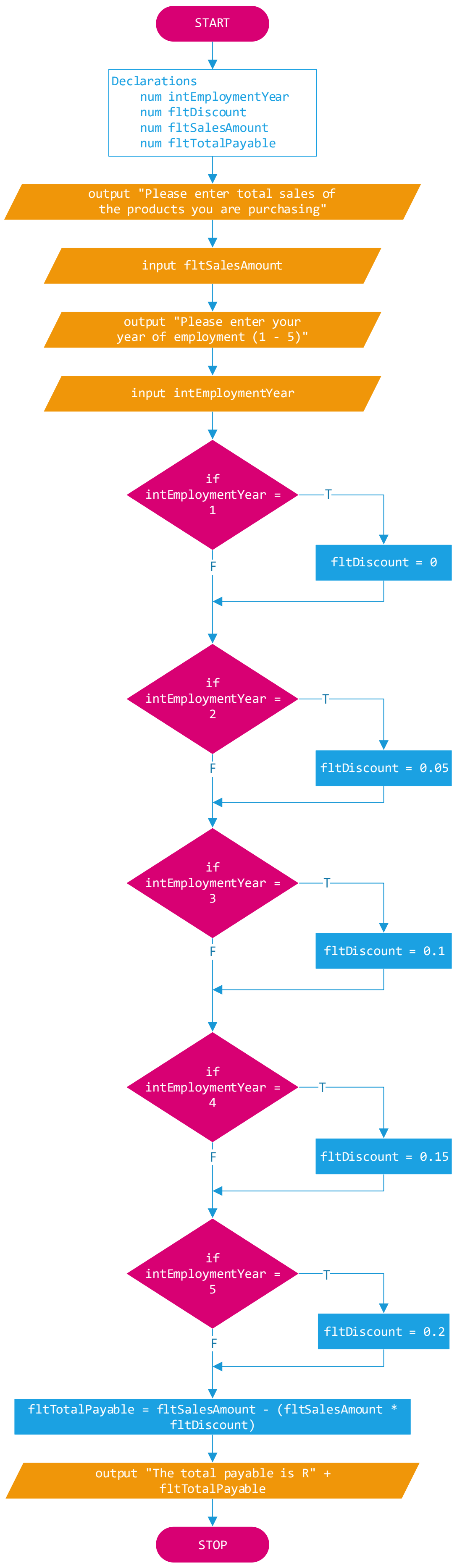
==

-- Consider the following: Employees working for a major fashion retailer are allowed to purchase from the store. In their first year of employment, there are no discounts offered, but once a year of employment is completed, then the discount applies. Currently the store is testing out this feature for their employees and they are only applying the discount for employees who have been working between 2 and 5 years. *(Once more data is provided, the company will expect this program to be updated)*.

Year	**Discount on Sales Amount**
1	0%
2	5%
3	10%
4	15%
5	20%

** pseudocode

```
0. Start
1. Declarations
        num intEmploymentYear
        num fltDiscount
        num fltSalesAmount
        num fltTotalPayable

2.      output "Please enter total sales of the products you are purchasing"
3.      input fltSalesAmount

4.      output "Please enter your year of employment (1 - 5)"
5.      input intEmploymentYear

6.      if intEmploymentYear = 1 then
                fltDiscount = 0
        endif

7.      if intEmploymentYear = 2 then
                fltDiscount = 0.05
        endif

8.      if intEmploymentYear = 3 then
                fltDiscount = 0.1
        endif

9.      if intEmploymentYear = 4 then
                fltDiscount = 0.15
        endif

10.     if intEmploymentYear = 5 then
                fltDiscount = 0.2
        endif

11.     fltTotalPayable = fltSalesAmount - (fltSalesAmount * fltDiscount)

12.     output "The total payable is R" + fltTotalPayable
13. Stop
```


START
Declarations
num intEmploymentYear
num fltDiscount
num fltSalesAmount
num fltTotalPayable
output "Please enter total sales of the products you are purchasing"
input fltSalesAmount
output "Please enter your year of employment (1 - 5)"
input intEmploymentYear
if intEmploymentYear = 1
T
fltDiscount = 0
F
if intEmploymentYear = 2
T
fltDiscount = 0.05
F
if intEmploymentYear = 3
T
fltDiscount = 0.1
F
if intEmploymentYear = 4
T
fltDiscount = 0.15
F
if intEmploymentYear = 5
T
fltDiscount = 0.2
F
fltTotalPayable = fltSalesAmount - (fltSalesAmount * fltDiscount)
output "The total payable is R" + fltTotalPayable
STOP

- Now we have dealt with this type of solution above by using straight-through logic, coding sequential if..then statements (each if..then statement is on its own). While it is acceptable, the solution can be made a little more efficient with nested if..then statements (using positive or negative logic | The following example code uses positive logic)

```
0. Start
1. Declarations
        num intEmploymentYear
        num fltDiscount
        num fltSalesAmount
        num fltTotalPayable

2.      output "Please enter total sales of the products you are purchasing"
3.      input fltSalesAmount

4.      output "Please enter your year of employment (1 - 5)"
5.      input intEmploymentYear

6.      if intEmploymentYear = 1 then
                fltDiscount = 0
        else
                if intEmploymentYear = 2 then
                        fltDiscount = 0.05
                else
                        if intEmploymentYear = 3 then
                                fltDiscount = 0.1
                        else
                                if intEmploymentYear = 4 then
                                        fltDiscount = 0.15
                                else
                                        fltDiscount = 0.2
                                endif
                        endif
                endif
        endif

7.      fltTotalPayable = fltSalesAmount - (fltSalesAmount * fltDiscount)

8.      output "The total payable is R" + fltTotalPayable
9. Stop
```

- While the above solution is now a little more efficient, and we eliminated the last if..then statement (rule while developing positive logic, nested if..then statements), there is still more that could be done.

- *Take note of the new statement in the next section.*

**** the case structure**
==
-- When there are several distinct possible values for a single variable and depending on that value, you proceed to execute a series of instructions, then we need to consider the case statement

**** structure**

```
case of {variable}
        {value}: {instructions}
        {value}: {instructions}
        {value}: {instructions}
        [default]: {instructions}
endcase
```

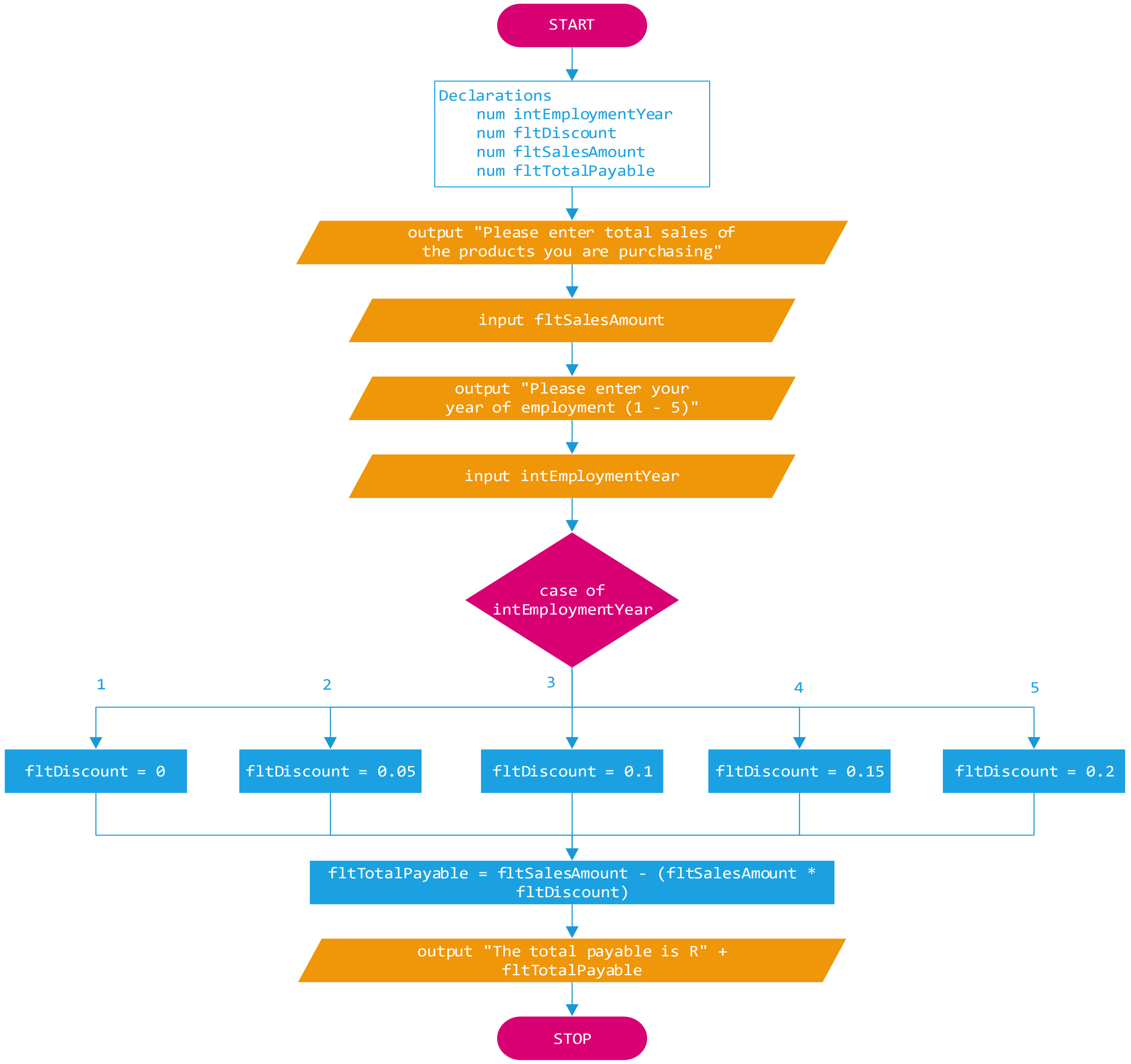
- The case statement allows for the variable to be evaluated and if it equates to a value from the list of distinct values, then the instructions listed are executed

- The [default] piece is optional, where if the value of the variable is not in the distinct list of values being dealt with, then the [default] section is executed. (Think of the default as being a kind of else in an if..then - if the variable is not equal to any of the values, then perform that else section)

-- Now, let us look at the problem and solution from before, since there are distinct year numbers provided, 1,2,3,4,5 - instead of using the nested if..then statements, we could now consider the case statement for this solution.

**** pseudocode**

```
0. Start
1. Declarations
        num intEmploymentYear
        num fltDiscount
        num fltSalesAmount
        num fltTotalPayable

2.      output "Please enter total sales of the products you are purchasing"
3.      input fltSalesAmount

4.      output "Please enter your year of employment (1 - 5)"
5.      input intEmploymentYear

6.      case of intEmploymentYear
                1: fltDiscount = 0
                2: fltDiscount = 0.05
                3: fltDiscount = 0.1
                4: fltDiscount = 0.15
                5: fltDiscount = 0.2
        endcase

11.     fltTotalPayable = fltSalesAmount - (fltSalesAmount * fltDiscount)

12.     output "The total payable is R" + fltTotalPayable

13. Stop
```

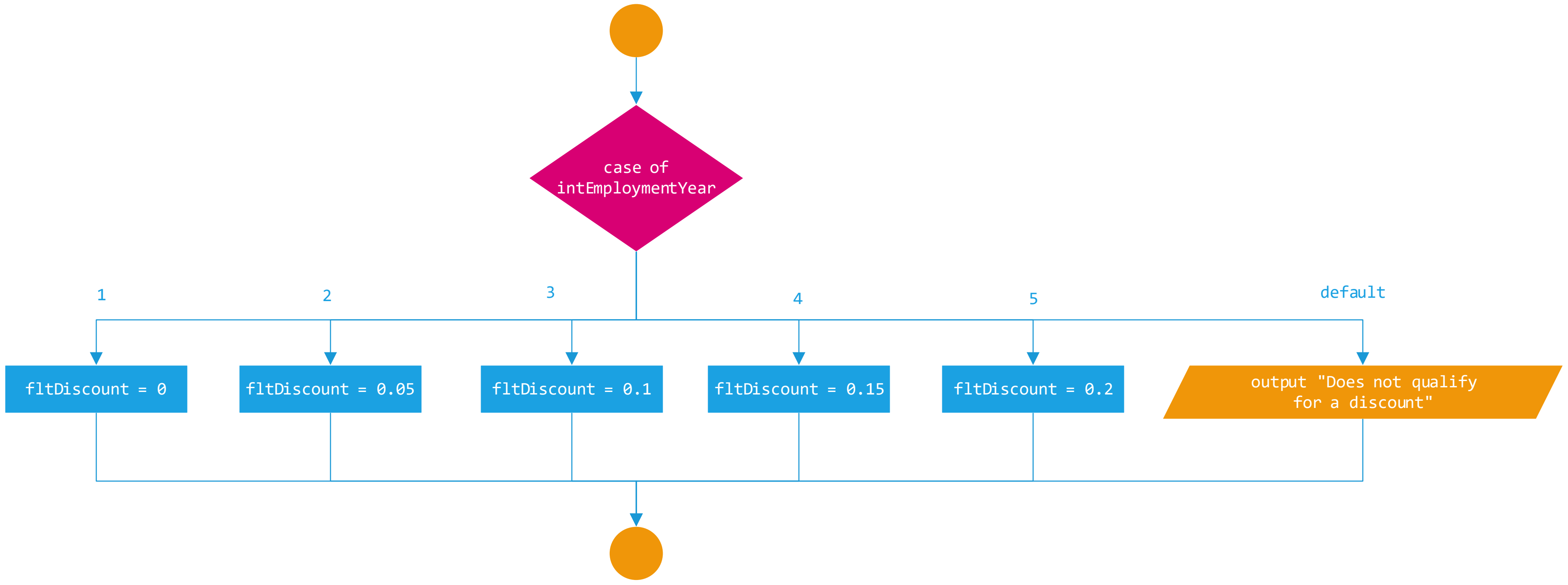

START
Declarations
num intEmploymentYear
num fltDiscount
num fltSalesAmount
num fltTotalPayable
output "Please enter total sales of the products you are purchasing"
input fltSalesAmount
output "Please enter your year of employment (1 - 5)"
input intEmploymentYear
case of intEmploymentYear
1
2
3
4
5
fltDiscount = 0
fltDiscount = 0.05
fltDiscount = 0.1
fltDiscount = 0.15
fltDiscount = 0.2
fltTotalPayable = fltSalesAmount - (fltSalesAmount * fltDiscount)
output "The total payable is R" + fltTotalPayable
STOP

- The above solution is offering a much simpler series of instructions as compared to the previous programs.

- As you can note, there is no default section, because it is optional and since there is no discount offered for an invalid year; we feel that is not necessary to deal with it.

- If we wanted to, we could introduce the default piece to the case statement, to print a kind of error message.

```
0. Start
1. Declarations
        num intEmploymentYear
        num fltDiscount
        num fltSalesAmount
        num fltTotalPayable

2.      output "Please enter total sales of the products you are purchasing"
3.      input fltSalesAmount

4.      output "Please enter your year of employment (1 - 5)"
5.      input intEmploymentYear

6.      case of intEmploymentYear
                1: fltDiscount = 0
                2: fltDiscount = 0.05
                3: fltDiscount = 0.1
                4: fltDiscount = 0.15
                5: fltDiscount = 0.2
                default: output "Does not qualify for a discount"
        endcase

11.     fltTotalPayable = fltSalesAmount - (fltSalesAmount * fltDiscount)

12.     output "The total payable is R" + fltTotalPayable

13. Stop
```



- However, the introduction of the default piece of code - maybe the output statement may not be the best choice to have made
- Considering that if the user enters anything other than 1, 2, 3, 4, or, 5 – then the discount is always going to be 0%

- Maybe the following is a better choice

```
6.      case of intEmploymentYear
                1: fltDiscount = 0
                2: fltDiscount = 0.05
                3: fltDiscount = 0.1
                4: fltDiscount = 0.15
                5: fltDiscount = 0.2
                default: fltDiscount = 0
        endcase
```

- if we did want to output that the number entered is incorrect (an error) - we could have chosen a little different route, and again, it is a possibility, but maybe not the best.

```
0. Start
1. Declarations
        num intEmploymentYear
        num fltDiscount
        num fltSalesAmount
        num fltTotalPayable

2.      output "Please enter total sales of the products you are purchasing"
3.      input fltSalesAmount

4.      output "Please enter your year of employment (1 - 5)"
5.      input intEmploymentYear

6.      if (intEmploymentYear < 1) OR (intEmploymentYear > 5) then
                output "The year of employment is invalid, and no discount is applied"
                fltTotalPayable = fltSalesAmount
        else
                case of intEmploymentYear
                        1: fltDiscount = 0
                        2: fltDiscount = 0.05
                        3: fltDiscount = 0.1
                        4: fltDiscount = 0.15
                        5: fltDiscount = 0.2
                endcase
                fltTotalPayable = fltSalesAmount - (fltSalesAmount * fltDiscount)
        endif

7.      output "The total payable is R" + fltTotalPayable

8. Stop
```

- The above solution now meets the needs of our thinking and deals with the information in a much nicer way than the original

----

**** Menus and the case structure**

==

-- One of the more interesting uses of the case statement is when we are required to present and deal with a menu of choices for a user to choose from.

- **Story**
- Allow the user an opportunity to choose from a menu of choices, to solve the following areas of 2 dimensional shapes:

Menu Choice	**Program**
1.	Area of a circle
2.	Area of a square
3.	Area of a rectangle
4.	Area of a triangle

- After the user chooses an option, you must offer the menu choices repeatedly until the user chooses to exit the program.
- Please offer an option #5 to end the program.
- Make use of modules to deal with the input, processing and output of the above-mentioned menu choices (making the main program more readable and a little efficient)

Please note that the development of the program will require the use of *a loop logic structure* - This solution will use the do..while loop.

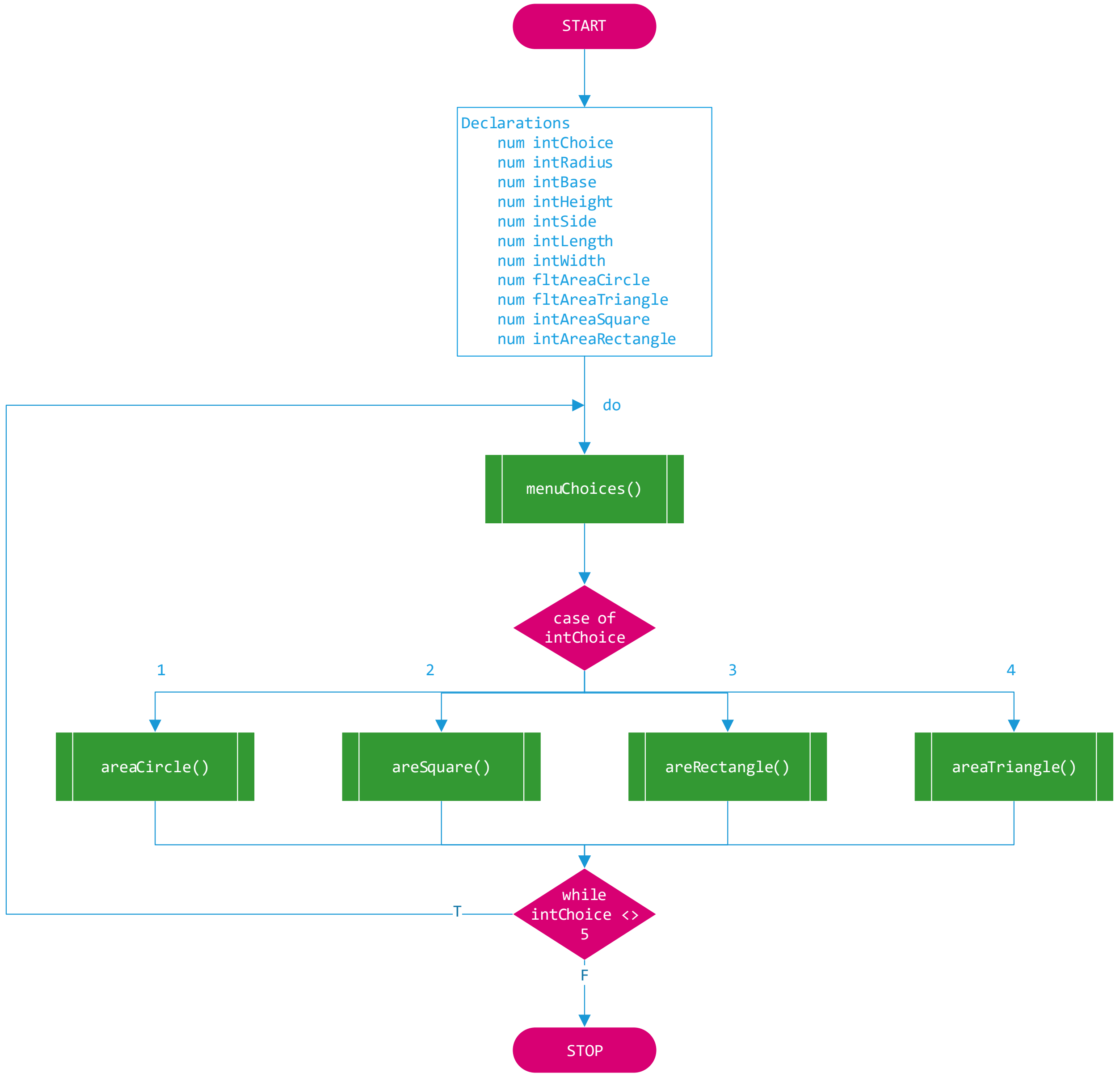
While we have begun the discussion of the next chapter and the notes will be shared soon - please try your best to understand why and how the loop logic code is being used in this solution.

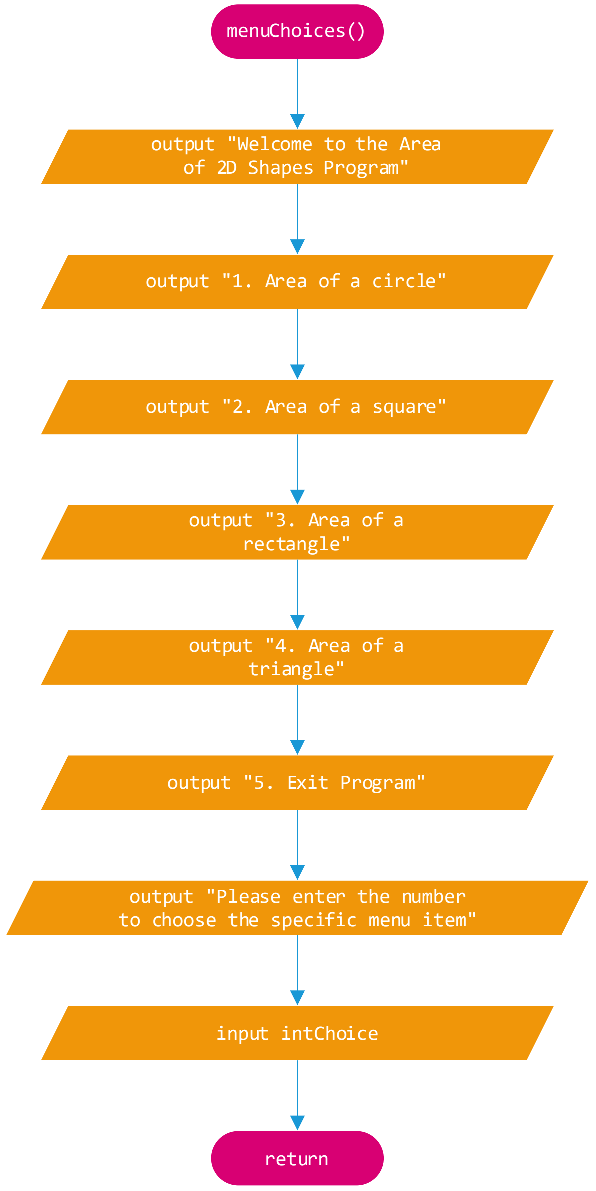
** pseudocode

```
        0. Start
        1. Declarations
                num intChoice
                num intRadius
                num intBase
                num intHeight
                num intSide
                num intLength
                num intWidth
                num fltAreaCircle
                num fltAreaTriangle
                num intAreaSquare
                num intAreaRectangle

        2.      do
                        menuChoices()
                        case of intChoice
                                1: areaCircle()
                                2: areaSquare()
                                3: areaRectangle()
                                4: areaTriangle()
                        endcase
                while (intChoice <> 5)

        11. Stop
================================================================================================
        0. menuChoices()
        1.      output "Welcome to the Area of 2D Shapes Program"
        2.      output "1. Area of a circle"
        3.      output "2. Area of a square"
        4.      output "3. Area of a rectangle"
        5.      output "4. Area of a triangle"
        6.      output "5. Exit Program"
        7.      output "Please enter the number to choose the specific menu item"
        8.      input intChoice
        9. return
====================================================================================================
        0. areaCircle()
        1.      output "Please enter the radius of the circle"
        2.      input intRadius

        3.      fltAreaCircle = 3.14 * (intRadius ^ 2)
        4.      output "The area of the circle is " + fltAreaCircle

        5. return
====================================================================================================
        0. areaSquare()
        1.      output "Please enter the length of one side of the square"
        2.      input intSide

        3.      intAreaSquare = intSide * intSide
        4.      output "The area of the square is " + intAreaSquare

        5. return
====================================================================================================
        0. areaRectangle()
        1.      output "Please enter the length of the rectangle"
        2.      input intLength

        3.      output "Please enter the width of the rectangle"
        4.      input intWidth

        5.      intAreaRectangle = intLength * intWidth
        6.      output "The area of the rectangle is " + intAreaRectangle

        7. return
```

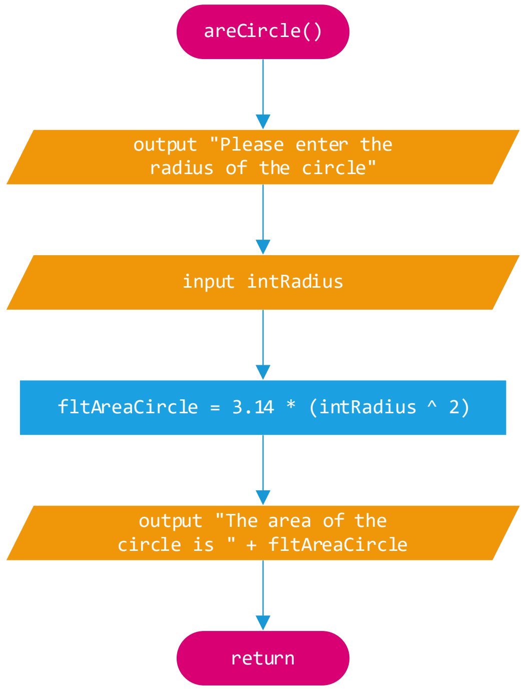
```
=====================================================================================================
        0. areaTriangle()
        1.      output "Please enter the length of the base of the triangle"
        2.      input intBase

        3.      output "Please enter the length of the height of the triangle"
        4.      input intHeight

        5.      fltAreaTriangle = 0.5 * intBase * intHeight
        6.      output "The area of the triangle is " + fltAreaTriangle

        7. return
=====================================================================================================
```

The flowchart should also provide a better understanding as well.

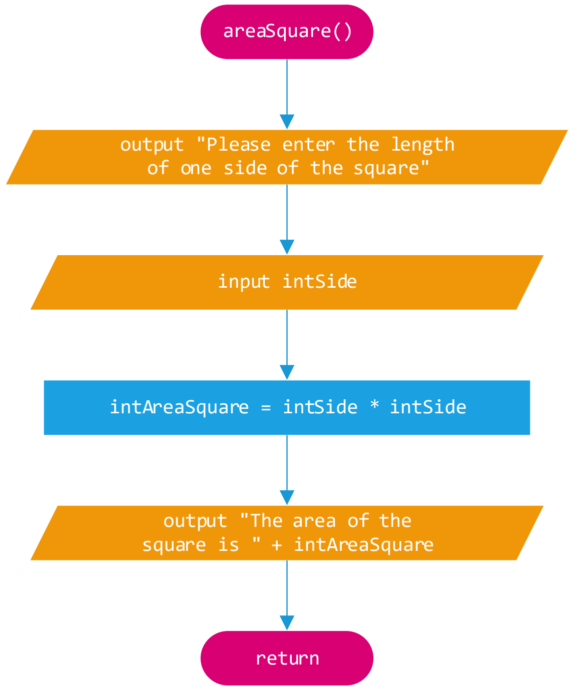

START
Declarations
num intChoice
num intRadius
num intBase
num intHeight
num intSide
num intLength
num intWidth
num fltAreaCircle
num fltAreaTriangle
num intAreaSquare
num intAreaRectangle
do
menuChoices()
case of intChoice
1
2
3
4
areaCircle()
areSquare()
areRectangle()
areaTriangle()
while intChoice <> 5
T
F
STOP



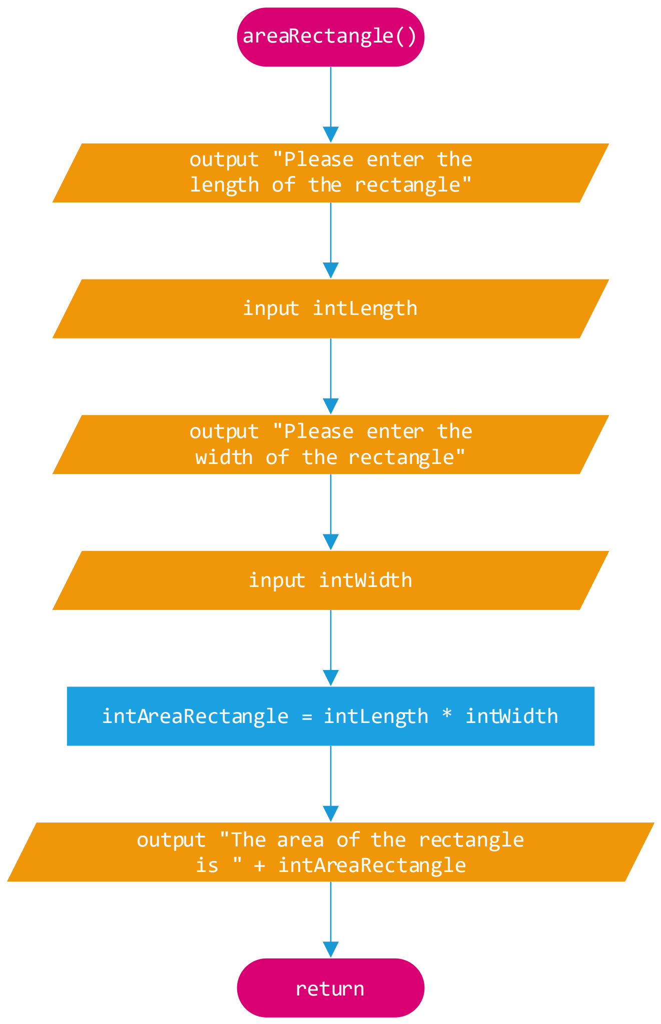
```
menuChoices()
output "Welcome to the Area of 2D Shapes Program"
output "1. Area of a circle"
output "2. Area of a square"
output "3. Area of a rectangle"
output "4. Area of a triangle"
output "5. Exit Program"
output "Please enter the number to choose the specific menu item"
input intChoice
return
```

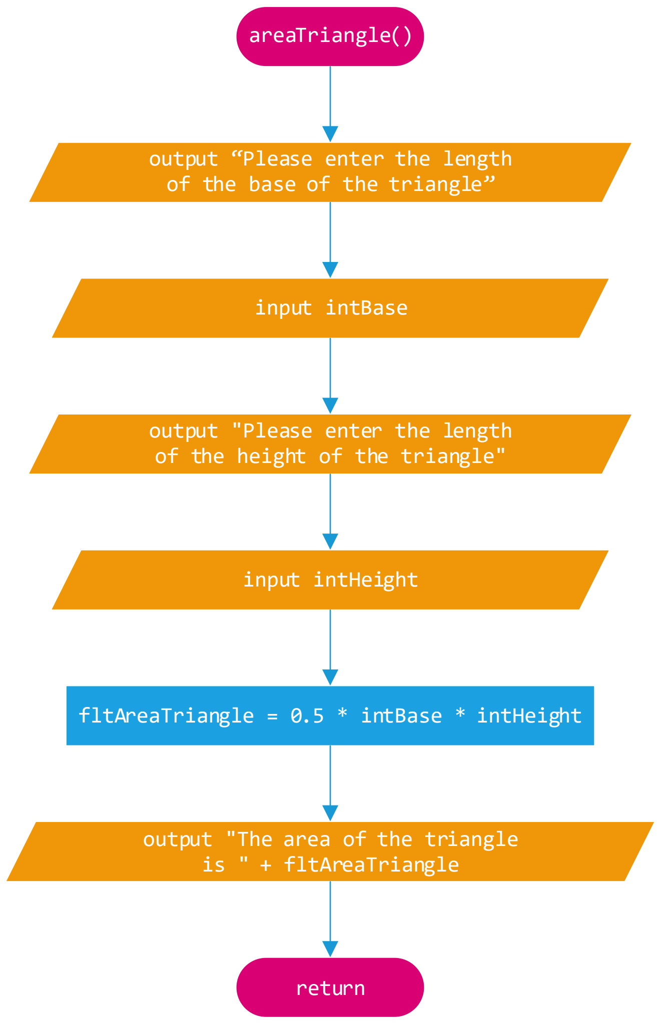


```
areCircle()
output "Please enter the radius of the circle"
input intRadius
fltAreaCircle = 3.14 * (intRadius ^ 2)
output "The area of the circle is " + fltAreaCircle
return
```



```
areaSquare()
output "Please enter the length of one side of the square"
input intSide
intAreaSquare = intSide * intSide
output "The area of the square is " + intAreaSquare
return
```


areaRectangle()
output "Please enter the length of the rectangle"
input intLength
output "Please enter the width of the rectangle"
input intWidth
intAreaRectangle = intLength * intWidth
output "The area of the rectangle is " + intAreaRectangle
return


areaTriangle()
output “Please enter the length of the base of the triangle”
input intBase
output "Please enter the length of the height of the triangle"
input intHeight
fltAreaTriangle = 0.5 * intBase * intHeight
output "The area of the triangle is " + fltAreaTriangle
return

- Please do remember that the use of the case statement is many. So, it is important to remember that when there is a singular value for many choices, then it is likely that we would choose to use a case statement instead of multiple nested if..then statements.