```
==================================================
VBA ERRORS - SYSTEMATIC NOTES
==================================================

OPTION EXPLICIT
--------------------------------------------------
• Always put at top of module
• Forces variable declaration
• Catches errors early

──────────────────────────────────────────────────

1. SYNTAX ERROR
--------------------------------------------------
Definition:
• Mistakes in code structure
• VBA catches before running
• Shown in red immediately

Examples:

WRONG - Missing Then:
Sub Syntax_IfWithoutThen_Wrong()
    Dim x As Integer
    x = 5
    If x > 0      ' <-- missing Then
        MsgBox "Positive"
    End If
End Sub

WRONG - Invalid name:
Sub 123Start()      ' <-- names cannot start with number
    MsgBox "Hi"
End Sub

──────────────────────────────────────────────────

2. COMPILE-TIME ERROR
--------------------------------------------------
Definition:
• Found when VBA checks code before running
• Type mismatch
• Undeclared variables
• Shows error dialog before execution

Example:
Sub CompileError_VarNotDeclared_Correct()
    x = 10        ' Variable not declared
    MsgBox x
End Sub

Error Message: "Compile Error: Variable not defined"

──────────────────────────────────────────────────

3. RUN-TIME ERROR
--------------------------------------------------
Definition:
• Happens during execution
• Invalid operations
• Examples: divide by zero, missing sheet, file not found

Example:
Sub RuntimeErrorExample()
```

```
    Dim a As Integer, b As Integer, c As Double
    a = 10
    b = 0
    c = a / b     ' Run-time error 11: Division by zero
    MsgBox c
End Sub

Error Message: "Run-time error 11: Division by zero"
```

---

4. LOGICAL ERROR

Definition:

- Code runs but gives wrong result
- Hardest to catch
- No VBA warning message
- Only detected by checking output

```
Example:
Sub LogicalErrorExample()
    Dim length As Double, width As Double, area As Double
    length = 10
    width = 5
    area = length + width    ' Wrong: should multiply
    MsgBox "Area is: " & area  ' Output: 15 (wrong, expect 50)
End Sub

Expected Output: 50
Actual Output: 15 (Wrong)
```

SUMMARY TABLE

Error Type	When It Happens	Example Problem	How VBA Shows It
Syntax Error	While typing	Missing quotes	Red highlight
Compile Error	Before run (check)	Type mismatch	Compile error dialog
Run-Time Error	During execution	Divide by zero	Error number dialog
Logical Error	Runs completely	Wrong formula	Wrong output only

---

KEY DIFFERENCES

Syntax Error
→ Caught immediately while typing
→ Red underline in editor
→ Must fix before code can run

Compile Error
→ Found before running code
→ When you try to run, VBA shows error dialog
→ Code structure is wrong

Run-Time Error
→ Code starts running
→ Stops during execution
→ Usually invalid operation (divide by zero, etc.)

Logical Error
→ Code runs completely
→ Produces wrong result
→ No error message from VBA
→ Only you know it's wrong by checking output

HOW TO PREVENT EACH ERROR TYPE

Syntax Errors:
✓ Use correct spelling
✓ Check for matching quotes
✓ Use correct keywords (If...Then, For...Next, etc.)
✓ Variable names start with letter only

Compile Errors:
✓ Use Option Explicit at top of module
✓ Always declare variables with Dim
✓ Use correct data types
✓ Check variable names match declaration

Run-Time Errors:
✓ Check conditions before operations (divide by zero)
✓ Verify objects exist before using (sheet exists)
✓ Check file paths exist
✓ Use error handling (On Error)

Logical Errors:
✓ Review your formulas
✓ Test with known values
✓ Use Debug.Print to check values
✓ Step through code with F8

QUICK REFERENCE - ERROR CODES

Error 11 → Division by zero
Error 13 → Type mismatch
Error 1004 → Application-defined error (often worksheet problem)
Error 91 → Object variable or With block

variable not set

BEST PRACTICES

1. Always use Option Explicit
2. Always declare variables with Dim
3. Use meaningful variable names
4. Add comments explaining code
5. Test code with sample data
6. Use F8 to step through code
7. Check conditions before operations
8. Handle errors with On Error
9. Read error messages carefully
10. Keep code organized and clean

==========
END OF VBA ERRORS - SYSTEMATIC NOTES
==========