#  Converting Denary to binary

### Subtraction Method

 Dim currentBITValue, remainingVal, Denary As Integer

 Dim binstring As String

 'input

 Console.WriteLine("enter number")

 Denary = Console.ReadLine()

 remainingVal = Denary

 ' loop through 8 bits starting at 2^7 --> 2^0

 For n = 7 To 0 Step -1

 currentBITValue = 2 ^ n

 If remainingVal >= currentBITValue Then ' is current bit graeter that 2^n

 binstring = binstring & "1"

 remainingVal = remainingVal - currentBITValue

 Else

 binstring = binstring & "0"

 End If

 Next

 Console.WriteLine(binstring)

 Console.ReadLine()

### The Divide by 2 algorithm

The Divide by 2 algorithm assumes that we start with an integer greater than 0. A simple iteration then continually divides the decimal number by 2 and keeps track of the remainder. The first division by 2 gives information as to whether the value is even or odd. An even value will have a remainder of 0. It will have the digit 0 in the ones place. An odd value will have a remainder of 1 and will have the digit 1 in the ones place. We think about building our binary number as a sequence of digits; the first remainder we compute will actually be the last digit in the sequence.



Solution in Vb.net

 Dim remainingVal, Denary As Integer

 Dim binstring As String

 'input

 Console.WriteLine("enter number")

 Denary = Console.ReadLine()

 remainingVal = Denary

 Do

 binstring = Denary Mod 2 & binstring

 Denary = Math.Floor(Denary / 2)

 Loop Until Denary < 1

 Console.WriteLine(binstring)

 Console.ReadLine()