Basic Hardware Concepts

Programming Paradigms

  Procedural
  Object Oriented
  Event Driven

What is VB.Net?

  Windows Applications (Object Oriented, Event Driven)
  Console Applications (Procedural or Object Oriented)

The Object Model

  Class
  Objects (Built-in and user-defined)
  Properties
  Methods

Steps in Writing a Typical VB project.

  Planning:  (GUI, properties, pseudocode or flowchart)
  Coding:   (Convert the GUI to Forms and Controls, Set the properties, Convert the Pseudocode to VB code, Test and Debug.)

The Software Development Life Cycle (SDLC)

  Planning
  Analysis
  Design
  Implementation
  Testing
  Maintenance

Errors (syntax vs. run-time vs. logical errors)

Variables (represents memory, has a type and size)

  Dim strName as string
  integer, double, decimal, boolean, char, byte, string, etc.
  Global vs. Local
  Why initialize variables?

Constants

  Const    TAX_RATE As Decimal = 0.08

Variable and Constant Scope

  1) Module level (variable declared within the scope of class or module)
  2) Local level (variable declared within a procedure)

Compiler Directives

  Option Explicit ON  (Variables cannot be used without being declared first.)
  Option Strict ON   (Makes VB a strongly typed language, No automatic type conversion. Must use the type conversion functions)

Type Conversion functions

  CInt(x), CStr(x), CDec(x), Clng(x), Cdbl()

GUI Components:

  - Forms, Label, TextBox, CheckBox, Button, RadioButton, ListBox, ComboBox, PictureBox, GroupBox

Concatenation and Continuation characters:  (& and _)

Arithmetic operators (+, -, /, \, *, MOD, ^)

Relational operators (=, <, >, >=, <=)

Input and output

  Console Mode:  Console.ReadLine(), Console.WriteLine()
  Windows Applications:  MessageBox.Show(), InputBox()
Branching /Conditional:
Using the IF Statement:
  (if, if-then-else, nested if statements)

Using the (Select Case) statement:
  Select Case Expression
    case X
      Code to run
    case Y
      Code to run
    case else
      Default case
  End select

Loops
(For.. next, Do while...Loop, do Until... Loop)
Necessary conditions for a loop (how to get in, and how to get out)

Truth Table

<table>
<thead>
<tr>
<th>OR</th>
<th>AND</th>
<th>NOT</th>
</tr>
</thead>
<tbody>
<tr>
<td>T OR T = T</td>
<td>T AND T = T</td>
<td>NOT T = F</td>
</tr>
<tr>
<td>T OR F = T</td>
<td>T AND F = F</td>
<td>NOT F = T</td>
</tr>
<tr>
<td>F OR T = T</td>
<td>F AND T = F</td>
<td></td>
</tr>
<tr>
<td>F OR F = F</td>
<td>F AND F = F</td>
<td></td>
</tr>
</tbody>
</table>

Problem Solving Methodology
Top down design
Break the problem into smaller, more manageable tasks.
Divide and conquer
Encourages modular design
Defers the details till later
Functions and Procedures

Procedures and Functions:
Passing arguments
(Pass by value vs. Pass by reference, when?, why?)
Formal vs. actual parameters
Returning values from functions
Via the return statement vs. the function’s parameter list.
Event Procedures (hint: look for “Handles” keyword at the end of procedure declaration)

Arrays
Creating many variables and accessing them via one variable name
Uses of arrays
Operations on the Array:
  Array as an Abstract Data Type (ADT)
  Initialize, load, print, search, sort, etc...

Structures, Classes and Objects
Encapsulation, Inheritance, Polymorphism
Creating new classes
Private vs. Public class variables
Private vs. Public methods
Instantiating objects
Class Constructor
Overloading
Overriding

File Concepts and operations
imports system.io
The stream concept
Reading, Writing