National 5

Computing Science

Software Design and Development

Practical Workbook

Contents

Level 1: Variables (Input and Output)

Learning Intentions

Outcome 1
We are learning to analyse and design simple computing solutions using a range of design techniques.

Outcome 2
We are learning how write programs using variables and sequence and to debug them when they don't work.

Outcome 3
We are learning how to test and evaluate our programs to prove their success.

Success Criteria

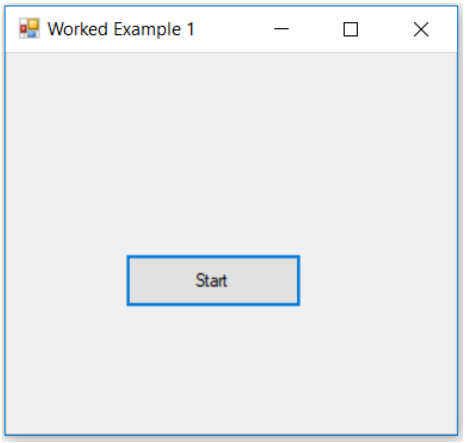
Outcome 1	I can analyse a problem and identify inputs, processes and outputs.
	I can analyse a problem and identify required variables and data types
	I can design a solution to simple problems using flow charts and/or structure diagrams
	I can design a solution to simple problems using pseudocode
Outcome 2	I can write code correctly to declare variables, with the correct data types, and initialise them
	I can use the correct sequence of commands to collect inputs, perform calculations and display correct outputs.
	I can correctly use variables within my code.
	I can debug code on my own by correcting syntax, execution and logic errors.
Outcome 3	I can carry out testing of my program to prove it works
	I can create a test plan and record results of testing accurately
	I can evaluate the success of my program in terms of fitness for purpose and readability

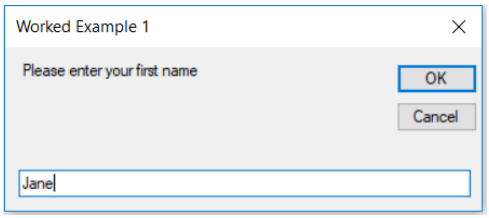
Worked Example 1: Greetings — Level 1

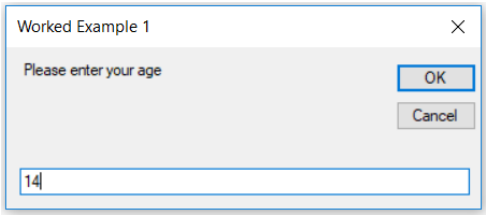
Problem Specification

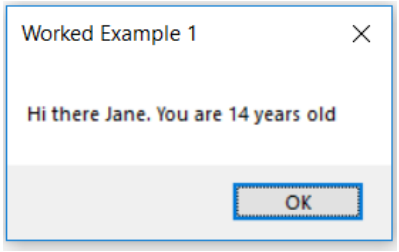
A program is required that will ask the user to enter their first name and age.

The program should then display a message that reads something like, "Hi there Jane. You are 14 years old".









Analysis

Inputs	Process	Outputs
• First Name • Age	Concatenate first name and age with message	• First Name • Age

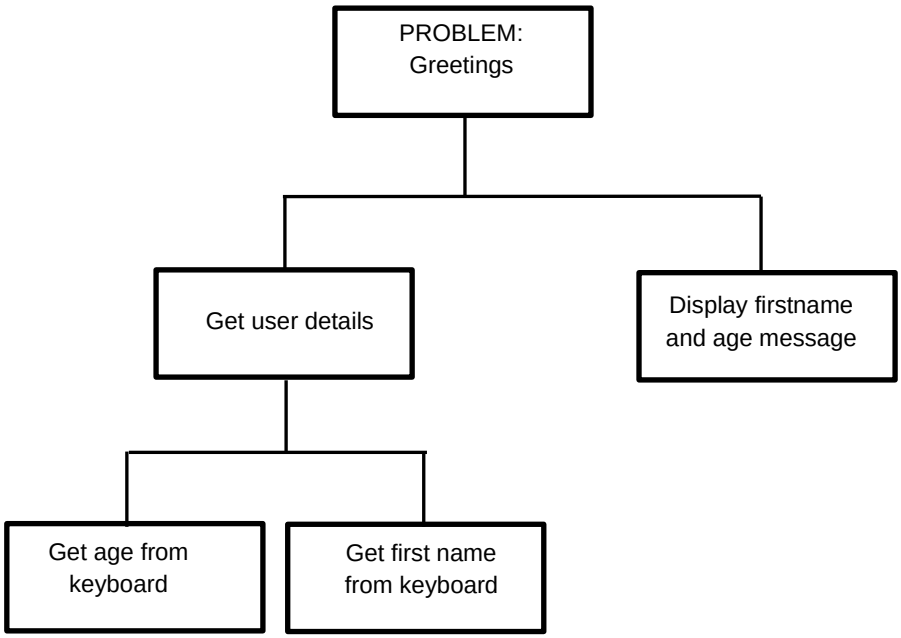
Data Items	Data Types
First Name	String
Age	Integer

A name uses LETTERS

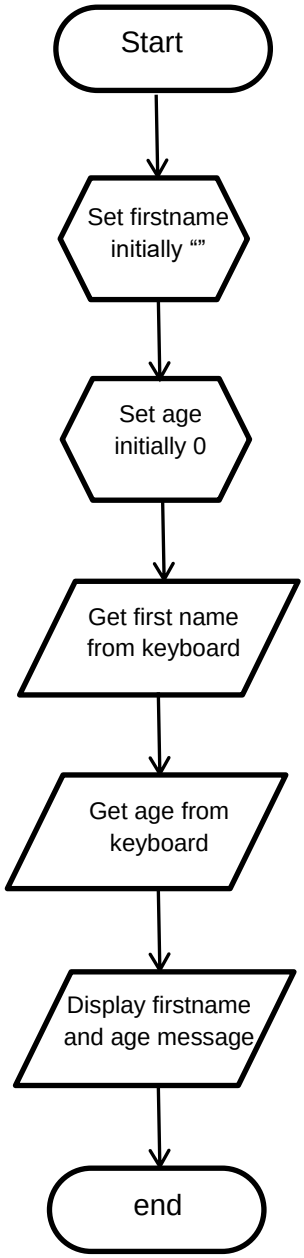
An age is a WHOLE NUMBER

Design

Structure Diagram



Flow Chart



Pseudocode

Algorithm

1. Initialise variable
2. Ask user to enter their details
3. Display message stating user details

Refinements

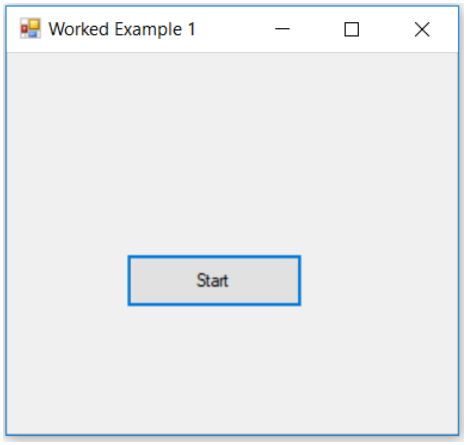
2.1 Ask user to enter first name

2.2. Ask user to enter age

3.1 Display "Hi there ", firstname, ". You are ", age, " years old."

Implementation

Create a new Visual Basic project called "*Worked Example 1*"



Add a button with the following properties set:

(Name)	**btnStart**
Text	**Start**

Double click the ***button*** and add the code below:

```
Public Class Form1

Private Sub btnStart_Click(sender As Object, e As EventArgs) Handles btnStart.Click

        Dim firstName As String
        Dim age As Integer

        firstName = ""
        age = 0

        firstName = InputBox("Please enter your first name")
        age = InputBox("Please enter your age")

        MsgBox("Hi there " & firstName & ". You are " & age & " years old")

End Sub

End Class
```

Declare Variables

Initialise Variables

Get Inputs

Display Output

Testing

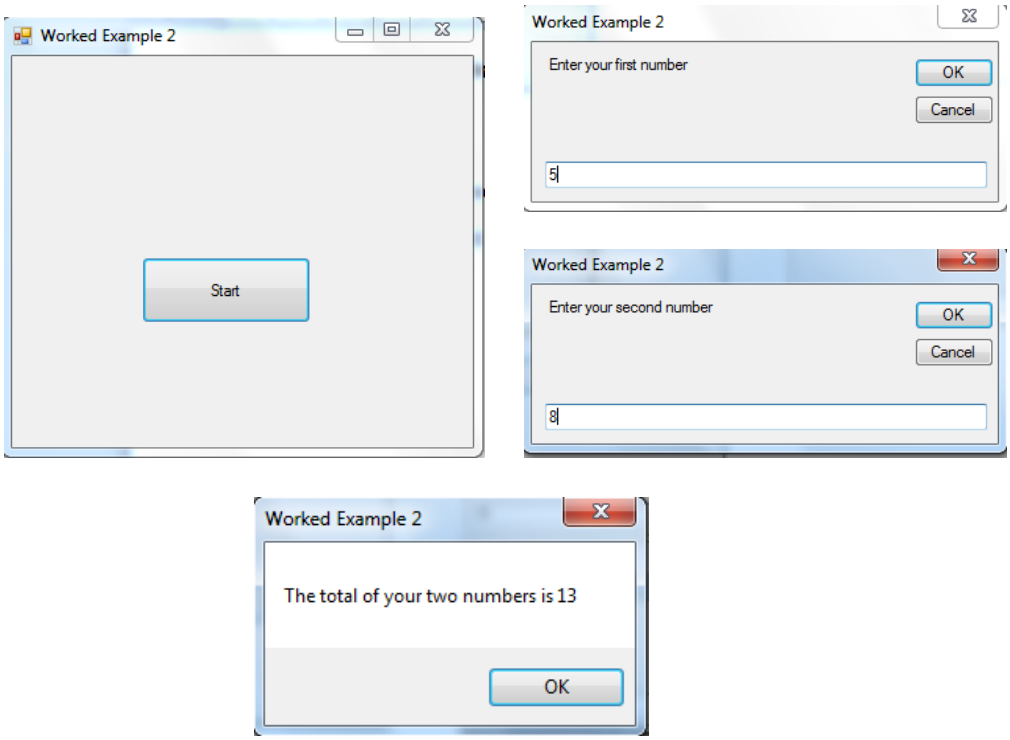
Run the program

- Make sure the firstname and age can be entered.
- Make sure the output is displayed correctly.

Problem Specification

A program is required that will ask the user to enter two whole numbers.

The program should add the numbers together and display the total using a suitable message.



Analysis

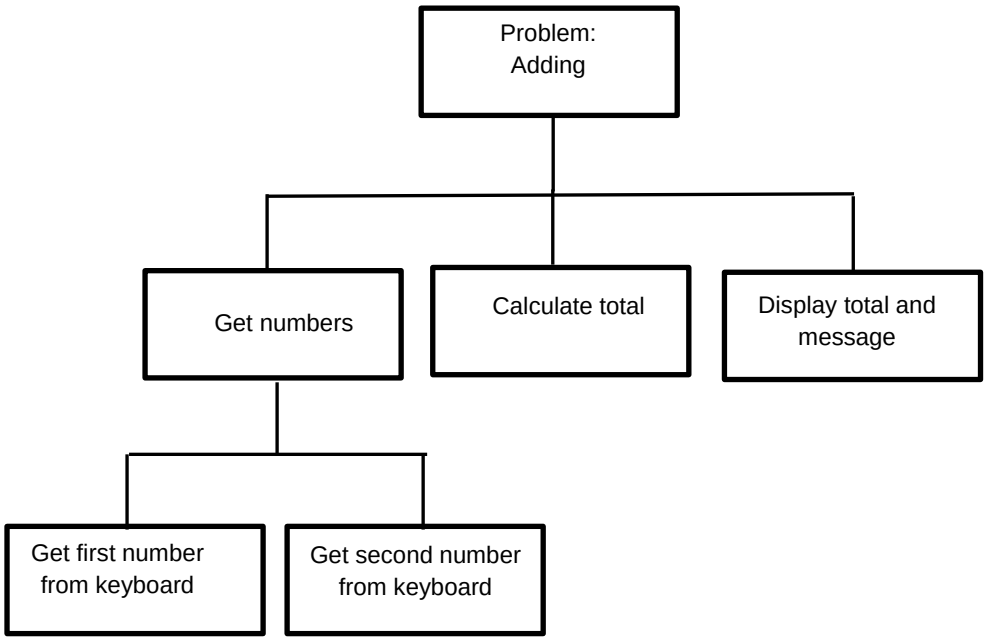
Inputs	Process	Outputs
• First Number • Second Number	Calculate the total by adding the two numbers together	• Total

Data Items	Data Types
First Number	Integer
Second Number	Integer
Total	Integer

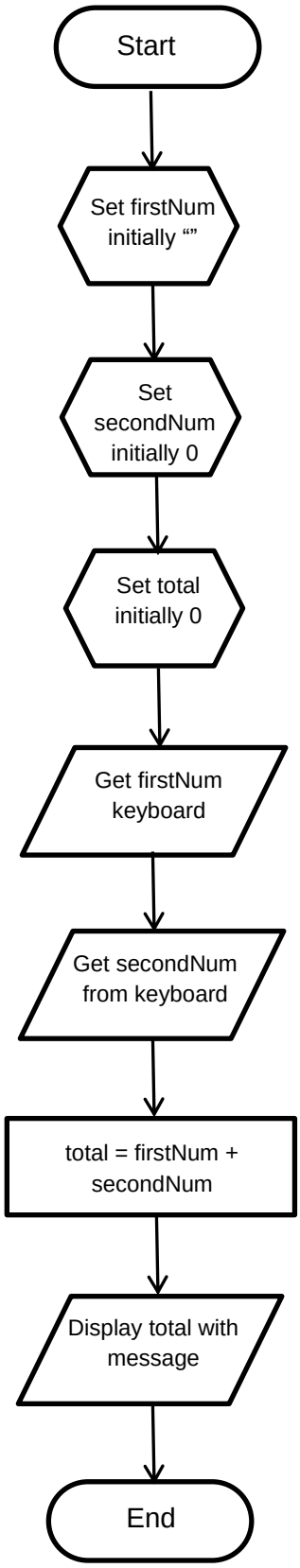
All of the numbers to be entered are WHOLE NUMBERS

Design

Structure Diagram



Flow Chart



Pseudocode

Algorithm

1. Initialise variable
2. Ask user to enter their number
3. Calculate total
4. Display message stating total

Refinements

2.1 Ask user to enter first number

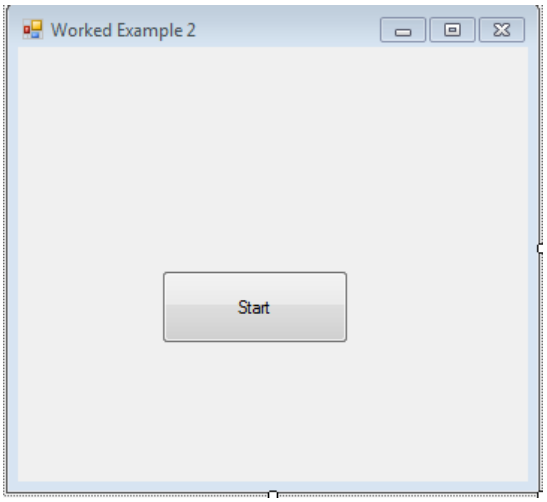
2.2. Ask user to enter second number

3.1 Total is calculated as first number + second number

4.1 Display "The total of your two numbers is ", total

Implementation

Create a new Visual Basic project called "*Worked Example 2*"



Add a button with the following properties set:

(Name)	**btnStart**
Text	**Start**

Double click the ***button*** and add the code below:

```
Public Class Form1


Private Sub btnStart_Click(sender As Object, e As EventArgs) Handles btnStart.Click

        Dim firstNum As Integer
        Dim secondNum As Integer
        Dim total As Integer

        firstNum = 0
        secondNum = 0
        total = 0

        firstNum = InputBox("Enter your first number")
        secondNum = InputBox("Enter your second number")

        total = firstNum + secondNum

        MsgBox("The total of your two numbers is " & total)
End Sub

End Class
```

Declare Variables

Initialise Variables

Get Inputs

Expression assigning value to variable

Display Output

Testing

Run the program

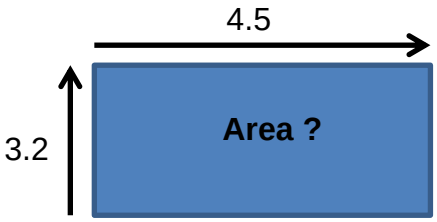
- Make sure the first number and second number can be entered.
- Make sure the total is displayed correctly
- Run the program a few times and check the total is correct each time.

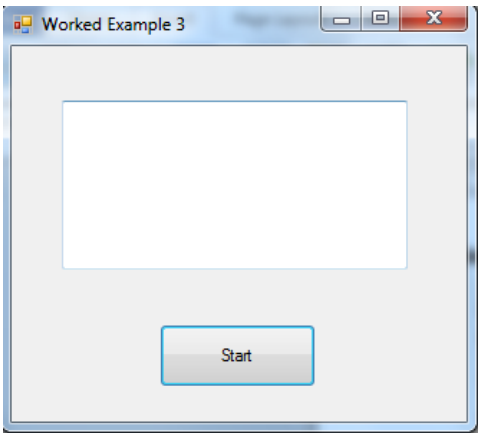
Worked Example 3: Calculations Level 1

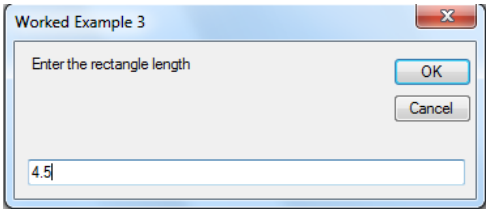
Problem Specification

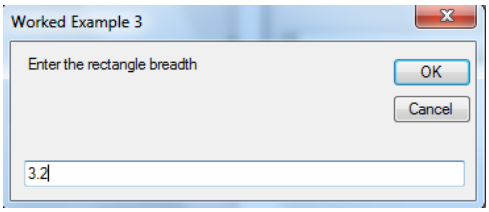
A program is required that will ask the user to enter the length and breadth of a rectangle.

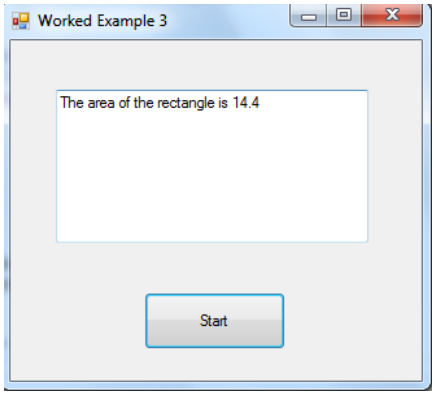
The program should use this information to calculate the area of the rectangle and display it in a suitable message.











Analysis

Inputs	Process	Outputs
• Length • Breadth	Calculate the area by multiplying the two numbers together	• Area

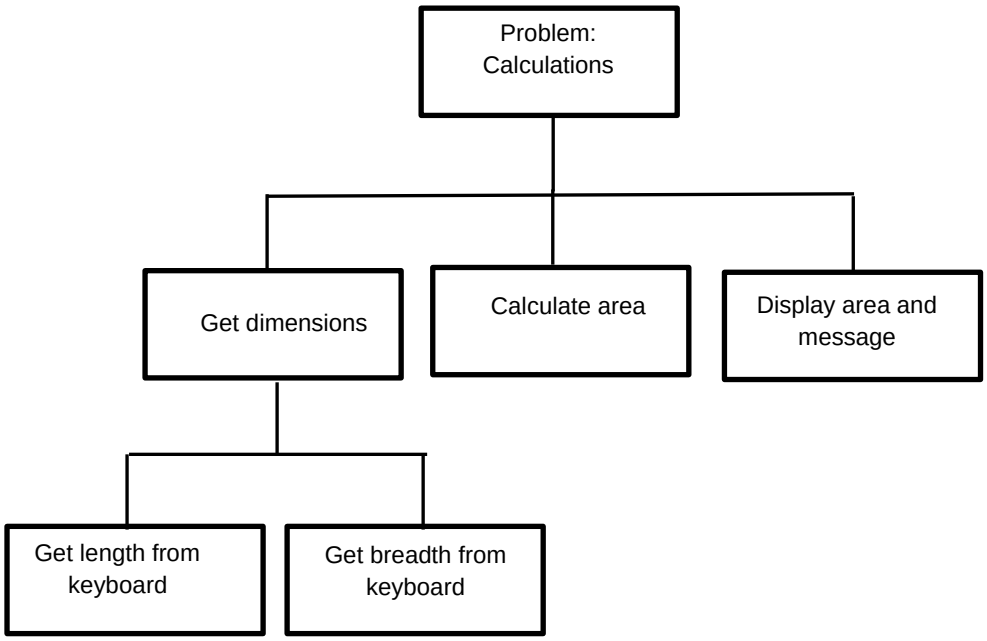
Variables	Data Types
Length	Single
Breadth	Single
Area	Single

All of the numbers to be entered have decimal points.

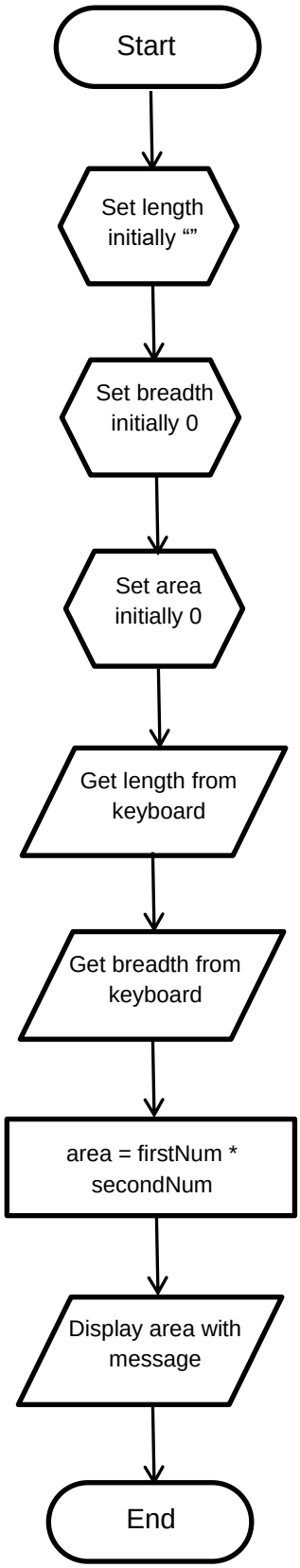
They are REAL NUMBERS

Design

Structure Diagram



Flow Chart



Pseudocode

Algorithm

1. Initialise variable
2. Ask user to enter rectangle dimensions
3. Calculate area
4. Display message stating area

Refinements

2.1 Ask user to enter length

2.2. Ask user to enter breadth

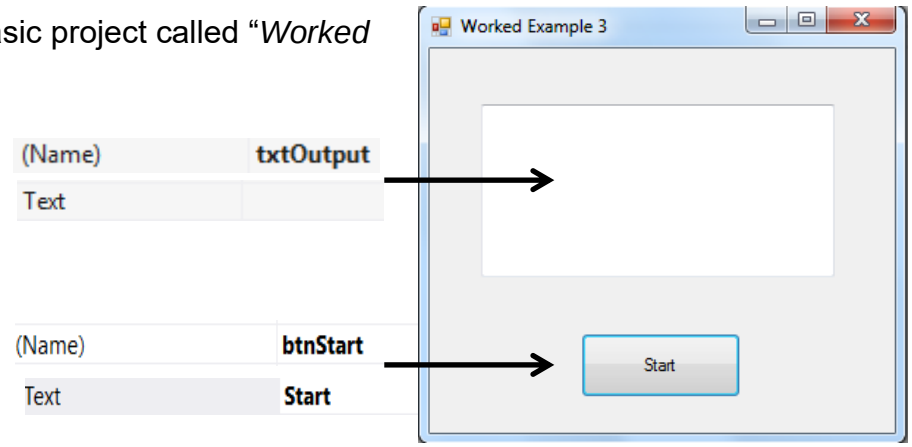
3.1 Area is calculated as length * breadth

4.1 Display "The area of the rectangle is ", area

Implementation

Create a new Visual Basic project called "*Worked Example 3*"

Add a button and a textbox as shown:



Double click the ***button*** and add the code below:

```
Public Class Form1


Private Sub btnStart_Click(sender As Object, e As EventArgs) Handles btnStart.Click

        Dim length As Single
        Dim breadth As Single
        Dim area As Single

        length = 0.0
        breadth = 0.0
        area = 0.0

        length = InputBox("Enter the rectangle length")
        breadth = InputBox("Enter the rectangle breadth")

        area = length * breadth

        txtOutput.AppendText("The area of the rectangle is " & area)
End Sub

End Class
```

Declare Variables

Initialise Variables

Get Inputs

Expression assigning value to variable

Display Output

Testing

Run the program

- Make sure the length and breadth can be entered.
- Make sure the area is displayed correctly
- Run the program a few times and check the area is correct each time.

Mobile Phone

Program Specification

A program is required to conduct a survey for a mobile phone company. The program should ask a user to enter their first name, the type of phone they have and what their favourite app is.

A message confirming their details should be displayed that reads something like, “Your name is _____. The type of phone you have is a _____ and your favourite app is _____”.

Social Network

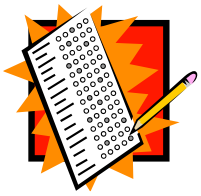
Program Specification

Create a program to let other pupils know about your favourite social networks. The user will need to enter their name and what their favourite social network site is. The program should also ask how many friends of followers they have.

The program should display the users name and their favourite social networking together with the number of friends of followers they have.

Test Scores

Program Specification



A program is required that will calculate the total score a pupil achieved from three test scores. The program should ask the user for their test scores in English, Maths and History. The total score should be calculated by the program and displayed on the screen with a suitable message.

MP3 Calculation

Program Specification

A program is required to calculate the total cost of mp3s being purchased from a company called Mytunes. Each song costs £1.99 and they the program should ask a user to enter the number of mp3 songs would like to purchase.

The program should calculate the total cost of all songs and display it on screen using a suitable message.

Theme Park

Program Specification

The Dalton Towers theme park require a program to calculate the cost of tickets for people visiting the park. The cost of a day wrist band for adults is £20 and for children it costs £15.

The program should ask the user how many adult tickets and how many child tickets they would like. The total cost should then be calculated and this information displayed with a suitable message.

Travel Time

Program Specification

A bus company require a program to help them to create their new route timetables. The program should ask the user the distance between bus stops and the speed limit the bus can travel at.

Using this information, the program should calculate the time it will take the bus to travel the distance given and display it on screen in a suitable message. (time = distance ÷ speed)

Level 2: Selection Statements (IF)

Learning Intentions

Outcome 1
We are learning how to analyse and design programs that can make decisions by using a range of design techniques.

Outcome 2
We are learning how write and debug programs that use selection (IF) statements to allow the program to make decisions based on inputs

Outcome 3
We are learning how to test our programs using normal and extreme test data.

Success Criteria

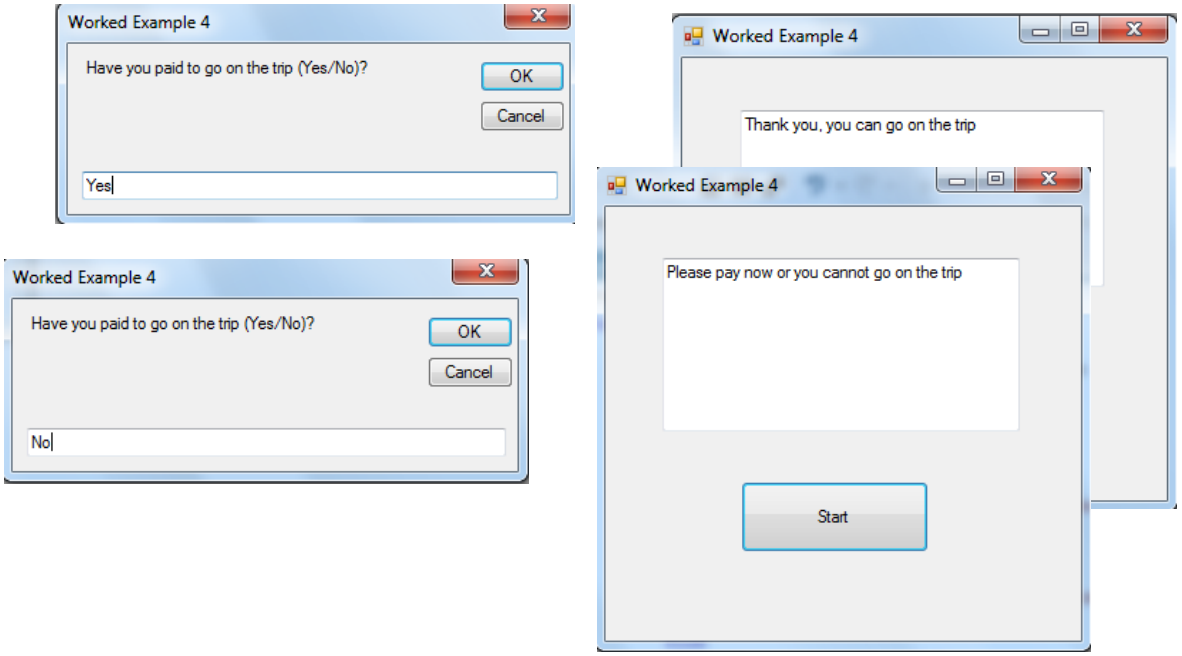
Outcome 1	I can analyse a problem and identify inputs, processes and outputs.
	I can analyse a problem and identify required variables and data types
	I can design solutions to problems that require selection (IF) statements using flow charts and/or structure diagrams
	I can design a solution to problems that require selection (IF) statements using pseudocode
Outcome 2	With some help, I can write programs involving IF statements.
	I can make the correct use of IF statements and sequence to create working programs that make decisions.
	I can correctly use variables within my IF statements.
	I can debug code on my own by correcting syntax, execution and logic errors.
Outcome 3	I can carry out testing of my program to prove it works
	I can create a test plan that involves normal and extreme test data and record results of testing accurately
	I can evaluate the success of my program in terms of fitness for purpose and readability

Worked Example 4: School Trip Level 2

Problem Specification

Pupils in a school are going on a trip to the zoo. Each pupil must pay the cost of the trip and a program is required to record details of whether pupils have paid or not.

The program should ask the user if they have paid. If they say yes then a message should be displayed saying, “Thank you, you can go on the trip”. If they say no then the message should read, “Please pay now or you cannot go on the trip”.



Analysis

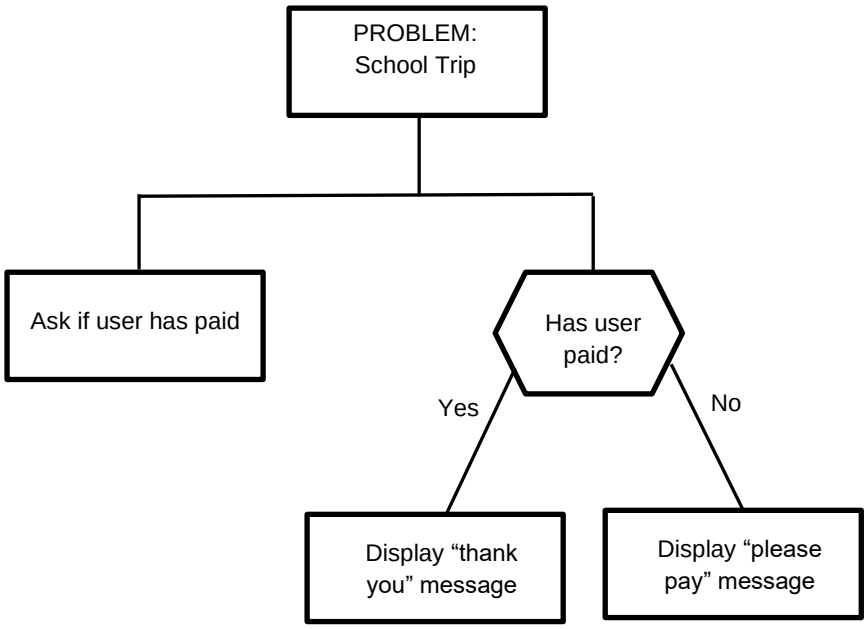
Inputs	Process	Outputs
• Has user paid	Decide if user can go to the zoo or not	Different message depending on whether paid is yes or no.

Data Items	Data Types
Paid	String

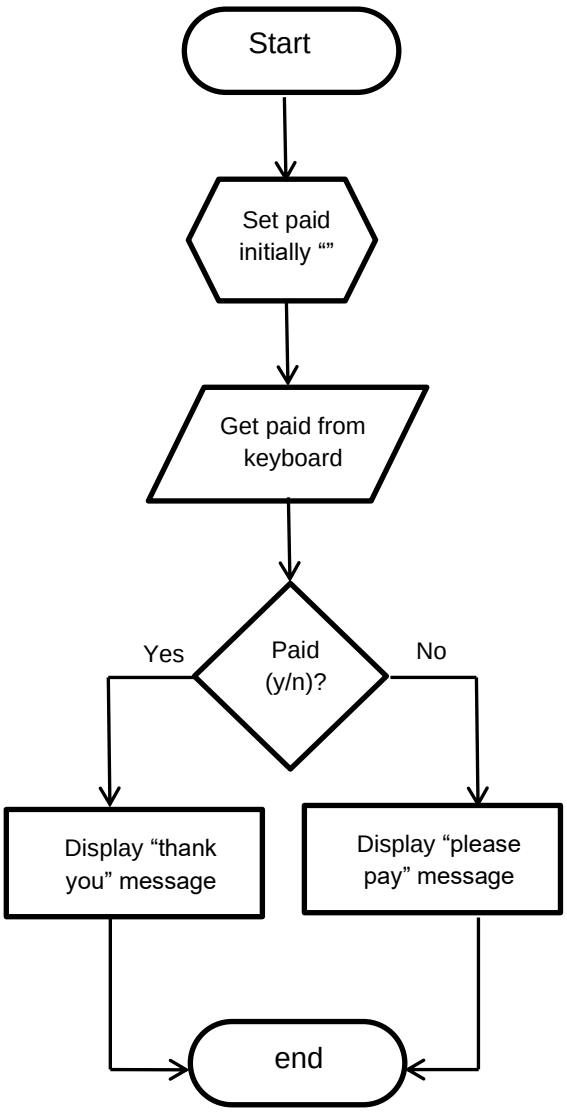
Paid will contain either the word Yes or No

Design

Structure Diagram



Flow Chart



Pseudocode

Algorithm

1. Initialise variable

2. Ask user if they have paid and display correct message

Refinements

2.1 Ask user to enter Yes or No and store in paid

2.2 IF paid = “Yes” Then

2.3 display "Thank you, you can go on the trip."

2.4 Else

2.5 display "Please pay no or you cannot go on the trip."

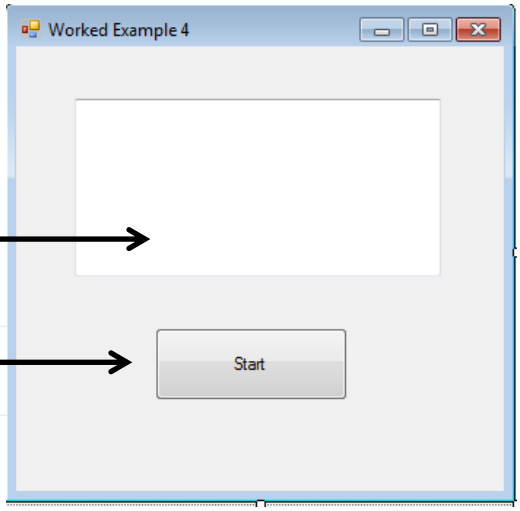
2.6 End if

Implementation

Create a new Visual Basic project called "*Worked Example 4*"

Add a button and a textbox as shown:

(Name)	txtOutput
Text	
(Name)	btnStart
Text	Start



Double click the ***button*** and add the code below:

```
Public Class Form1

Private Sub btnStart_Click(sender As Object, e As EventArgs) Handles btnStart.Click

        Dim paid As String

        paid = ""

        paid = InputBox("Have you paid to go on the trip (Yes/No)")

        If paid="Yes" Then
              txtOutput.AppendText("Thank you, you can go on the trip")
        Else
              txtOutput.AppendText("Please pay now or you cannot go on the trip")
        End If

End Sub

End Class
```

Declare Variable

Initialise Variable

Get Input

Selection

Testing

- Make sure the user can enter yes or no to say whether they have paid
- Make sure the output is displayed correctly for a yes.
- Make sure the output is displayed correctly for a no

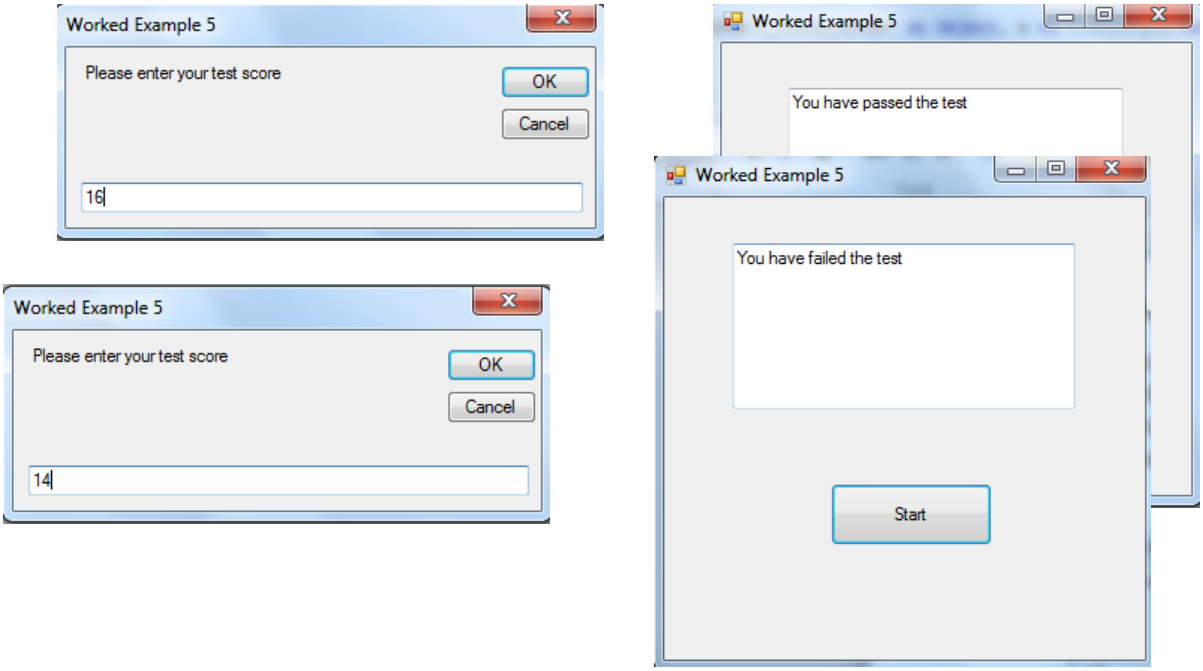
Run the program and complete the test table

Test	Type	Input	Expected Output	Actual Result	Result (pass/fail)
1	Normal	Paid= “Yes”	Thank you message	Result =	
2	Normal	Paid = “No”	Please Pay message	Result =	

Worked Example 5: Pass Mark — Level 2

Problem Specification

A program is required to ask pupils how many marks they got in their test out of 30. If a pupil got 15 marks or more, a message should indicate that the pupil has passed the test. Otherwise, the message should indicate that the pupil has failed.



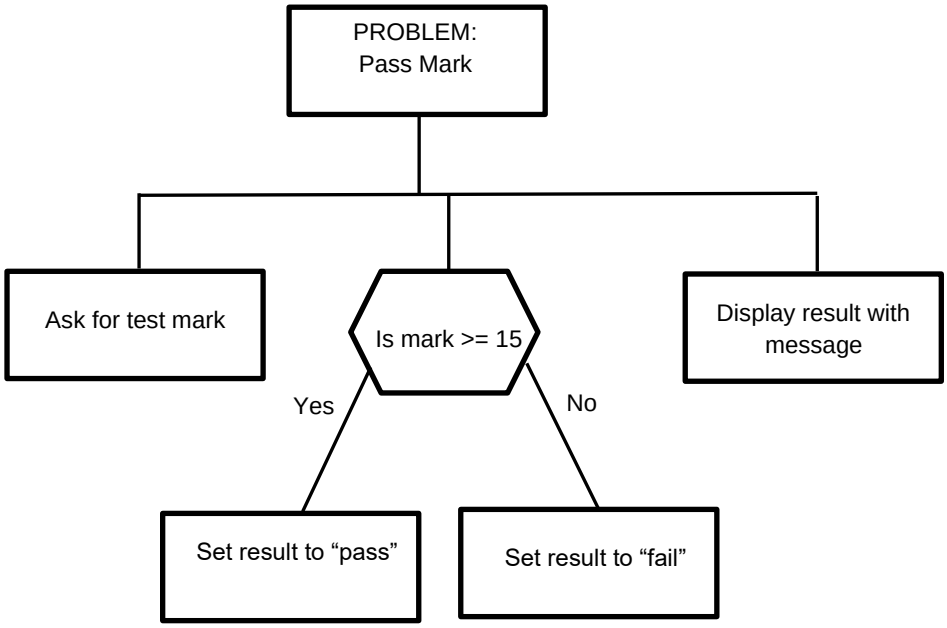
Analysis

Inputs	Process	Outputs
• Test Mark	Decide if pupil has passed or not	Test Result

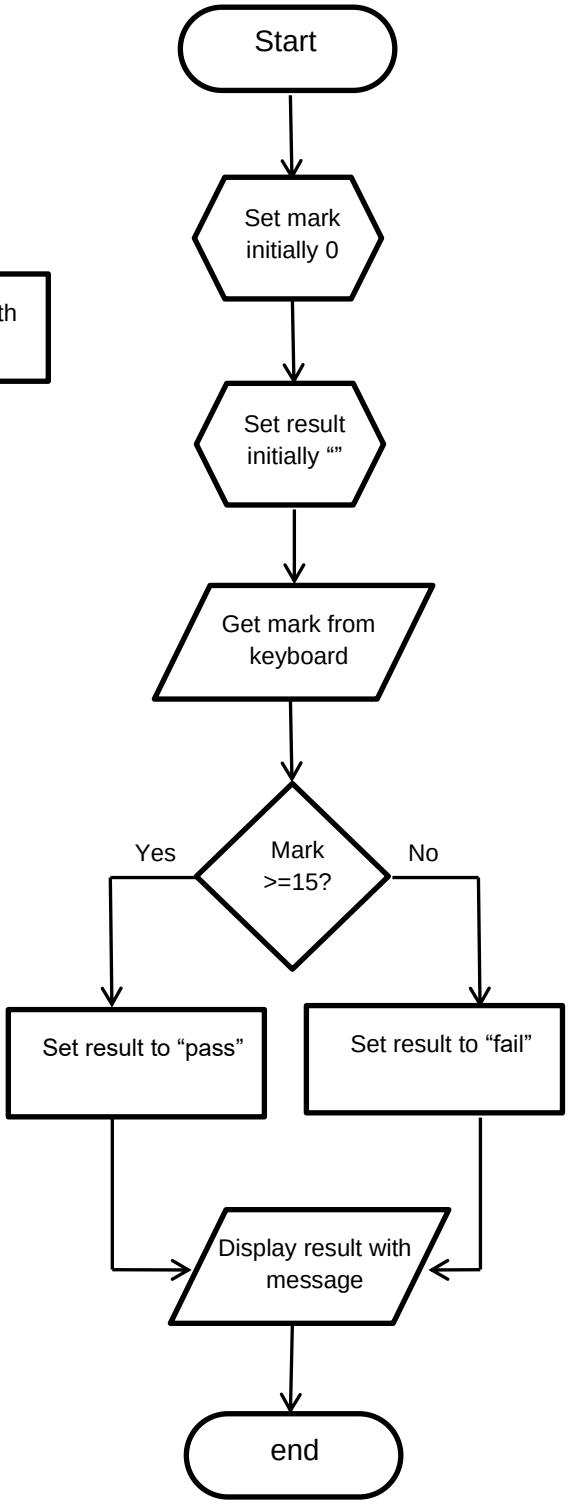
Data Items	Data Types	
Mark	Integer	Mark will contain a WHOLE NUMBER
Result	String	Result will contain Pass or Fail

Design

Structure Diagram



Flow Chart



Pseudocode

Algorithm

1. Initialise variables
2. Ask user for test mark
3. Decide result
4. Display result

Refinements

2.1 Ask user to enter Yes or No and store in paid

3.1 If mark >= 15 Then

3.2 set result to “passed”

3.3 Else

3.4 set result to “failed”

3.5 End If

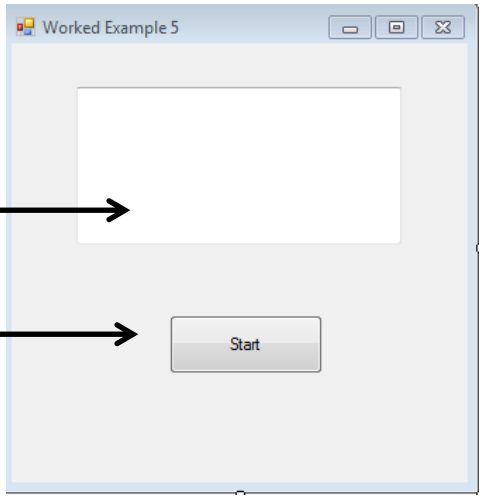
4.1 display “You have ”, result, “ the test.”

Implementation

Create a new Visual Basic project called "*Worked Example 5*"

Add a button and a textbox as shown:

(Name)	txtOutput
Text	
(Name)	btnStart
Text	Start



Double click the ***button*** and add the code below:

```
Public Class Form1

Private Sub btnStart_Click(sender As Object, e As EventArgs) Handles btnStart.Click

    Dim mark As Integer
    Dim result As String

    mark = 0
    result = ""

    mark = InputBox("Please enter your test score")

    If mark >= 15 Then
        result = "passed"
    Else
        result = "failed"
    End If

    txtOutput.AppendText("You have " & result & " the test")
End Sub

End Class
```

Declare Variable

Initialise Variable

Get Input

Selection

Display Output

Testing

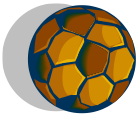
- Make sure the user can enter a mark
- Make sure the correct output is displayed for a mark over 15.
- Make sure the output is displayed correctly for a mark under 15.
- Make sure the output is displayed correctly for a mark of exactly 15.

Run the program and complete the test table

Test	Type	Input	Expected Output	Actual Result	Result (pass/fail)
1	Normal	Mark = 8	Result = “Fail”	Result =	
2	Normal	Mark = 25	Result = “Pass”	Result =	
3	Extreme	Age = 15	Result = “Pass”	Result =	
4	Extreme	Age = 14	Group = “Fail”	Result =	

Worked Example 6: Football Training Level 2

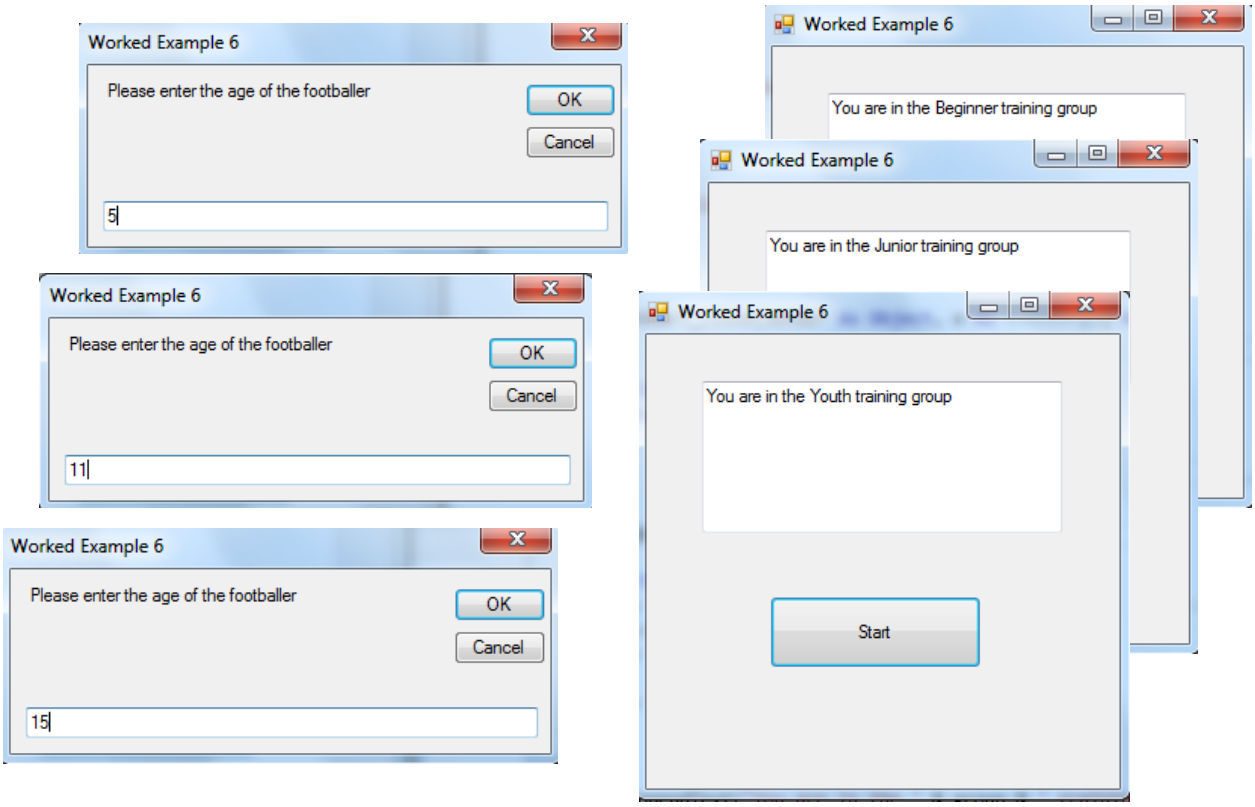
Problem Specification



A program is required to allocated boys and girls to the correct football training group depending on their age. There are three training groups which footballers can be in:

Under 8 years: Beginners
8 – 12 years: Juniors
13 – 17 years: Youths

The program should ask for the footballer's age and display a message telling them which training group to join.



Analysis

Inputs	Process	Outputs
• Age	Decide on the correct training group	• Group

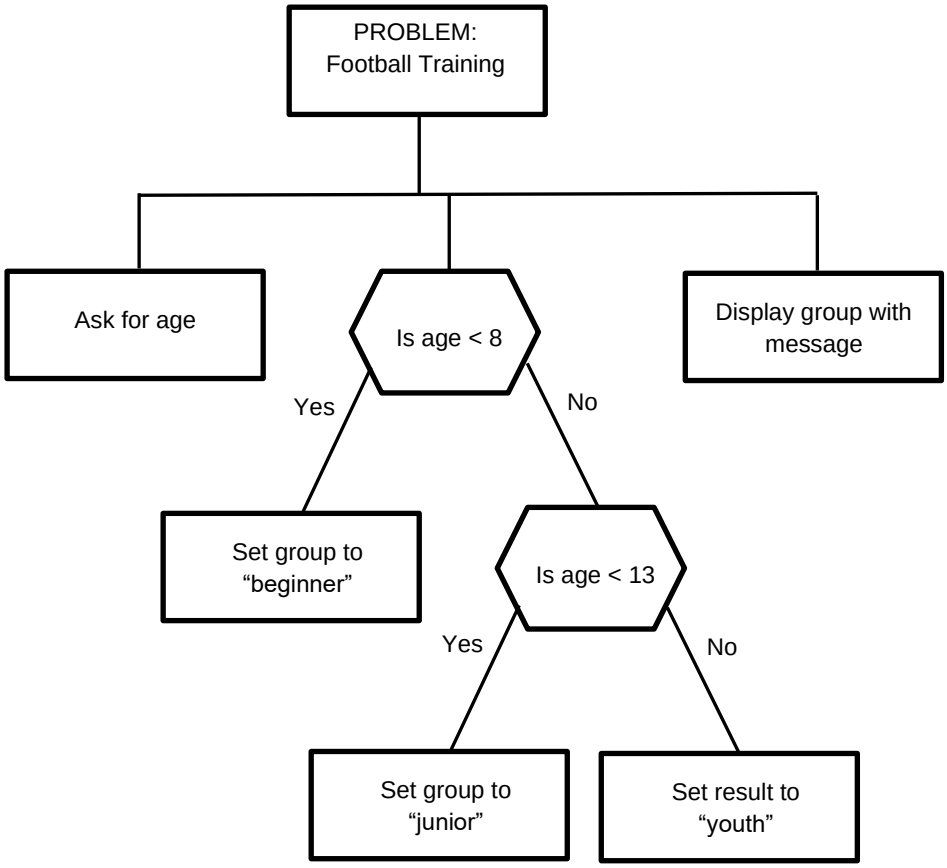
Data Items	Data Types
Age	Integer
Group	String

Age will contain a WHOLE NUMBER

Group will contain text

Design

Structure Diagram



Pseudocode

Algorithm

1. Initialise variables
2. Ask user for age
3. Decide group
4. Display group

Refinements

2.1 Ask user to enter their age

3.1 If age < 8 Then

3.2 set group to “beginner”

3.3 Else If age < 13 Then

3.4 set group to “junior”

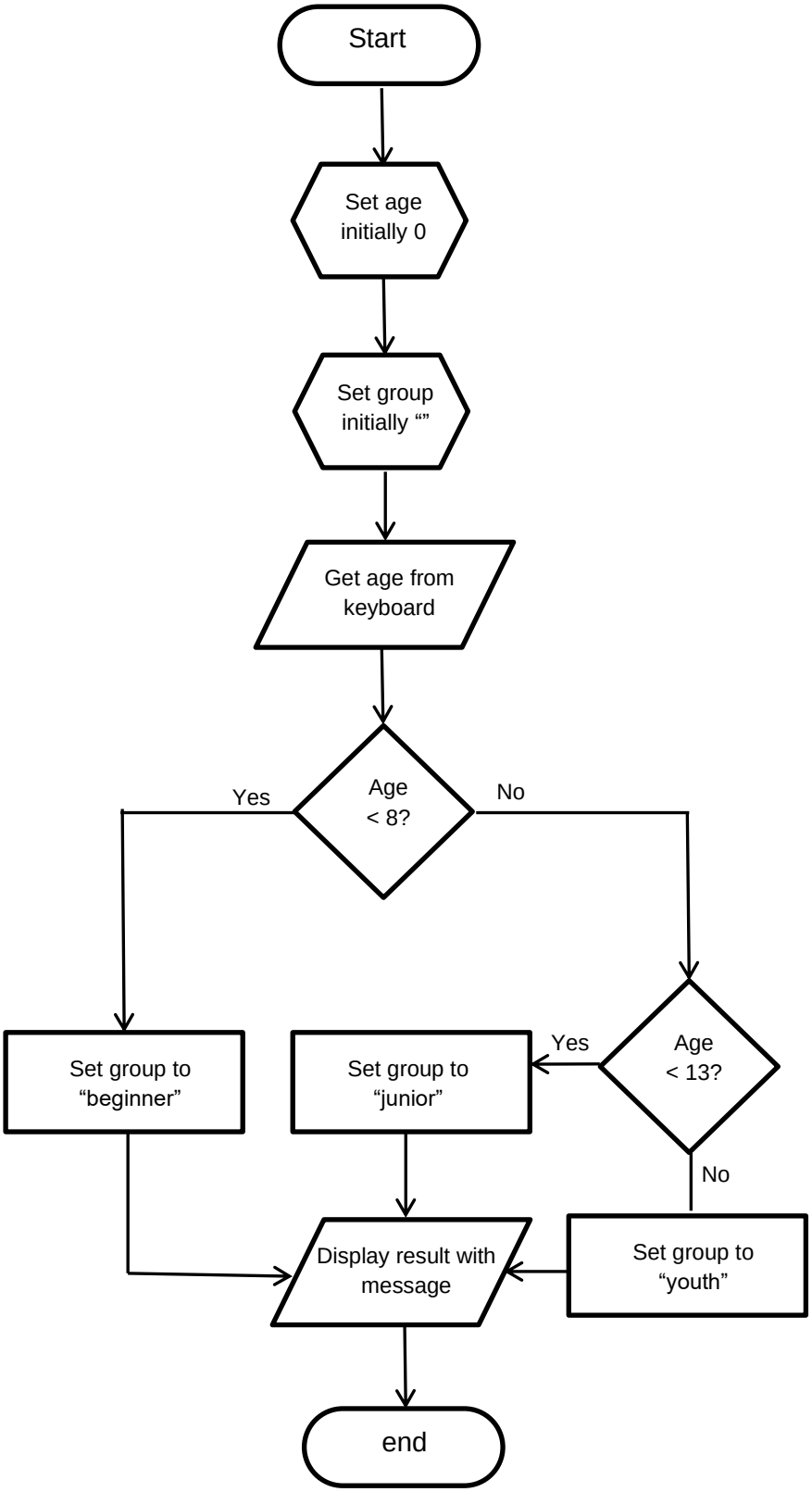
3.5 Else

3.6 set group to “youth”

3.7 End If

4.1 display “You are in the ”, group, “ group.”

Flow Chart


Start
Set age initially 0
Set group initially ""
Get age from keyboard
Age < 8?
Yes
No
Set group to "beginner"
Set group to "junior"
Age < 13?
Yes
No
Set group to "youth"
Display result with message
end

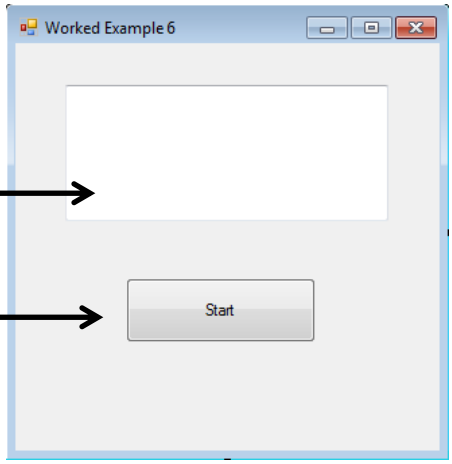
Implementation

Create a new Visual Basic project called "*Worked Example 6*"

Add a button and a textbox as shown:

(Name)	txtOutput
Text	

(Name)	btnStart
Text	Start



Double click the ***button*** and add the code below:

```
Public Class Form1

Private Sub btnStart_Click(sender As Object, e As EventArgs) Handles btnStart.Click

    Dim age As Integer
    Dim group As String

    age = 0
    group = ""

    age = InputBox("Please enter the age of the footballer")

    If age < 8 Then
        group = "Beginner"
    ElseIf age < 13 Then
        group = "Junior"
    Else
        group = "Youth"
    End If

    txtOutput.AppendText("You are in the " & group & " training group")

End Sub

End Class
```

Declare Variable

Initialise Variable

Get Input

Selection

Display Output

Testing

- Make sure the user can enter an age
- Make sure the correct output is displayed for an age under 8
- Make sure the output is displayed correctly for an age between 8 and 12.
- Make sure the output is displayed correctly for an age over 12

Run the program and complete the test table

Test	Type	Input	Expected Output	Actual Result	Result (pass/fail)
1	Normal	Age = 5	Group = “Beginner”	Group =	
2	Normal	Age = 10	Group = “Junior”	Group =	
3	Normal	Age = 15	Group = “Youth”	Group =	
4	Extreme	Age = 8	Group = “Junior”	Group =	
5	Extreme	Age = 13	Group = “Youth”	Group =	

Password

Program Specification

A program is required that will ask the user to enter the password to their social network account. The program should allow the user to enter their password.

If the user has entered the correct password then a message should indicate that access has been granted, otherwise the user should be told that the password was incorrect.

General Knowledge

Program Specification

A general knowledge program is to be created by a team of programmers. Your task is to create one question for the program. The question should be displayed on screen and the user asked to enter the correct answer.

If the user enters the answer correctly then an appropriate message should be displayed. If the user enters the wrong answer then a message should indicate this also.

Golf Tournament

Program Specification

Players in a golf tournament must score a **maximum** of 140 strokes over their first two rounds in order to qualify, or make “the cut”, for the final part of the competition.

A program is required to ask the golfer to enter their scores, separately, for round one and round two. The program should then calculate the total and display a message to indicate whether the golfer has qualified or not.

Talent Contest

Program Specification

Acts in a talent contest receive votes from 9 judges. Judges are asked whether an act should progress to the next round and they can vote either yes or no. A program is required to ask how many yes votes and how many no votes were given.

If there are more yes votes than no votes, a message should inform the act that they have progresses, otherwise they should be told that they are out of the contest.

Top 40

Program Specification

A program is required to identify where a band has reached in the UK Singles Chart. The program should ask for the band/singer name and which position they are in the chart.

If the position is between 1 and 10, the program should display a message indicating that the band are in the top 10. If the band are between 11 and 40 then a top 40 message should be displayed.

If the band are in any other position, a message should tell the user that the band are not in the top 40.

Sports Day

Program Specification

A program is required to record a runner's times in the 10,000 metres race. The runner is asked to enter their 3 most recent times (in minutes) for the race. The program should then calculate the overall average time from these three races.

If the average time is less than 30 minutes, a message should tell the runner they are doing very well. Otherwise a message should inform the user that they need to run faster to get closer to the world record.

Level 3: Iteration (Loops)

Learning Intentions

Outcome 1
We are learning how to analyse and design programs that can make use of repetition (loops).

Outcome 2
We are learning how write and debug programs that use fixed and conditional loops to reduce the lines of code required.

Outcome 3
We are learning how to test our programs using normal, extreme and exceptional test data.

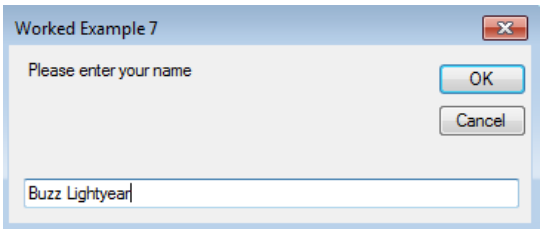
Success Criteria

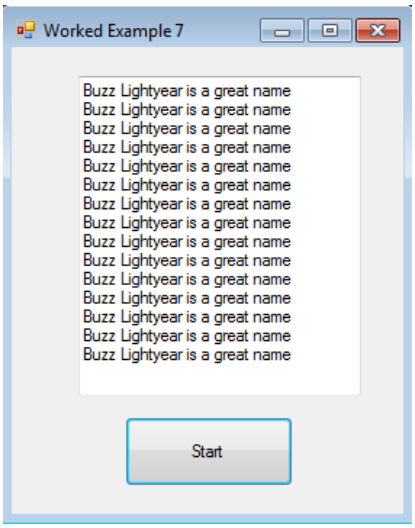
Outcome 1	I can analyse a problem and identify inputs, processes and outputs.
	I can analyse a problem and identify required variables and data types
	I can design solutions to problems that require iteration (loops) using flow charts and/or structure diagrams
	I can design solutions to problems that require iteration (loops) using pseudocode
Outcome 2	With some help, I can write programs involving loops.
	I can decide when I need to use either fixed or conditional loops in my programs.
	I can make the correct use of fixed and conditional loops to create working programs involved repetition.
	I can debug code on my own by correcting syntax, execution and logic errors.
Outcome 3	I can carry out testing of my program to prove it works
	I can create a test plan that involves normal, extreme and exceptional test data and record results of testing accurately
	I can evaluate the success of my program in terms of fitness for purpose and readability

Worked Example 7: Repeating Name — Level 3a

Problem Specification

A program is required to ask a user to enter their name. The program should then fill the screen with a message repeating 15 times which tells users that their name is great.





Analysis

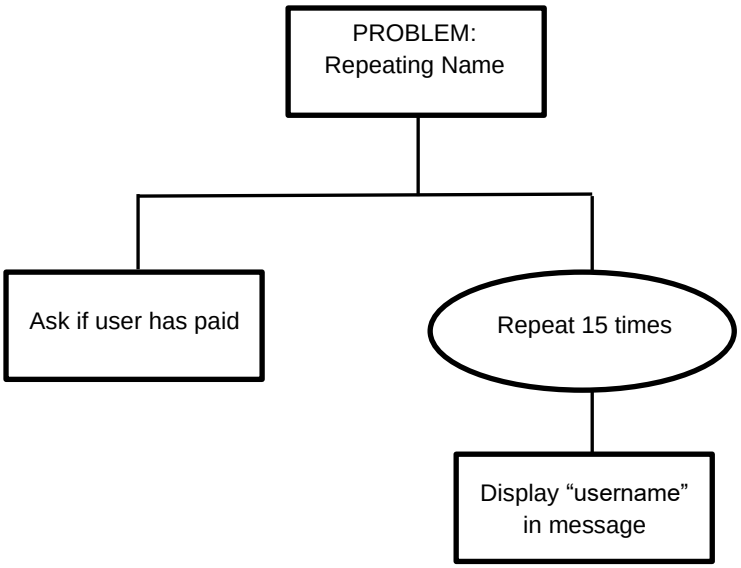
Inputs	Process	Outputs
• Username	Display user's name in a message 15 times	• Username

Data Items	Data Types
Username	String

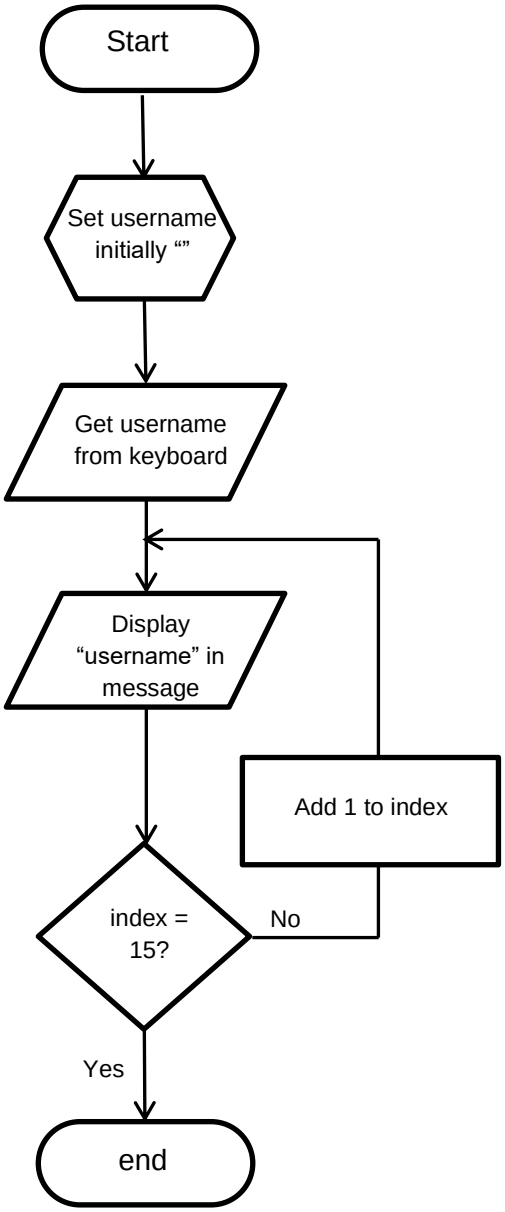
Username will contain text

Design

Structure Diagram



Flow Chart



Pseudocode

Algorithm

1. Initialise variable
2. Ask user for their name
3. Display username 15 times

Refinements

2.1 Ask user to enter name

3.1 Start fixed loop 15 times

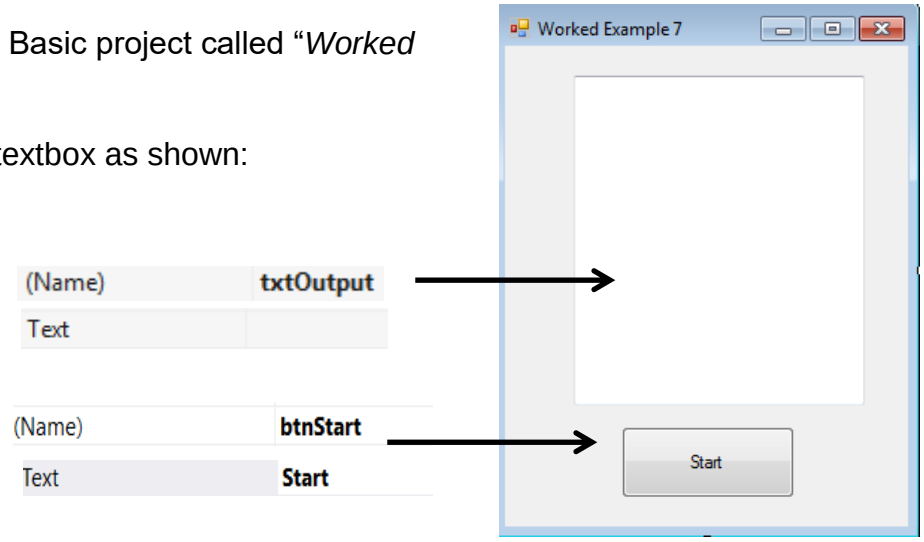
3.2 display username, " is a great name."

3.3 End fixed loop

Implementation

Create a new Visual Basic project called "*Worked Example 7*"

Add a button and a textbox as shown:



Double click the ***button*** and add the code below:

```
Public Class Form1

Private Sub btnStart_Click(sender As Object, e As EventArgs) Handles btnStart.Click

        Dim username As String

        username = ""

        username = InputBox("Please enter your name")


        For index = 1 To 15

            txtoutput.appendtext(username & " is a great name " & vbNewLine)

        Next

End Sub

End Class
```


Testing

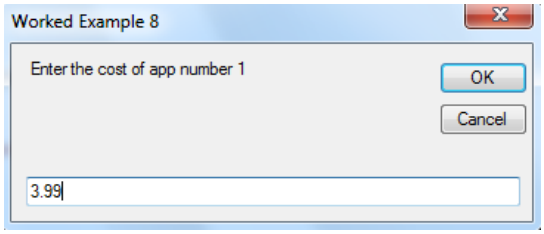
- Make sure the user can enter their name correctly
- Make sure the message containing the username is displayed 15 times.

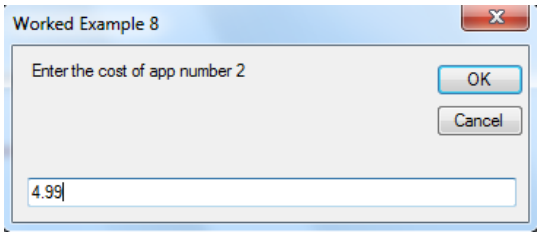
Worked Example 8: Mobile App Cost (Running Total) Level 3a

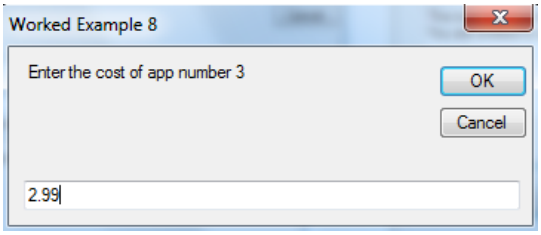
Problem Specification

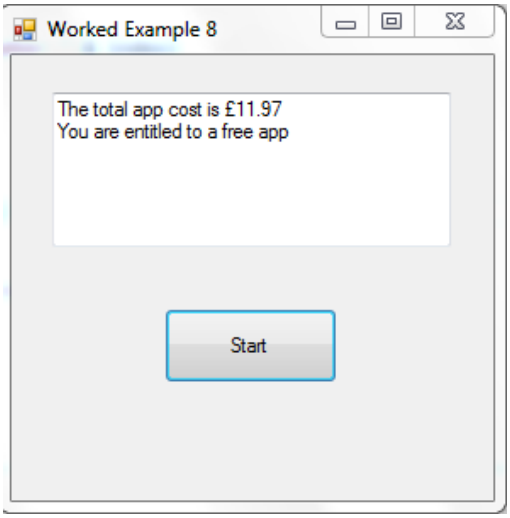
A program is required to ask a user to enter the cost of 3 apps they wish to buy. The program should calculate the total cost of the three apps and display it on screen.

If the total cost is more than £10 then a message should let the user know they are entitled to a free app of their choice.









Analysis

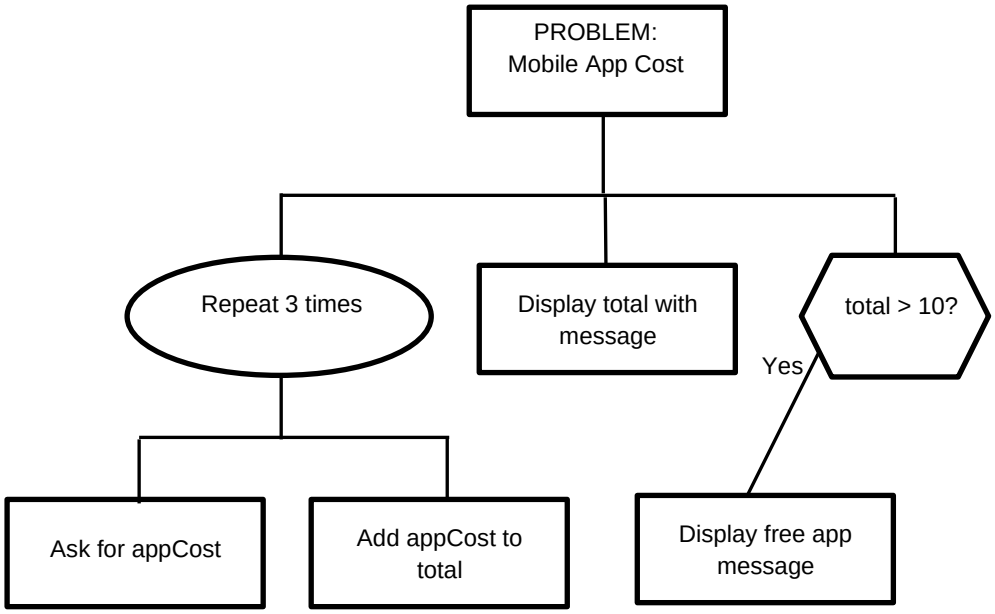
Inputs	Process	Outputs
• App Costs	• Calculate Total cost • Decide if total cost is more than10	• Total Cost • Free App Message

Data Items	Data Types
App Costs	Single
Total Cost	Single

Prices have 2 decimal places

Design

Structure Diagram



Pseudocode

Algorithm

1. Initialise variables
2. Ask user for three app costs and add to total
3. Display total cost
4. Display free message if total is over 10

Refinements

2.1 Start fixed loop repeating 3 times

2.2 ask user for app cost

2.3 add app price to total cost

2.4 End fixed loop

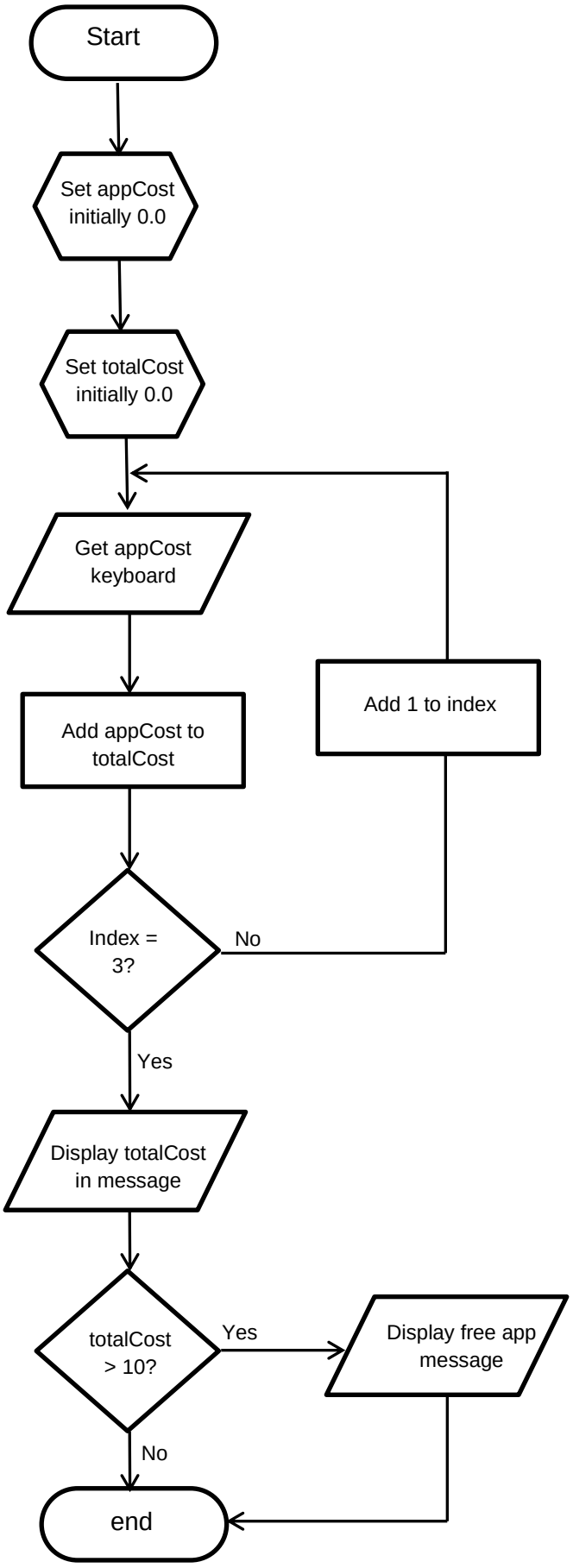
3.1 Display “The total app cost is £”, app cost

4.1 If app cost is more than 10 Then

4.2 Display “You are entitled to a free app”

4.3 End If

Flow Chart


Start
Set appCost initially 0.0
Set totalCost initially 0.0
Get appCost keyboard
Add appCost to totalCost
Add 1 to index
Index = 3?
No
Yes
Display totalCost in message
totalCost > 10?
Yes
Display free app message
No
end

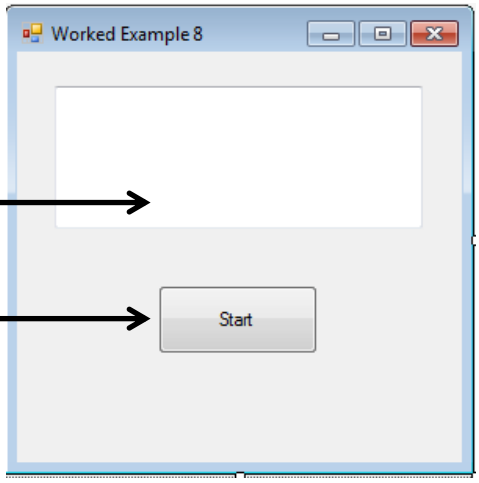
Implementation

Create a new Visual Basic project called "*Worked Example 8*"

Add a button and a textbox as shown:

(Name)	txtOutput
Text	

(Name)	btnStart
Text	Start



Double click the ***button*** and add the code below:

```
Public Class Form1

Private Sub btnStart_Click(sender As Object, e As EventArgs) Handles btnStart.Click

        Dim appCost As Single
        Dim totalCost As Single

        appCost = 0.0
        totalCost = 0.0

        For index = 1 To 3

              appCost = InputBox("Enter the cost of app number " & index)

              totalCost = totalCost + appCost

        Next

      txtOutput.AppendText("The total app cost is £" & totalCost & vbNewLine)


      If totalCost > 10 Then

           txtOutput.AppendText("You are entitled to a free app")

      End If


End Sub

End Class
```

Declare Variable

Initialise Variable

Fixed Loop

Selection

Testing

- Make sure the user can enter 3 costs
- Make sure the total cost is calculated and displayed correct
- Make the free app message is displayed if total is over 10
- Make sure free app message is *not* displayed if the total is 10 or below.

Run the program and complete the test table

Test	Type	Input	Expected Output	Actual Result	Result (pass/fail)
1	Normal	appCost1=2.99 appCost2=1.99 appCost3=4.99	totalCost = 9.97 No free app	totalCost = Free app / No Free app	
2	Normal	appCost1=4.00 appCost2=1.99 appCost3=5.00	totalCost = 10.99 Free app	totalCost = Free app / No Free app	
3	Extreme	appCost1=3.00 appCost2=3.00 appCost3= 4.01	totalCost = 10.01 Free app	totalCost = Free app / No Free app	
4	Extreme	appCost1=3.00 appCost2=3.00 appCost3= 4.00	totalCost = 10.00 No free app	totalCost = Free app / No Free app	

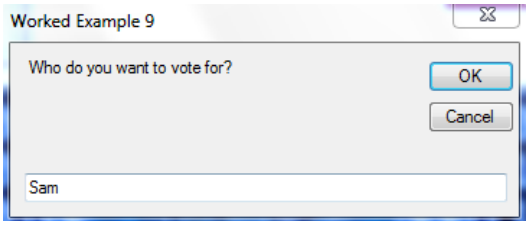
Worked Example 9: Club Captain (Counter) Level 3a

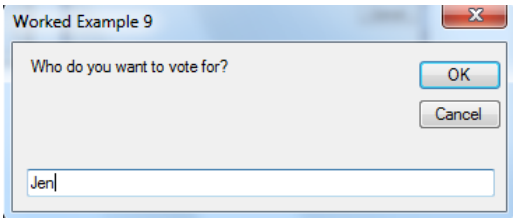
Problem Specification

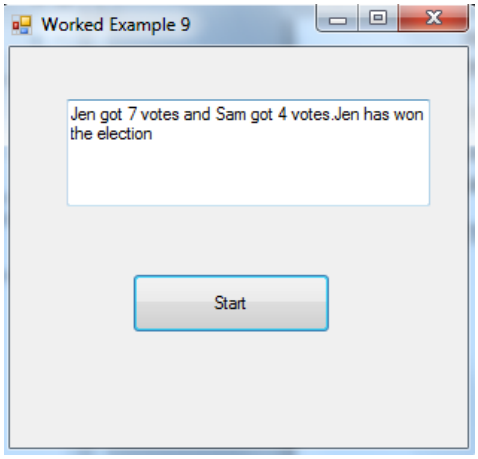
Sam and Jen want to be the captain of their social club. A program is required to count the number of votes obtained by each candidate during the club captain election.

There are 11 voters and each one should be asked whether they wish to vote for Sam or Jen. The program should count the number of votes for each candidate and display the results.

The program should also identify on screen the candidate who has won the election.







Analysis

Inputs	Process	Outputs
• Vote	• Count votes for each candidate • Decide who has won	• Sam counter • Jen counter • Winner message

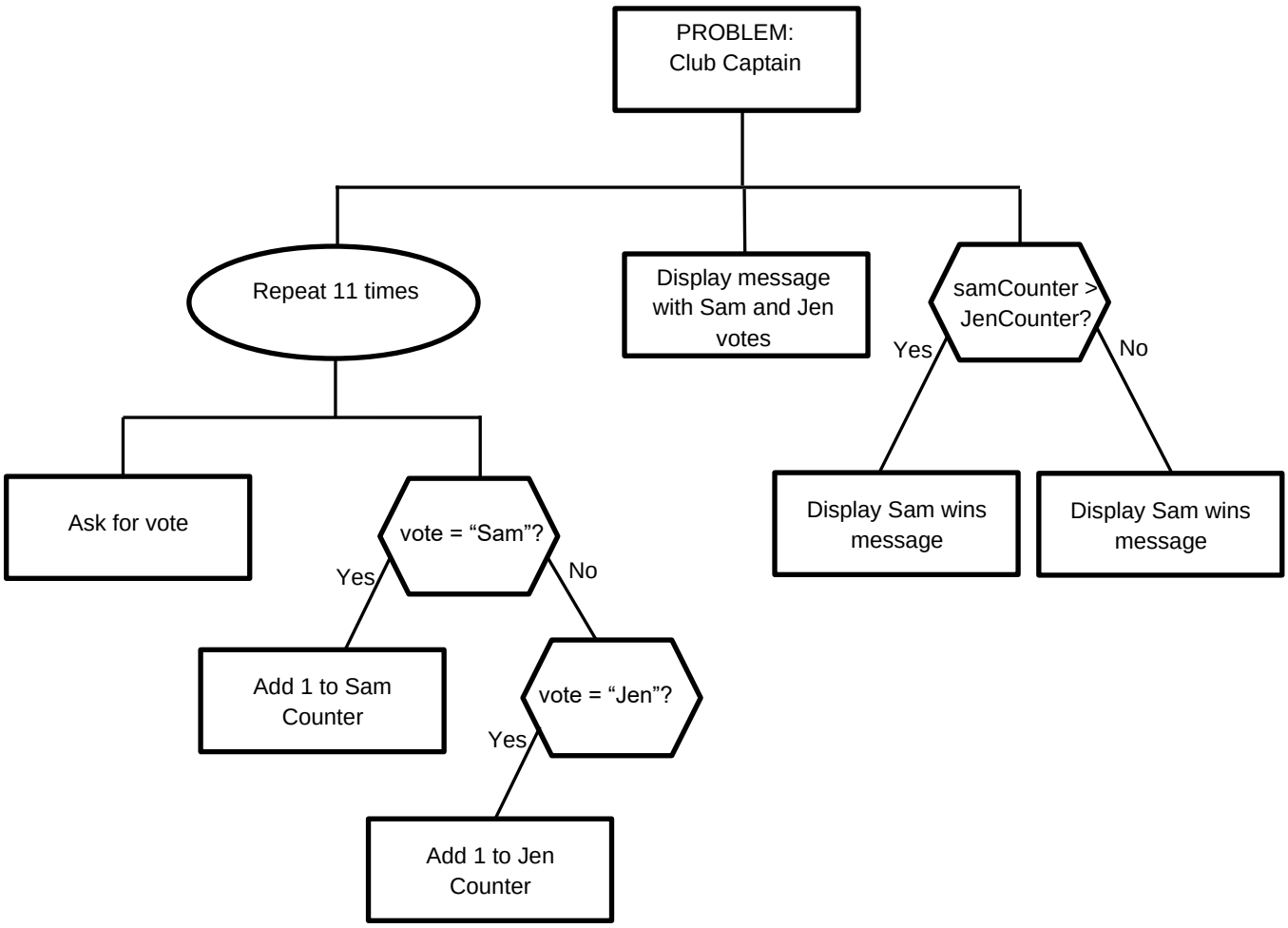
Data Items	Data Types
Vote	String
Sam Counter	Integer
Jen Counter	Integer

Vote will be Jen or Sam

Counters will be WHOLE NUMBERS

Design

Structure Diagram



Pseudocode

Algorithm

1. Initialise variables
2. Ask user for 11 votes and update counters
3. Display votes
4. Decide winner and display

Refinements

2.1 Start fixed loop repeating 11 times

2.2 ask user for vote

2.3 If vote is for Sam Then

2.4 add 1 to Sam Counter

2.5 Else If vote is for Jen Then

2.6 add 1 to Jen Counter

2.7 End If

2.8 End fixed loop

3.1 Display “Jen got ”, jenCounter, “ votes and Sam got ”, samCounter, “ votes”

4.1 If samCounter is greater than jenCounter Then

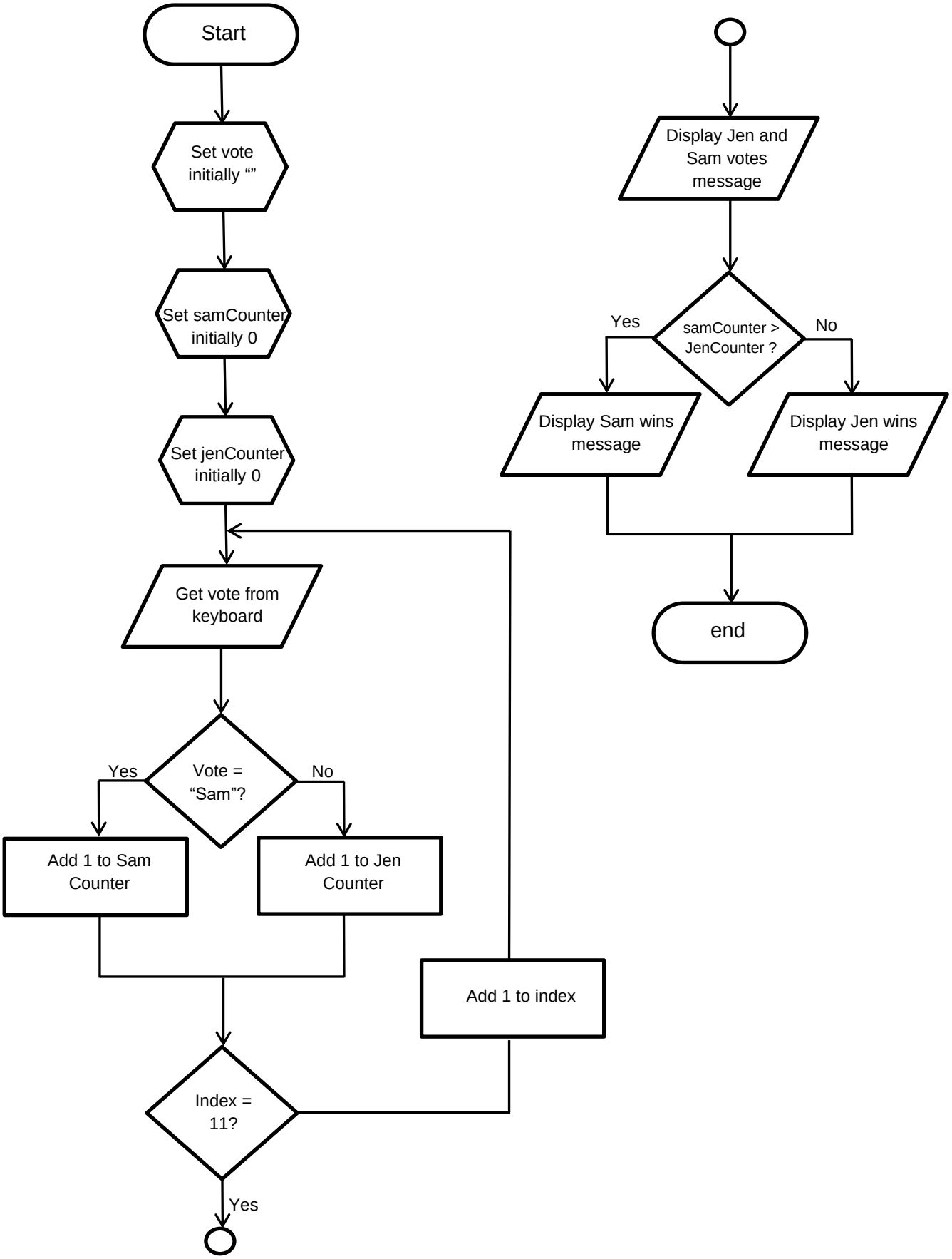
4.2 Display “Sam has won the election”

4.3 Else

4.4 Display “Jen has won the election”

4.5 End If

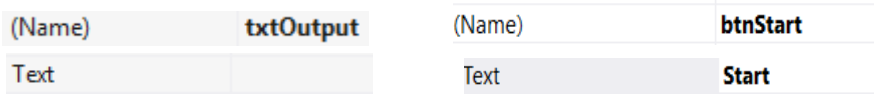
Flow Chart


Start
Set vote initially “”
Set samCounter initially 0
Set jenCounter initially 0
Get vote from keyboard
Vote = “Sam”?
Yes
No
Add 1 to Sam Counter
Add 1 to Jen Counter
Add 1 to index
Index = 11?
Yes
Display Jen and Sam votes message
samCounter > JenCounter ?
Yes
No
Display Sam wins message
Display Jen wins message
end

Implementation

Create a new Visual Basic project called "*Worked Example 9*"

Add a button called **btnStart** and a textbox called **txtOutput** as shown:



Double click the ***button*** and add the code below:

```
Public Class Form1

Private Sub btnStart_Click(sender As Object, e As EventArgs) Handles btnStart.Click

    Dim vote As String
    Dim samCounter As Integer
    Dim jenCOunter As Integer

    vote = ""
    samCounter = 0
    jenCOunter = 0

    For index = 1 To 11

        vote = InputBox("Who do you want to vote for?")

        If vote = "Sam" Then

            samCounter = samCounter + 1

        ElseIf vote = "Jen" Then

            jenCOunter = jenCOunter + 1

        End If

    Next

  txtOutput.AppendText("Jen got " & jenCOunter & " votes and Sam got " & samCounter & " votes.")


  If samCounter > jenCOunter Then

        txtOutput.AppendText("Sam has won the election")

  Else
        txtOutput.AppendText("Jen has won the election")

  End If

End Sub

End Class
```

Declare Variable

Initialise Variable

Selection

Fixed Loop

Selection

Testing

- Make sure the user can enter 11 votes
- Make sure the votes for Sam are counted and displayed correct
- Make sure the votes for Jen are counted and displayed correct
- Make sure the correct winner is identified

Run the program and complete the test table

Test	Type	Input		Expected Output	Actual Result	Result (pass/fail)
1	Normal	Vote1= Jen Vote2= Jen Vote3= Sam Vote4= Jen Vote5= Jen	Vote6= Jen Vote7= Sam Vote8= Jen Vote9= Sam Vote10= Jen Vote 11= Sam	samCounter = 4 jenCounter = 7 Winner is Jen	samCounter = jenCounter = 7 Winner is	
2	Normal	Vote1= Sam Vote2= Sam Vote3= Sam Vote4= Jen Vote5= Jen	Vote6= Sam Vote7= Sam Vote8= Sam Vote9= Sam Vote10= Sam Vote 11= Sam	samCounter = 9 jenCounter = 2 Winner is Sam	samCounter = jenCounter = Winner is	
3	Extreme	Vote1= Sam Vote2= Sam Vote3= Sam Vote4= Jen Vote5= Jen	Vote6= Sam Vote7= Sam Vote8= Sam Vote9= Jen Vote10= Jen Vote 11= Jen	samCounter = 6 jenCounter = 5 Winner is Sam	samCounter = jenCounter = Winner is	
3	Extreme	Vote1= Jen Vote2= Sam Vote3= Sam Vote4= Jen Vote5= Jen	Vote6= Sam Vote7= Sam Vote8= Sam Vote9= Jen Vote10= Jen Vote 11= Jen	samCounter = 5 jenCounter = 6 Winner is Jen	samCounter = jenCounter = Winner is	

Holiday

Program Specification

A program is required to ask the user where they are going on holiday and for how long (in days). The program should then display a message on screen 10 times that includes this information.

For example: “I am going to Spain on holiday for 14 days”.

Concert Tickets

Program Specification

A program should ask the user to enter the cost of tickets to five concerts. The program should calculate the total cost of the five concerts and display this on screen.

If the total cost is more than £500 then a message should let the user know they have qualified for a free ticket to a concert of their choice.

Race

Program Specification

An athlete is training for the Commonwealth Games. She requires a program to help her to monitor her progress. The program should ask the athlete how many races she has run. It should then ask her to enter the time taken (in minutes) for each of these races.

The program should then calculate and display the average time to run all four races. If the average is below 4 minutes then she has qualified and a message should be displayed to indicate this.

Talent Contest

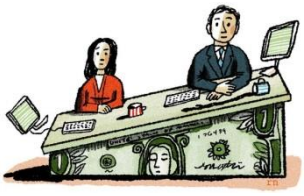
Program Specification

A travel agent would like to compare the popularity of holiday destinations. A program is required to ask the user which destination they prefer from the two most popular choices of USA and Spain.

The program should ask 11 customers and display the totals for each destination. The program should also display a message indicating which destination was the most popular.

Bonus

Program Specification



Workers are paid £8.50 per hour and they work for five days per week. A program is required to ask the workers to enter the number of hours they worked in each of the five days, one day at a time.

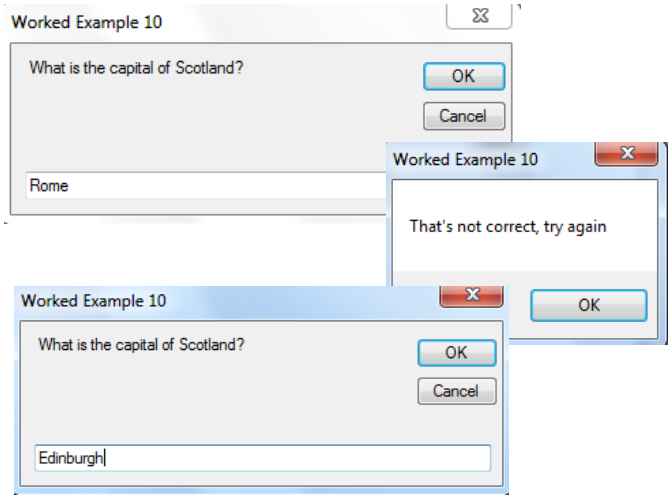
The program should add up the total number of hours worked and calculate the weekly wage for the worker. If the worker has worked for more than 40 hours in the week, a £50 bonus should be added.

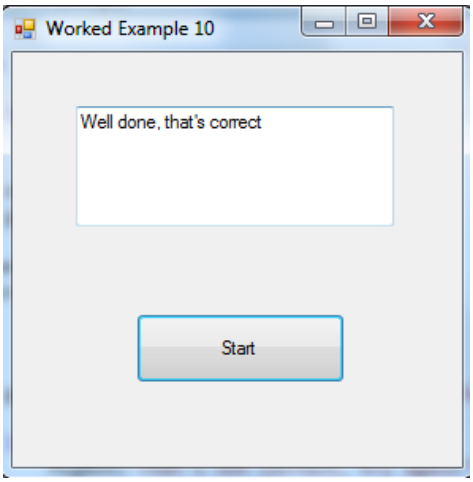
The program should display the total hours worked, the total wages (without bonus), a message indicating if a bonus was awarded or not, and a final total with any bonus added.

Worked Example 10: Guess — Level 3b

Problem Specification

A program is required to ask the user to guess the answer to a question. The program will keep asking the user to guess until they get the correct answer.





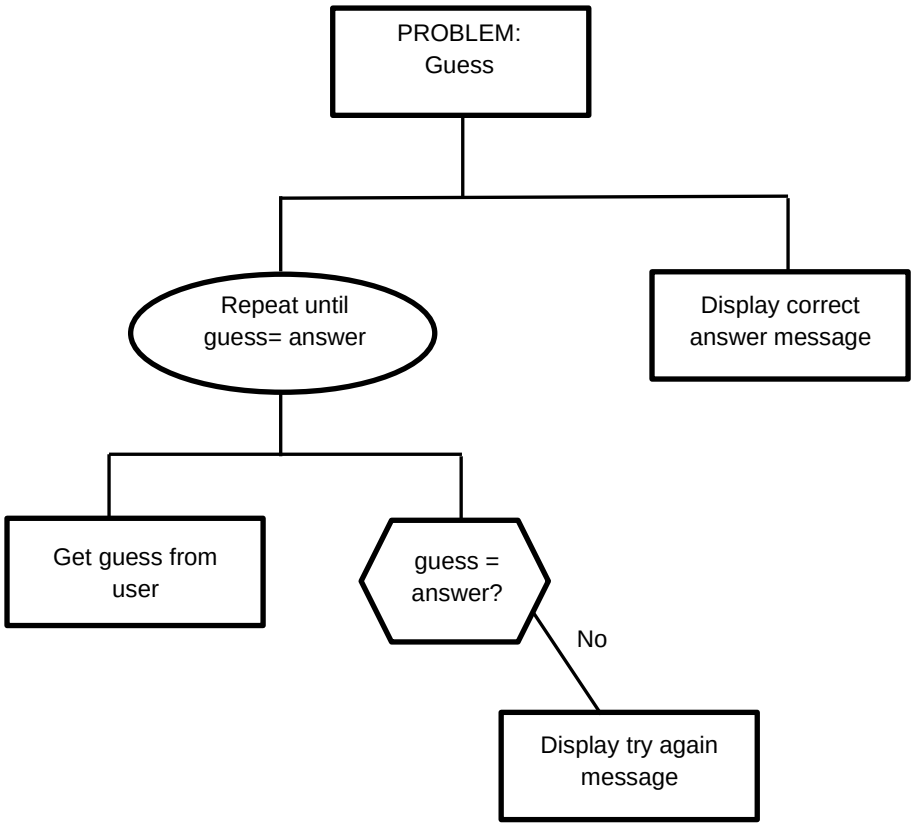
Analysis

Inputs	Process	Outputs
• Guess	Ask question until answer is correct	Message when incorrect Message when correct

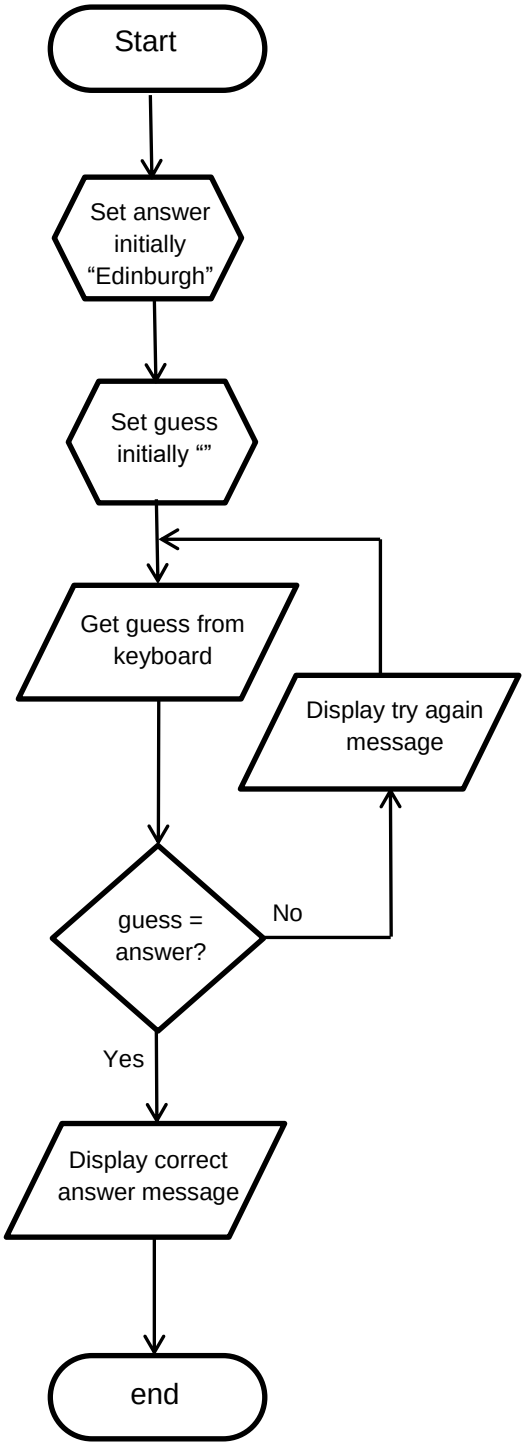
Data Items	Data Types	
Guess	String	Guess will contain text
Answer	String	Answer will contain text

Design

Structure Diagram



Flow Chart



Pseudocode

Algorithm

1. Initialise variable
2. Get user guess
3. Display correct answer message

Refinements

2.1 Start conditional loop

2.2 get guess from user

2.3 If guess does not match answer Then

2.4 display “That’s not correct, try again”

2.5 End If

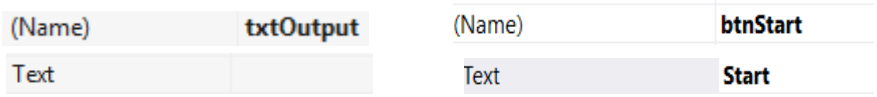
2.6 Repeat until guess matches answer

3.1 Display “Well done, that’s correct”

Implementation

Create a new Visual Basic project called "*Worked Example 10*"

Add a button called **btnStart** and a textbox called **txtOutput** as shown:



Double click the ***button*** and add the code below:

```
Public Class Form1

Private Sub btnStart_Click(sender As Object, e As EventArgs) Handles btnStart.Click

        Dim answer As String
        Dim guess As String

        answer = "Edinburgh"
        guess = ""

        Do

            guess = InputBox("What is the capital of Scotland?")

            If guess <> answer Then
                MsgBox("That's not correct, try again")
            End If

        Loop Until guess = answer


        txtOutput.AppendText("Well done, that's correct")

End Sub

End Class
```

Declare Variable

Initialise Variable

Conditional Loop

Testing

- Make sure the user can enter their guess correctly
- Make sure the try again message is displayed when a wrong guess is entered
- Make sure the user is asked to guess again each time a wrong guess is entered
- Make sure the well done message is displayed when a correct guess is entered

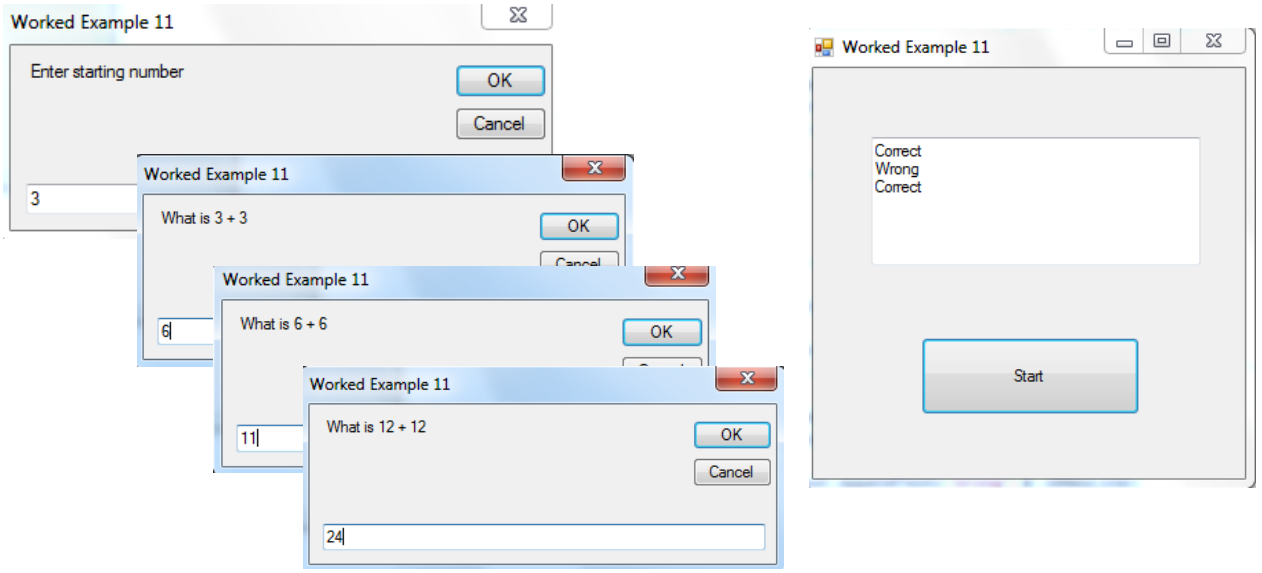
Run the program and complete the test table

Test	Type	Input	Expected Output	Actual Result	Result (pass/fail)
1	Normal	Guess= “Edinburgh”	Well done message displayed		
2	Exceptional	Guess = “Rome”	Try again message displayed User asked to input again		
3	Exceptional	Guess = “Paris”	Try again message displayed User asked to input again		

Problem Specification

A program is required to test user's ability to double numbers. The program should ask the user for a starting number (e.g. 4). The program will then ask the user to enter the answer to 4+4. For each question, the user will be informed if they got it correct or wrong.

The answer to the previous question will then become the new starting number (so 8 in this case). The program will repeat these questions until the answer to a question is more than 100.



Analysis

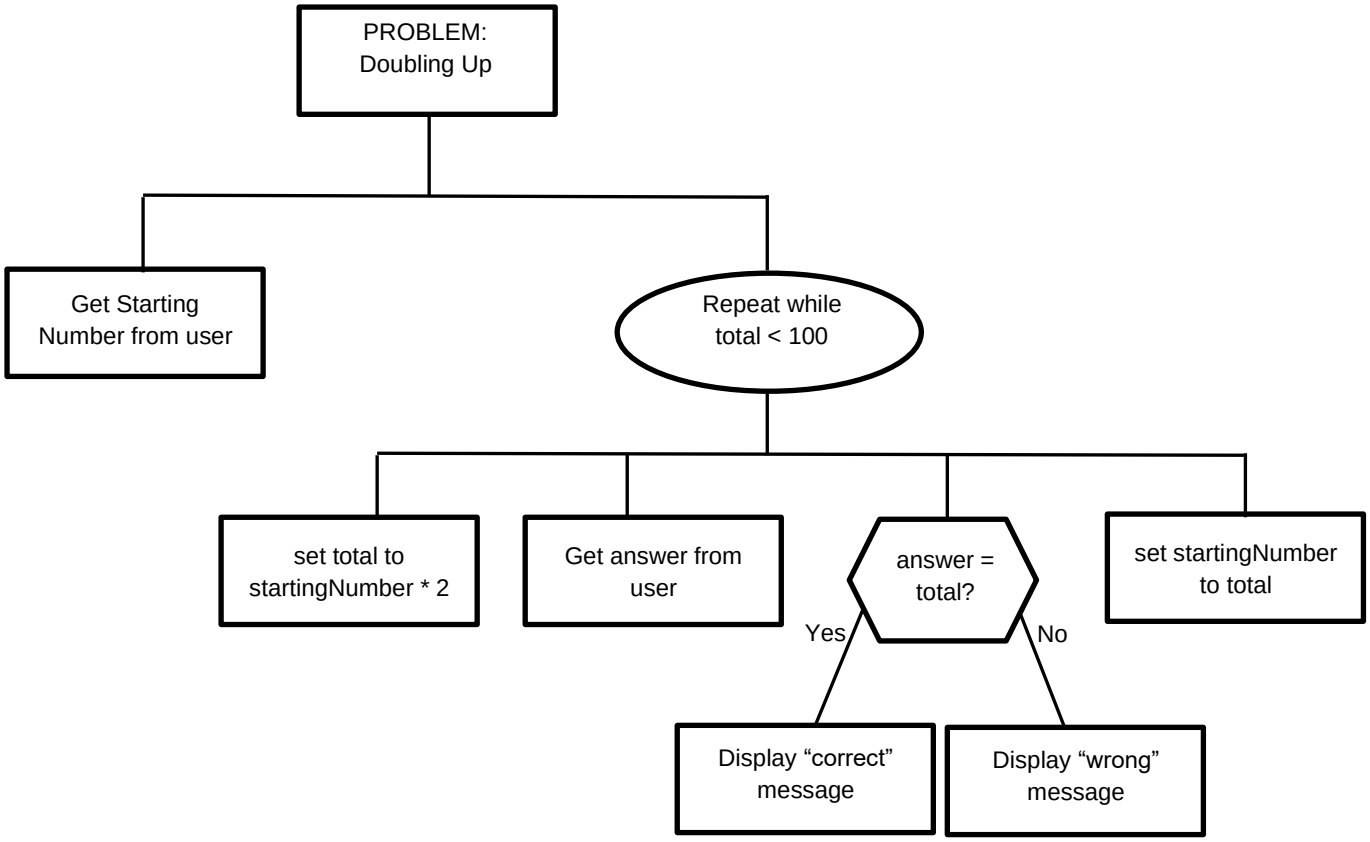
Inputs	Process	Outputs
• Starting Number • User Answer	Calculate total Check user answer against total Pick new starting number Repeat until total is over 100	Message when correct Message when wrong

Data Items	Data Types
Starting Number	Integer
User Answer	Integer
Total	Integer

Numbers will all be WHOLE NUMBERS

Design

Structure Diagram



Pseudocode

Algorithm

1. Initialise variable
2. Get starting number
3. Ask questions

Refinements

2.1 Ask user to enter starting nuymber

3.1 Repeat conditional loop while total is under 100

3.2 set total to starting number times 2

3.3 get answer from user

3.4 If answer = total THEN

3.5 display “Correct”

3.6 Else

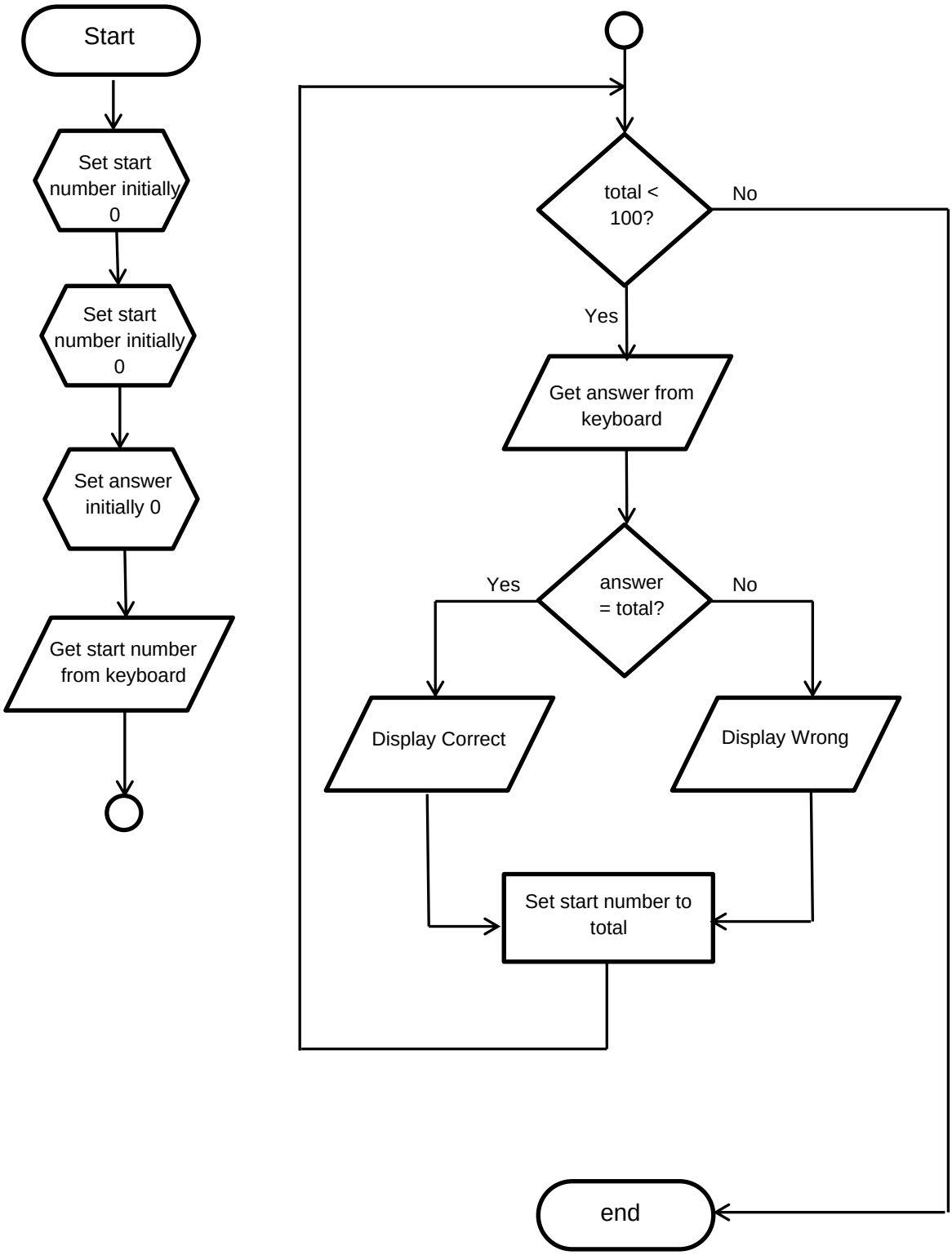
3.5 display “Wrong”

3.6 End If

3.7 set starting number to total

3.8 End conditional loop

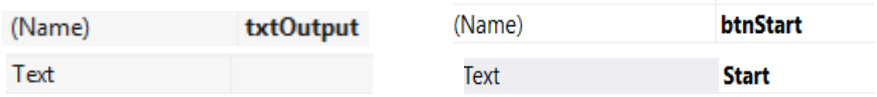
Flow Chart


Start
Set start number initially 0
Set start number initially 0
Set answer initially 0
Get start number from keyboard
total < 100?
No
Yes
Get answer from keyboard
answer = total?
Yes
No
Display Correct
Display Wrong
Set start number to total
end

Implementation

Create a new Visual Basic project called "*Worked Example 11*"

Add a button called **btnStart** and a textbox called **txtOutput** as shown:



Double click the ***button*** and add the code below:

```
Public Class Form1

Private Sub btnStart_Click(sender As Object, e As EventArgs) Handles btnStart.Click

        Dim startingnumber As Integer
        Dim total As Integer
        Dim useranswer As Integer

        startingnumber = 0
        total = 0
        useranswer = 0

        startingnumber = InputBox("Enter starting number")

        Do While startingnumber < 100
            total = startingnumber * 2
            useranswer = InputBox("What is " & startingnumber & " + " & startingnumber)

            If useranswer = total Then
                txtOutput.AppendText("Correct" & vbNewLine)
            Else
                txtOutput.AppendText("Wrong" & vbNewLine)
            End If

            startingnumber = total
        Loop

        txtOutput.AppendText("Well done, that's correct")

End Sub
End Class
```

Declare Variable

Initialise Variable

Conditional Loop start

Conditional Loop end

Testing

- Make sure the user can enter a starting number
- Make sure the question is displayed as starting value + starting value
- Make sure the answer is correct if the user enters double the starting value
- Make sure each question uses the answer from the previous
- Make sure the questions stops before the starting value reaches 100

Run the program and complete the test table

Test	Type	Input	Expected Output	Actual Result	Result (pass/fail)
1	Normal	startingValue= 8 answer= 16 answer= 32 answer= 64 answer= 128	Correct Correct Correct Correct		
2	Normal	startingValue= 8 answer= 12 answer= 4 answer= 112 answer= 3	Wrong Wrong Wrong Wrong		
3	Normal	startingValue= 30 answer= 60 answer= 87 answer= 180	Correct Wrong Correct		
4	Exceptional	startingValue= 120	No Output (loop doesn’t start)		

Bonus Ball

Program Specification

A prize draw at a school fete asks users to select a “bonus ball”. Each ball is numbered from 0 to 49 and users must choose one ball to be their bonus ball. A program is required to ask users to enter a whole number below 50.

If the user enters a number of 50 or above, the program should display and error message and ask them to re-enter until they choose a valid number.

Once a valid number is entered, the user should be informed that they have successfully entered the prize draw.

Mystery Band

Program Specification

A program should give the user a clue to the name of a mystery band and asks them to guess the name the band. The program should continue to run until the user enters the correct guess. The program should count the number of attempts and display this in a suitable message.

Any user who guesses the name of the band in fewer than 5 guesses should be informed that they have been entered into a prize draw to win tickets to see the band.

Scrobble

Program Specification

Scrobble is a game where players earn points for making words with different letters. Letter have different points allocated to them.

A program is required to calculate the total score for letters in a word in Scrobble. The program should repeatedly ask the user to enter the score for a letter in the word until they type 0 (zero).

Once all letters have been entered the program will display the total score for the word. If the word is worth more than 10 points, a 5 point bonus is awarded. A message will tell the user whether they got a bonus or not and what their final score is.

Level 4: Input Validation and Pre-defined Functions

Learning Intentions

Outcome 1
We are learning how to analyse and design programs that can make use of pre-defined functions and input validation.

Outcome 2
We are learning how write and debug programs that use fixed and conditional loops to reduce the lines of code required.

Outcome 3
We are learning how to test programs that validate input by using normal, extreme and exceptional test data.

Success Criteria

	I can design solutions to problems that require input validation using flow charts and/or structure diagrams
	I can design solutions to problems that require input validation using pseudocode
Outcome 2	I can make use of Length, Random and Round pre-defined functions within my programs
	I can write and adapt the steps of an input validation standard algorithm in a range of programs
	I understand when to use AND, OR and NOT logical operators within conditional statements.
	I can debug code on my own by correcting syntax, execution and logic errors.
Outcome 3	I can create a test plan to test input validation using normal, extreme and exceptional test data and record results of testing accurately
	I can evaluate the success of my program in terms of fitness for purpose and readability

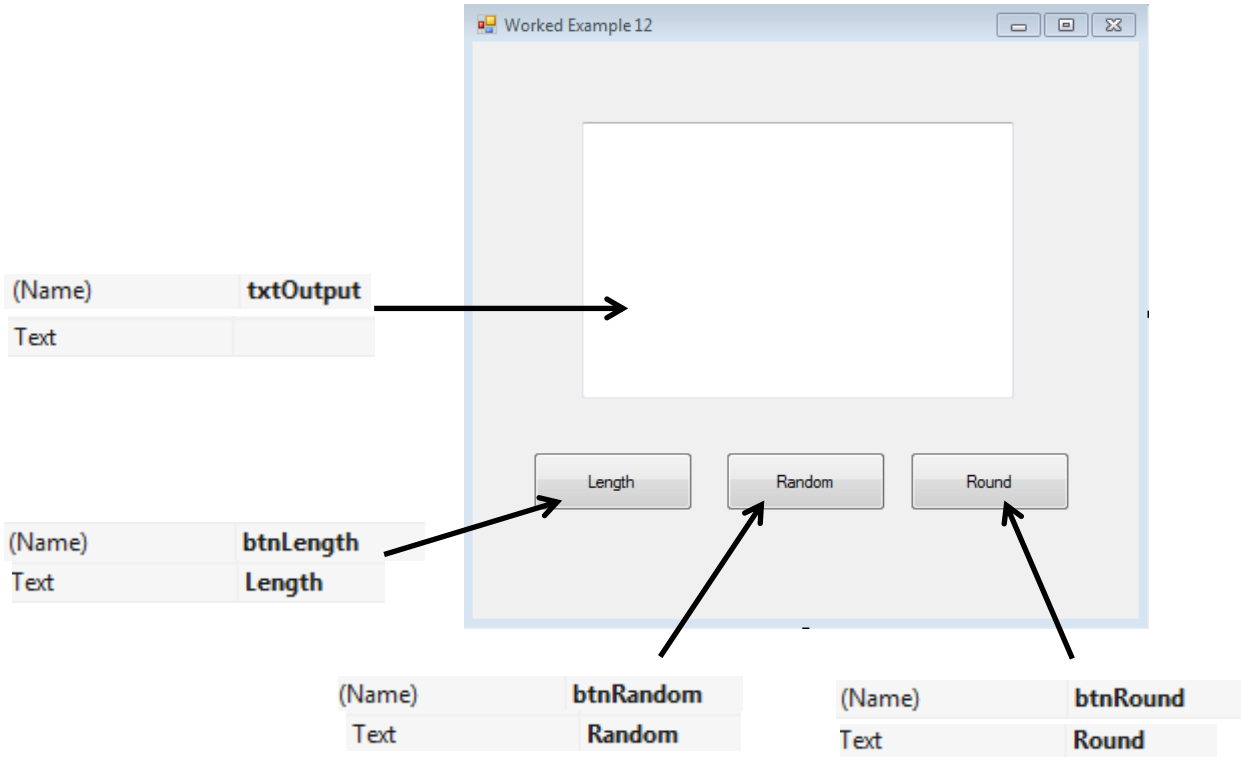
Practical Demonstration

This worked example will demonstrate how pre-defined functions can be used to save time when coding. These functions run sections of code that have already been written and allow us to carry out complex tasks more easily.

Implementation

Create a new Visual Basic project called "*Worked Example 12*"

Add 3 buttons (**btnLength, btnRandom, btnRound**) and 1 textbox (**txtOutput**) as shown:



Double click the ***Length button*** and add the code below:

```
Public Class Form1


Private Sub btnLength_Click(sender As Object, e As EventArgs) Handles btnLength.Click

        Dim word As String
        Dim wordLength As Integer

        word = ""
        wordLength = 0


        word = InputBox("Please enter a word")

        wordLength = Len(word)

        txtOutput.AppendText("The word has " & wordLength & " characters")


End Sub

End Class
```

LENGTH pre-defined function

Double click the ***Random button*** and add the code below:

```
Public Class Form1


Private Sub btnRandom_Click(sender As Object, e As EventArgs) Handles btnRandom.Click

        Dim randomNum As Integer

        randomNum = 0


        Randomize()

        randomNum = Int(Rnd() * 10) + 1


        txtOutput.AppendText("A random number between 1 and 10 is " & randomNum)


End Sub

End Class
```

RANDOM pre-defined function

Double click the ***Length button*** and add the code below:

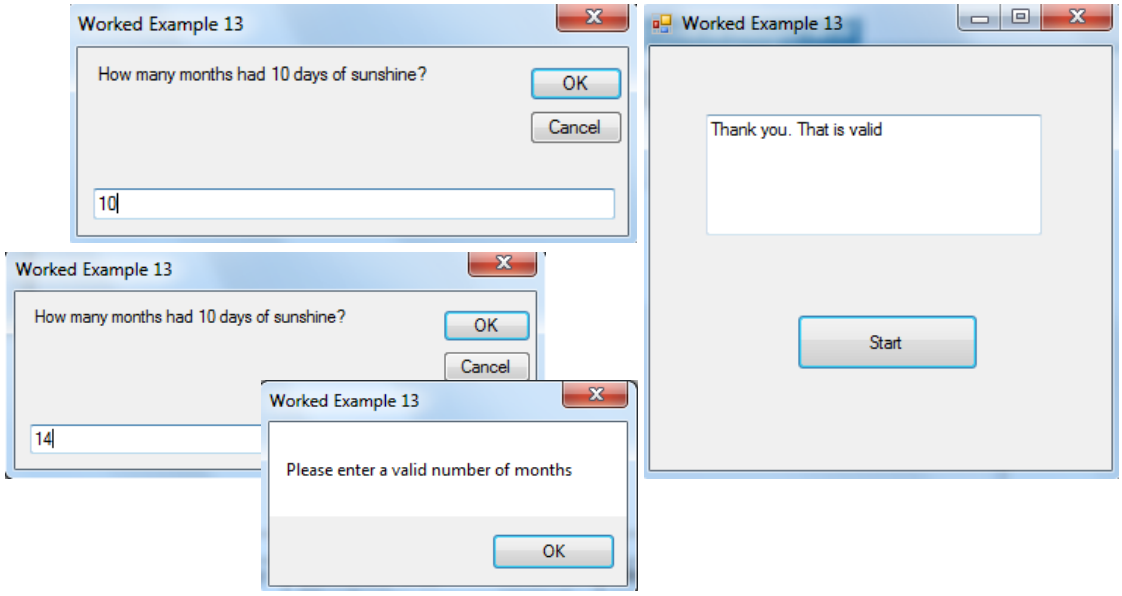
```
Public Class Form1


Private Sub btnLength_Click(sender As Object, e As EventArgs) Handles btnLength.Click


        Dim realNumber As Single
        Dim roundedNumber As Single

        realNumber = 0.0
        roundedNumber = 0.0


        realNumber = InputBox("Enter a real number")


        roundedNumber = Math.Round(realNumber, 1)


        txtOutput.AppendText("Your number rounded to 1 decimal place is " & roundedNumber)


End Sub

End Class
```

ROUND pre-defined function

Problem Specification

A program is required to ask a user to enter the number of months of the year in which there has been at least 10 days of sunshine. The minimum number of months will be 0 and the maximum will be 12.

The program should use ***Input Validation*** to make sure that users cannot enter a number that is less than zero or more than 12.



Analysis

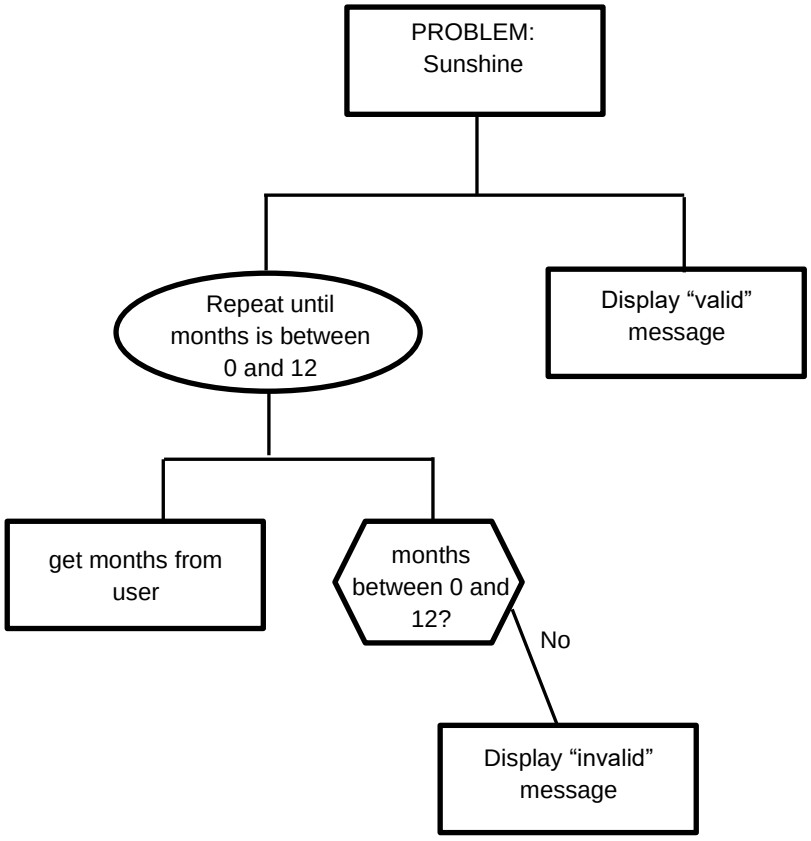
Inputs	Process	Outputs
• Months	Check months entered are valid	Invalid message Valid message

Data Items	Data Types
Months	Integer

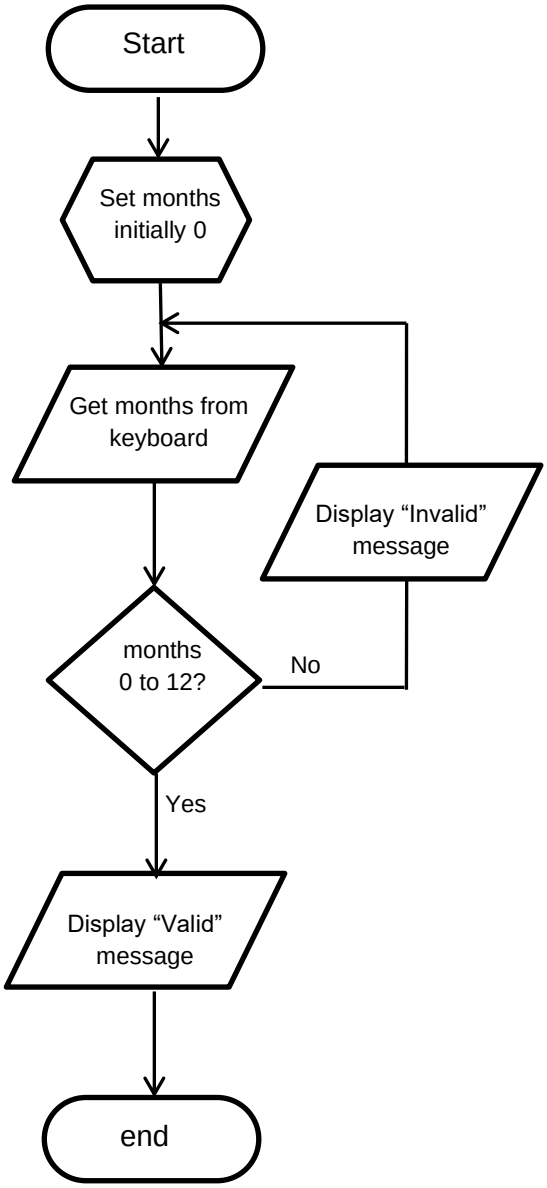
Month will be a WHOLE NUMBER

Design

Structure Diagram



Flow Chart



Pseudocode

Algorithm

1. Initialise variables
2. Get valid months
3. Display valid message

Refinements

2.1 Start conditional loop

2.2 get months from user

2.3 If months is not between 0 and 12 Then

2.4 display “Please enter a valid month”

2.5 End If

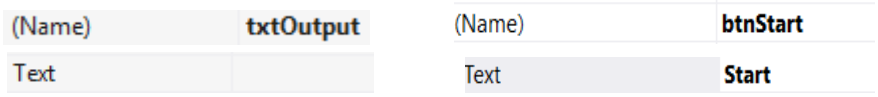
2.6 repeat loop until months is between 0 and 12

3.1 Display “Thank you. That is valid”

Implementation

Create a new Visual Basic project called "*Worked Example 13*"

Add a button called **btnStart** and a textbox called **txtOutput** as shown:



Double click the ***button*** and add the code below:

```
Public Class Form1

Private Sub btnStart_Click(sender As Object, e As EventArgs) Handles btnStart.Click

   Dim months As Integer


   Do

      months = InputBox("How many months had 10 days of sunshine?")

      If months < 0 Or months > 12 Then

         MsgBox("Please enter a valid number of months")

      End If

   Loop Until months >= 0 And months <= 12



   txtOutput.AppendText("Thank you. That is valid")

End Sub

End Class
```

Input Validation Algorithm

Testing

- Make sure the user can enter a number of months
- Make sure the invalid message appears if the number is not between 0 and 12
- Make sure the user has to re-enter if the number is not between 0 and 12
- Make sure the loop terminates and the valid message appears if 0 to 12 is entered

Run the program and complete the test table

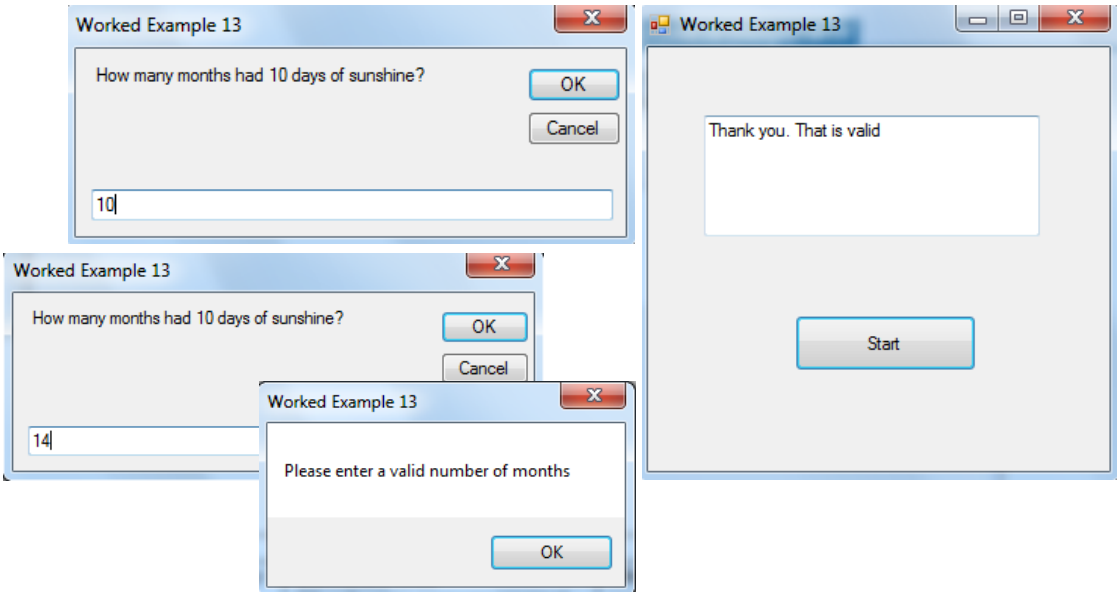
Test	Type	Input	Expected Output	Actual Result	Result (pass/fail)
1	Normal	months= 6	Valid message		
2	Normal	months= 10	Valid message		
3	Normal	months= 2	Valid message		
4	Extreme	months= 0	Valid message		
5	Extreme	months= 12	Valid message		
6	Exceptional	months= 13	Invalid message – re-enter		
7	Exceptional	months= -1	Invalid message – re-enter		

Worked Example 14: Cinema Trip — Level 4

Problem Specification

A program is required to ask a user if they would like to book a place on a cinema trip. The program should use Input Validation to make sure the user enters either Yes or No, no other response should be accepted.

If the user does not enter Yes or No, a warning message should appear and they should be asked to re-enter. When a valid response is received, the user should be told that their response has been accepted.



Analysis

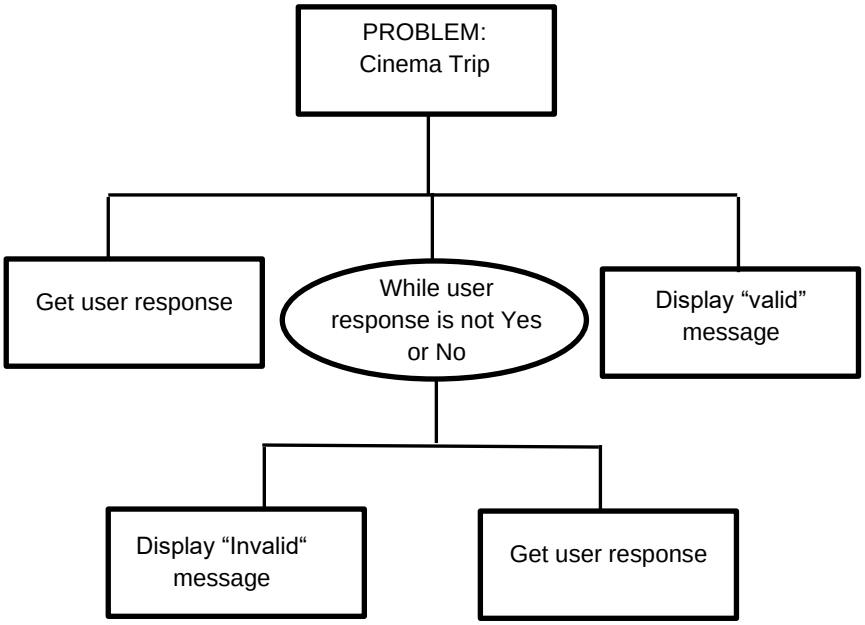
Inputs	Process	Outputs
• User Response	Check user response entered is valid	Invalid message Valid message

Data Items	Data Types
User Response	String

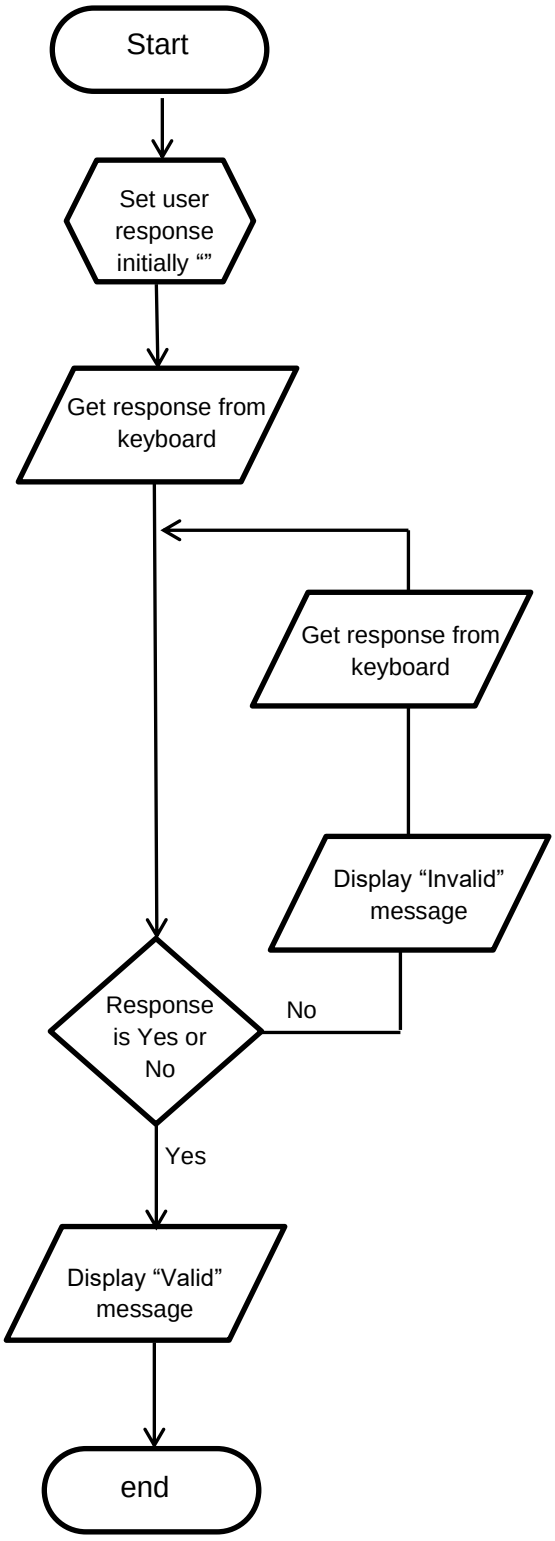
User Response will be either YES or NO

Design

Structure Diagram



Flow Chart



Pseudocode

Algorithm

1. Initialise variables
2. Get valid response
3. Display valid message

Refinements

2.1 get response from user

2.2 Start loop repeating while response is not Yes or No

2.3 display "Please enter only yes or no"

2.4 get response from user

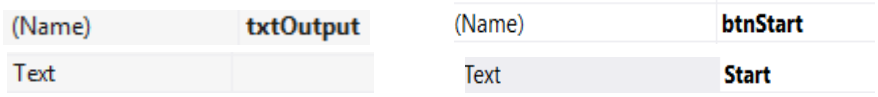
2.5 End conditional loop

3.1 Display "Your response has been accepted"

Implementation

Create a new Visual Basic project called "*Worked Example 14*"

Add a button called **btnStart** and a textbox called **txtOutput** as shown:



Double click the ***button*** and add the code below:

```
Public Class Form1

Private Sub btnStart_Click(sender As Object, e As EventArgs) Handles btnStart.Click

   Dim userResponse As String

   userResponse = ""


   userResponse = InputBox("Do you want to go to the cinema (Yes/No)?")

   While Not (userResponse = "Yes" Or userResponse = "No")

       MsgBox("Please enter only Yes or No")

       userResponse = InputBox("Do you want to go to the cinema (Yes/No)?")

   End While



   txtOutput.AppendText("Your response has been accepted")


End Sub

End Class
```

Input Validation Algorithm

Testing

- Make sure the user can enter Yes or No
- Make sure the invalid message appears if the response is not Yes or No
- Make sure the user has to re-enter if the response is not Yes or No
- Make sure the loop terminates and the valid message appears if Yes or No is entered

Run the program and complete the test table

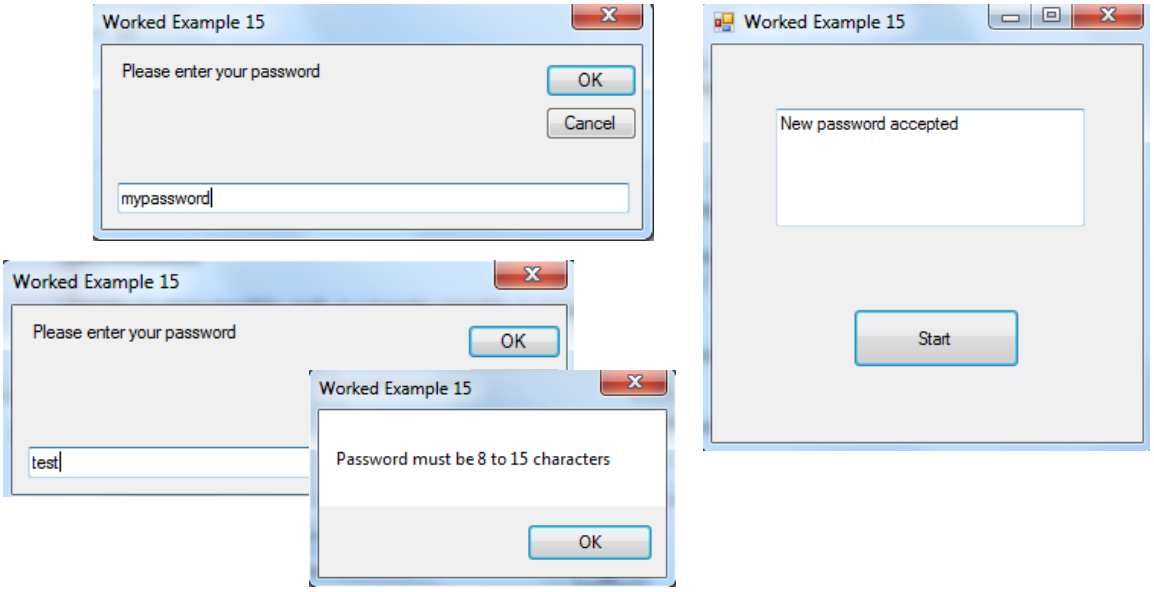
Test	Type	Input	Expected Output	Actual Result	Result (pass/fail)
1	Normal	userResponse = Yes	Valid message		
2	Normal	userResponse= No	Valid message		
3	Exceptional	userResponse= Maybe	Invalid message – Re-enter		

Worked Example 15: Password Reset Level 4

Problem Specification

A program is required to ask a user to create a new password. The password must contain between 8 and 15 characters.

The program should use ***Input Validation*** to make sure that users cannot enter a password that does not contain between 8 and 15 character.



Analysis

Inputs	Process	Outputs
• password	Check password is valid	Invalid message Valid message

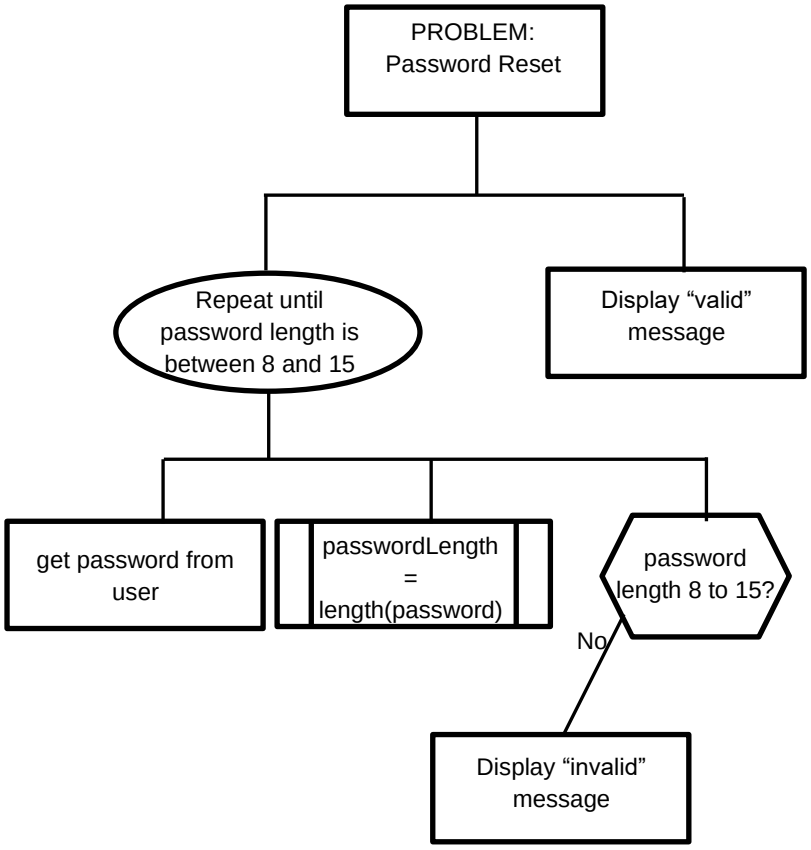
Data Items	Data Types
Password	String
Password Length	Integer

Password will be text

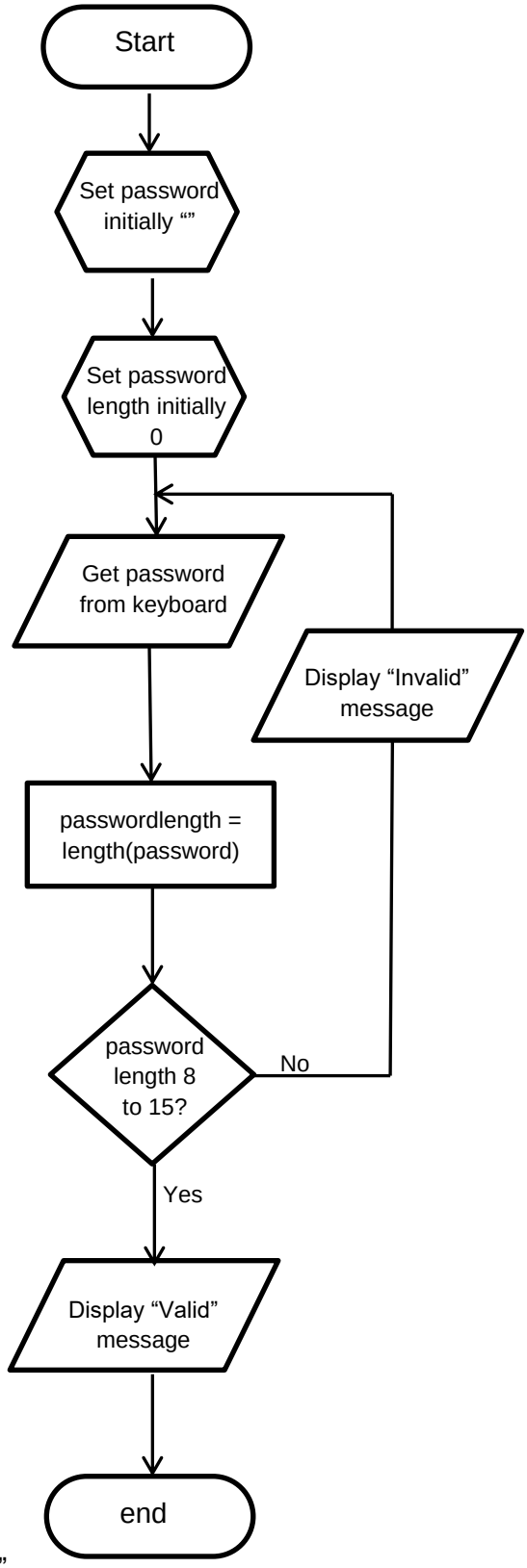
Password Length will be a WHOLE NUMBER

Design

Structure Diagram



Flow Chart



Pseudocode

Algorithm

1. Initialise variables
2. Get valid password
3. Display valid message

Refinements

2.1 Start conditional loop

2.2 get password from user

2.3 passwordlength equals length of password

2.3 If passwordlength is not between 8 and 15 Then

2.4 display “Password must be 8 to 15 characters”

2.5 End If

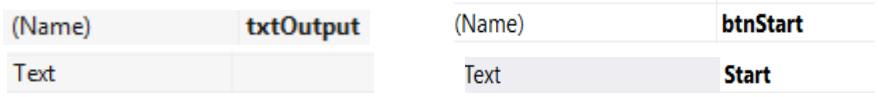
2.6 repeat loop until passwordlength is between 8 and 15

3.1 Display “New password accepted”

Implementation

Create a new Visual Basic project called "*Worked Example 15*"

Add a button called **btnStart** and a textbox called **txtOutput** as shown:



Double click the ***button*** and add the code below:

```
Public Class Form1


Private Sub btnStart_Click(sender As Object, e As EventArgs) Handles btnStart.Click


   Dim password As String
   Dim passwordLength As Integer


   Do
       password = InputBox("Please enter your password")

       passwordLength = Len(password)


       If passwordLength < 8 Or passwordLength > 15 Then

           MsgBox("Password must be 8 to 15 characters")

       End If

   Loop Until passwordLength >= 8 And passwordLength <= 15




   txtOutput.AppendText("New password accepted")



End Sub

End Class
```

Input Validation Algorithm

Testing

- Make sure the user can enter a password
- Make sure the invalid message appears if the password is not between 8 and 15 characters
- Make sure the user has to re-enter if the password is not between 8 and 15 characters
- Make sure the loop terminates and the valid message appears if 8 to 15 characters is entered

Run the program and complete the test table

Test	Type	Input	Expected Output	Actual Result	Result (pass/fail)
1	Normal	password= “mytestpassword”	Valid message		
2	Normal	password= “testpassword”	Valid message		
3	Extreme	password = “password”	Valid message		
4	Extreme	password= “mytestpassword1”	Valid message		
6	Exceptional	password= “test”	Invalid message – re-enter		
7	Exceptional	password= “mysuperduperpassword”	Invalid message – re-enter		

Guess Number

Program Specification

A program is required to prompt the user to guess a whole number between 1 and 20 which is randomly chosen by the computer. The user's guess should be validated.

If the guess is incorrect, the user should be told if the target number is bigger or smaller. This process should continue until the target number is guessed correctly.

The user should then be told how many valid guesses were made.

Language Course

Program Specification

Students taking a language course must pass examinations in French and German or Spanish to pass.

A pass is half marks or more. A program is required to take in the three marks which are validated as being whole numbers between 0 and 30.

The output should indicate if the student has passed or failed and gives the percentage mark to 1 decimal place.

Booking Reference

Program Specification

A program is required to check that a holiday booking reference is valid. Booking references all have exactly 7 characters (e.g TEN1563).

The program should ask the user to enter their booking reference. If the booking reference is not valid, the user should be prompted to re-enter until a valid booking reference is entered.

When a valid booking reference is entered, the program should display a message to the user indicating this.

School Dinners

Program Specification

The manager of a school cafeteria wants to use a computer system to calculate how much each customer has to pay. Members of staff have to pay VAT on their purchases but pupils do not.

If the customer is a member of staff then the program will calculate the VAT and add it to the total cost.

VAT is calculated using the formula: **VAT = 0.2 × total cost**

The system requires the following inputs:

- How many items the customer has to pay for
- The price of each item in pounds
- Whether the customer is a pupil or a member of staff (P for pupil and S for staff)

The output from the program should display the total cost of purchases, the type of customer, the amount of VAT to be paid and the final cost **e.g.**

- Total cost of purchases: £2.38
- Type of customer: S
- VAT: £0.42
- Final Cost: £2.80

All currency values should be rounded to 2 decimal places.

Code Breaking

Program Specification

A program is required to assist with breaking a code. The code cracking is done manually but the program should allow each cracked letter to be entered one at a time until the complete message is known.

Each time a letter from the code is cracked, it is entered into the program. The program will keep asking for letters until the user enters the # symbol.

When # is entered, the final cracked message is displayed on screen

An example of how the program should work is shown.

```
Please enter a letter: H
Please enter a letter: e
Please enter a letter: l
Please enter a letter: l
Please enter a letter: l
Please enter a letter: #

The message is Hello
```

Level 5: Arrays

Learning Intentions

Outcome 1
We are learning how to analyse and design programs that can make use of array data structures.

Outcome 2
We are learning how write and debug programs that combine array data structures with all of the other coding techniques we have learned.

Outcome 3
We are learning how to fully test programs that store data in arrays

Success Criteria

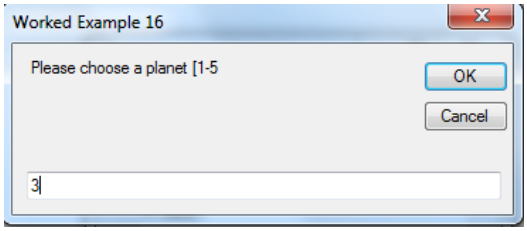
	I can design solutions to problems, using flow charts and/or structure diagrams, that require multiple values to be stored in arrays
	I can design solutions to problems, using pseudocode, that require multiple values to be stored in arrays
Outcome 2	I can declare 1D arrays, with appropriate sizes and data types
	I can write code that inputs, validates, processes and outputs data stored in arrays.
	I can debug code on my own by correcting syntax, execution and logic errors.
Outcome 3	I can create a test plan to test programs involving arrays using a full range of test data.
	I can evaluate the success of my program in terms of fitness for purpose and readability

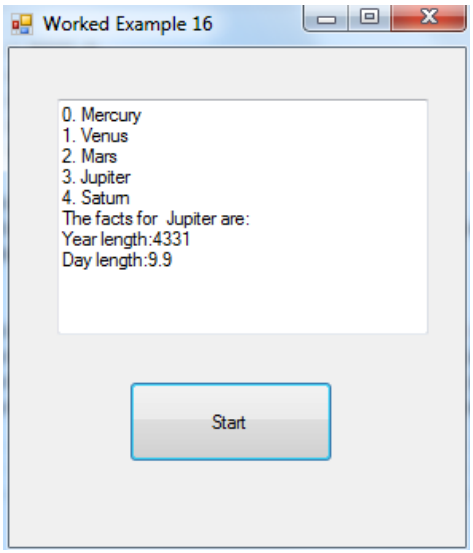
Worked Example 16: Solar System — Level 5

Problem Specification

Create a program containing the details of 5 planets in the solar system. The program should store details of each planet's name, days to orbit the sun (year) and hours to rotate (day).

A numbered list should be displayed with each planet's name and when the user selects a number, the details of that planet are displayed.





Analysis

Inputs	Process	Outputs
• User Choice	Display menu	Planet details

Data Items	Data Types
User Choice	Integer
Planets()	String
Years()	Integer
Days()	Single

A WHOLE NUMBER (0-4)

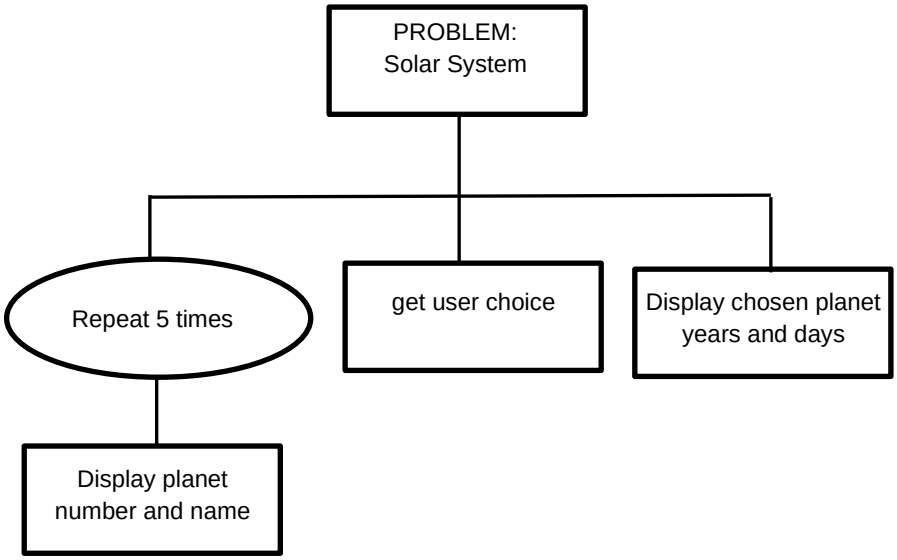
Planet Name will be Text

Years will be a WHOLE NUMBER

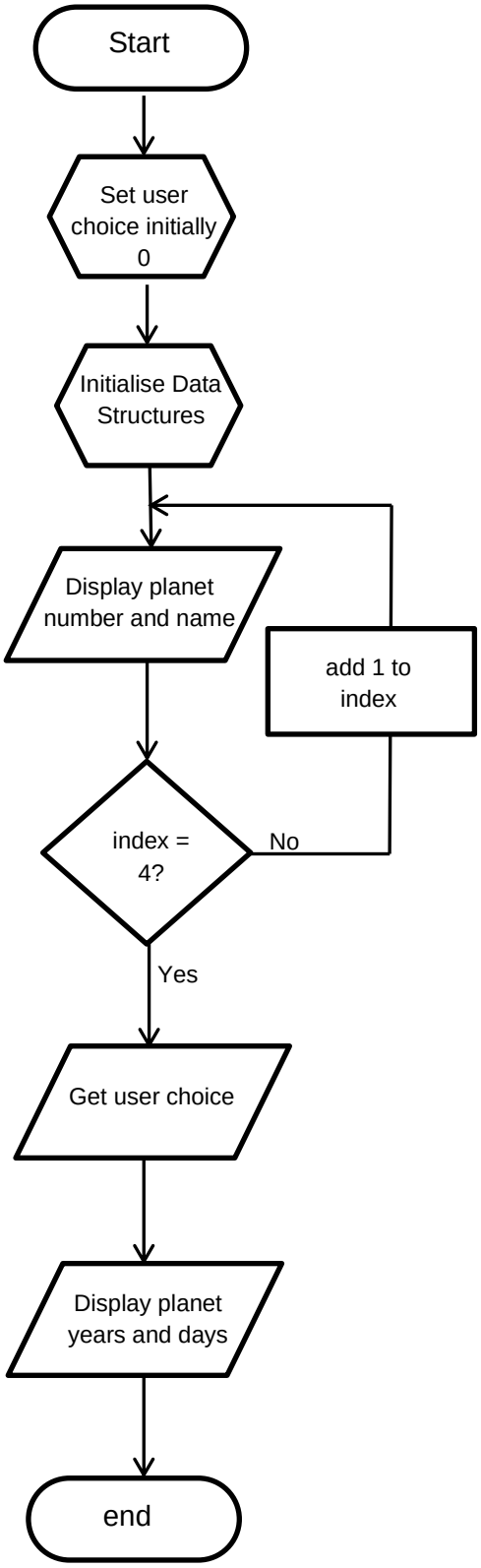
Days will be a REAL NUMBER

Design

Structure Diagram



Flow Chart



Pseudocode

Algorithm

1. Initialise variables and data structures
2. Display menu options
3. Get user choice
4. Display planet details

Refinements

2.1 Start fixed loop for each planet

2.2 display planet()

2.3 End fixed loop

3.1 get user choice from keyboard

4.1 Display “The facts for ”,planets(userChoice), “ are:”

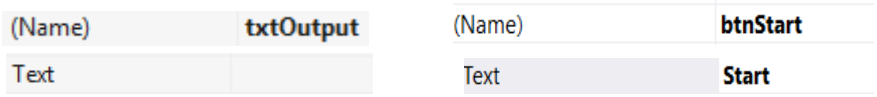
4.2 Display “Year length: ”,years(userChoice)

4.3 Display “Day length ”,days(userChoice

Implementation

Create a new Visual Basic project called "*Worked Example 16*"

Add a button called **btnStart** and a textbox called **txtOutput** as shown:



Double click the ***button*** and add the code below:

```
Public Class Form1


Private Sub btnStart_Click(sender As Object, e As EventArgs) Handles btnStart.Click

    Dim planets(5) As String
    Dim years(5) As Integer
    Dim days(5) As Single

    Dim user_choice As Single

    planets(0) = "Mercury"
    planets(1) = "Venus"
    planets(2) = "Mars"
    planets(3) = "Jupiter"
    planets(4) = "Saturn"

    years(0) = 88
    years(1) = 224
    years(2) = 687
    years(3) = 4331
    years(4) = 10747

    days(0) = 4222.6
    days(1) = 2802.0
    days(2) = 24.7
    days(3) = 9.9
    days(4) = 10.7


    For index = 0 To 4
        txtOutput.AppendText(index & ". " & planets(index) & vbNewLine)
    Next index


    user_choice = InputBox("Please choose a planet [0-4]")


    txtOutput.AppendText("The facts for  " & planets(user_choice) & " are:" & vbNewLine)
    txtOutput.AppendText("Year length:" & years(user_choice) & vbNewLine)
    txtOutput.AppendText("Day length:" & days(user_choice) & vbNewLine)


End Sub

End Class
```

Declare Arrays to store 5 items each

Initialise Arrays

Testing

- Make sure the menu is displayed correctly
- Make the user can choose a number
- Make sure the correct planet details are displayed

Run the program and complete the test table

Test	Type	Input	Expected Output	Actual Result	Result (pass/fail)
1	Normal	userchoice= 2	Days and Years for Mars		
2	Normal	userchoice= 3	Days and Years for Jupiter		
3	Extreme	userchoice= 0	Days and Years for Mercury		
4	Extreme	userchoice= 4	Days and Years for Saturn		
6	Exceptional	userchoice= 5	Invalid message – re-enter		
7	Exceptional	userchoice= -1	Invalid message – re-enter		

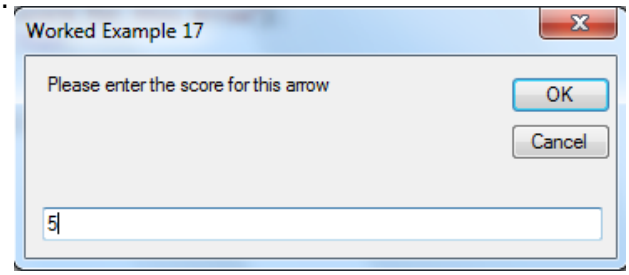
Can you fix the program to make sure tests 6 and 7 pass?

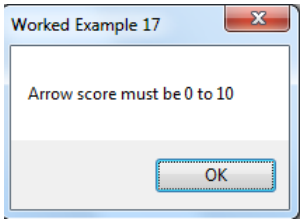
Worked Example 17: Archery Score — Level 5

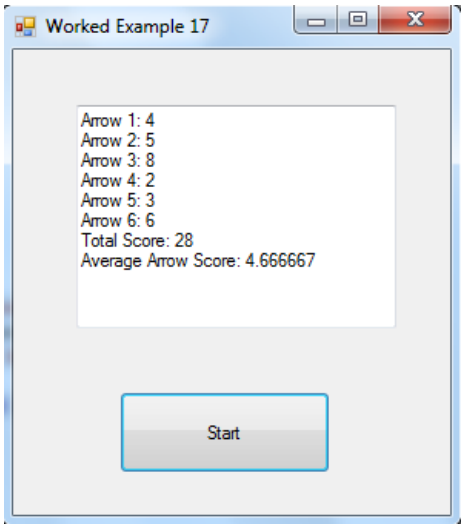
Problem Specification

Create a program to calculate the total and average score achieved from their 6 arrows. For each arrow, the score can be between 0 and 10 points.

The program should ask a competitor for their score for each arrow, one at a time. Once all arrow scores have been entered, the score for each arrow should be displayed on screen together with the total score and average score per arrow.







Analysis

Inputs	Process	Outputs
• Arrow Scores	Get Valid Arrow Scores Calculate total score Calculate Average score	Arrow Scores Total Score Average Score

Data Items	Data Types
Arrows()	Integer
Total	Integer
Average	Single

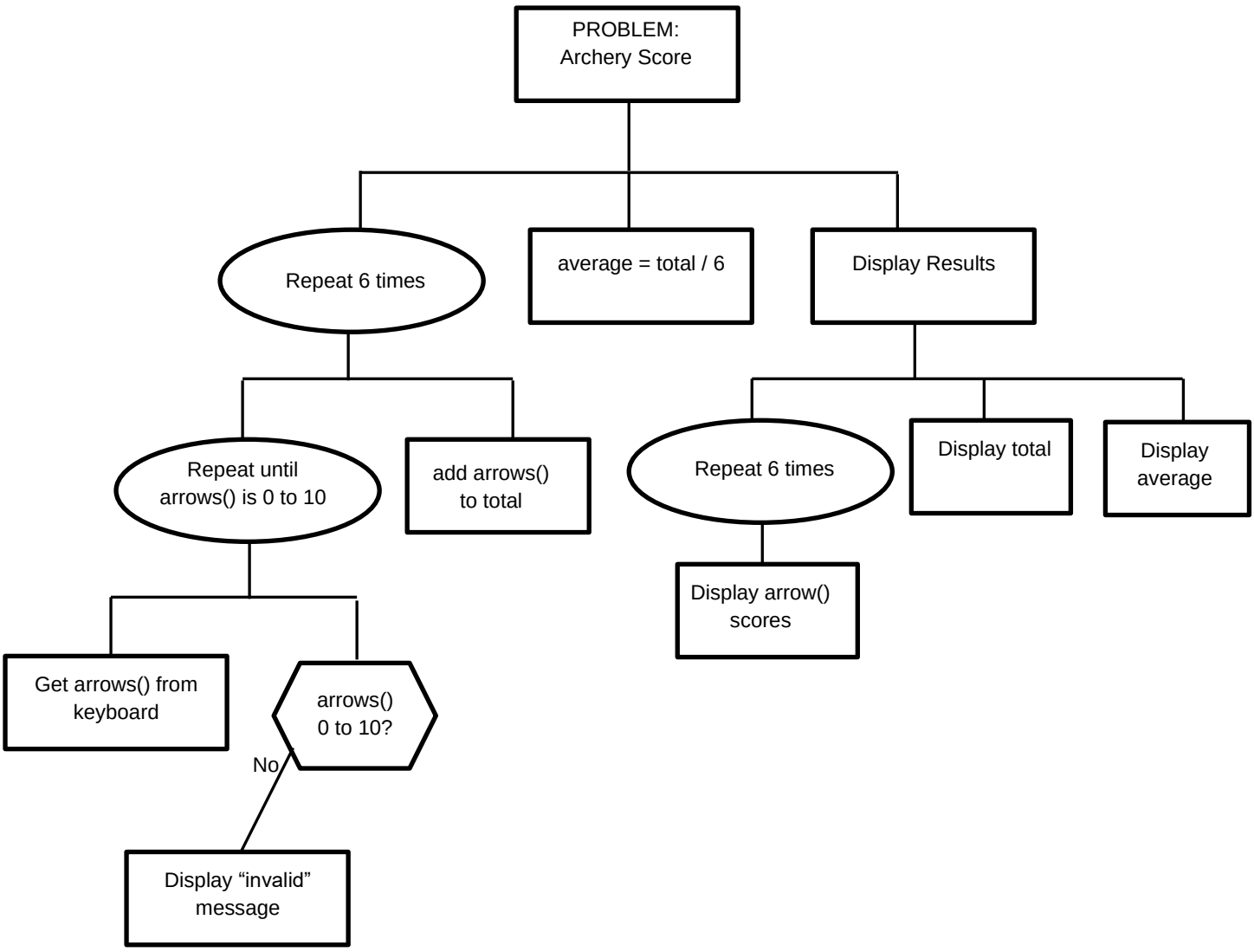
WHOLE NUMBERS (0-10)

Total is a WHOLE NUMBER

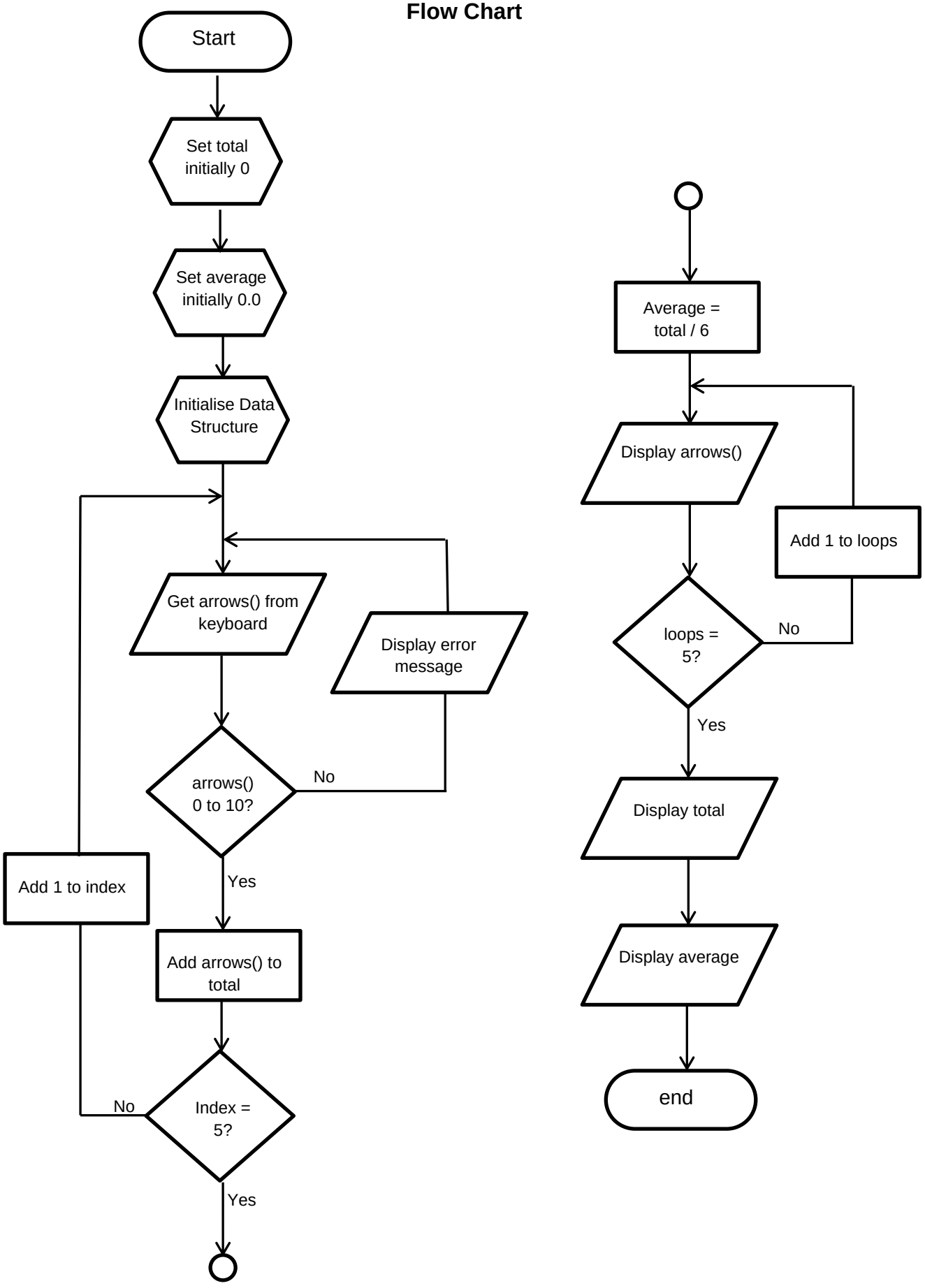
Average will be a REAL NUMBER

Design

Structure Diagram



Flow Chart


Start
Set total initially 0
Set average initially 0.0
Initialise Data Structure
Get arrows() from keyboard
Display error message
arrows() 0 to 10?
No
Yes
Add 1 to index
Add arrows() to total
Index = 5?
No
Yes
Average = total / 6
Display arrows()
Add 1 to loops
loops = 5?
No
Yes
Display total
Display average
end

Pseudocode

Algorithm

1. Initialise variables and data structure
2. Get valid arrow scores and add to total
3. Display arrow scores
4. Display total and average

Refinements

2.1 Start fixed loop for each arrow
2.2 start conditional loop
2.3 Get arrow() from user
2.4 If arrow() is not between 0 and 10 then
2.5 display “Arrow score must be 0 to 10”
2.6 End if
2.7 repeat until arrow score is between 0 and 10
2.8 add arrow() to total
2.9 end fixed loop

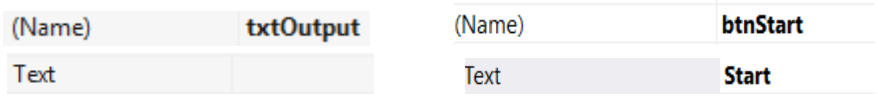
3.1 Start fixed loop for each arrow
3.2 display “arrow:” arrow()
3.3 end fixed loop

4.1 Display “Total score ”,total
4.2 Display “Average arrow score ”, average

Implementation

Create a new Visual Basic project called "*Worked Example 16*"

Add a button called **btnStart** and a textbox called **txtOutput** as shown:



Double click the ***button*** and add the code below:

```
Public Class Form1


Private Sub btnStart_Click(sender As Object, e As EventArgs) Handles btnStart.Click


   Dim arrows(6) As Integer
   Dim total As Integer
   Dim average As Single

   total = 0
   average = 0.0


   For index = 0 To 5


       Do
           arrows(index) = InputBox("Please enter the score for this arrow")
           If arrows(index) < 0 Or arrows(index) > 10 Then
               MsgBox("Arrow score must be 0 to 10")
           End If
       Loop Until arrows(index) >= 0 And arrows(index) <= 10


       total = total + arrows(index)

   Next

   average = total / 6



   For index = 0 To 5

       txtOutput.AppendText("Arrow " & index + 1 & ": " & arrows(index) & vbNewLine)

   Next


   txtOutput.AppendText("Total Score: " & total & vbNewLine)
   txtOutput.AppendText("Average Arrow Score: " & average & vbNewLine)


End Sub

End Class
```

Input Validation

Running Total

Testing

Run the program and complete the test table

Test	Type	Input	Expected Output	Actual Result	Result (pass/fail)
1	Normal	Arrow1=4 Arrow2=5 Arrow3=3 Arrow4=9 Arrow5=2 Arrow6=6	Total = 29 Average = 4.83		
2	Normal	Arrow1=8 Arrow2=7 Arrow3=7 Arrow4=2 Arrow5=3 Arrow6=4	Total = 31 Average = 5.17		
3	Extreme	Arrow1=0 Arrow2=0 Arrow3=0 Arrow4=0 Arrow5=0 Arrow6=0	Total = 0 Average = 0		
4	Extreme	Arrow1=10 Arrow2=10 Arrow3=10 Arrow4=10 Arrow5=10 Arrow6=10	Total = 60 Average = 10		
6	Exceptional	userchoice= 11	Invalid message – re-enter		
7	Exceptional	userchoice= -1	Invalid message – re-enter		

Can you fix the program to make averages display to 2 decimal places?

Cruise Ship

Program Specification

A cruise operator is offering special offers for any passengers who have previously sailed with them. The minimum number of previous trips is 1 and the maximum is 5 and this should be validated.

Discounts will be given a 5% discount per previous trip up to a maximum discount of 25%. A program is required to calculate the discounts to be offered to 10 passengers.

The program should ask for each passenger's name and the number of previous trips they have been on before calculating the discount.

The program should then display each passenger's name, trips and discount.

Charity Lottery

Program Specification

A group of 12 office workers decide to have a weekly charity lottery. Each worker is asked to contribute £1 and a draw is made.

The winner gets half of the contributions with the other half going to charity. If the winner selected has not paid that week, all contributions go to charity.

A program is required to input the names of each worker and whether they have paid or not. A winner is then selected at random and a message indicates the winner's name and whether they should get their winnings or if all money should go to charity.

Track List

Program Specification

A music download website requires a program to calculate the total file size and cost of tracks being downloaded. Tracks can cost from 50p to £2 and file sizes are between 1MB and 5MB. The user should not be able to enter invalid costs of file sizes.

The program should ask for the name, price and file size of 5 music tracks and store this information in arrays. The program should then calculate and display the total cost and download size for the tracks entered.

Football Scores

Program Specification

A football team would like a program to help them to review their performances from the past season. The program should ask for the names of the teams 9 opponents, the number of goals scored against these teams and the number of goals conceded to these teams.

The program should calculate the goal difference against each team (goals scored minus goals conceded) and display this on screen beside the information entered.

Finally, the program should calculate the total goals scored, goals conceded and goal difference and display this on screen.