**Visual Basic**

1. **Getting Started in a Project**
2. In Visual Studio, choose Visual Basic as the language (on the page for creating your project), then choose -> Windows->Windows Form App. Choose a folder as you did in C++, as well as a project name.
3. Your project will start with a visual form already showing. The source code file will show as soon as you create your first function. You could execute the program now, although it won’t do anything.
4. To begin, **select an object from the Toolbox** (either double click to automatically place an object on the form), or click once, then use the pointer to place the object on the form. If you can’t see the Toolbox, select **View Toolbox**. It should show up on the top or bottom where the Solution Explorer usually resides.
5. To **see properties of an object**, click on it in the form design, and the properties will be in the Properties tab window.
6. **To attach code to an object** (generally a button for our first program), double click on that object and VS will create a function for you. Write the code in there, then execute to see if works correctly.
7. **To create an executable file**, you will need to choose Configuration Manager from the Build Menu. Select “Release” instead of “Debug” from the project. Save that and now it will create a folder and Application file (executable all bundled up) for you, updating every time you execute the code (use the green arrow).
8. **Form of program and more basics:**

a) **Main program**:

Declarations

Functions and Subroutines

Code for main program

**Event driven programming** – Code is attached to objects, and some event (mouse over, mouse click, double click, form create, form close, etc) triggers the code to execute. Our fun is trying to figure out where the code goes. Sometimes, it may need to be in several places. Thus, putting that code in a function that each can call is best. If it needs to be done on start, double click on your main form. In VB, it creates a Form1\_load function in which to enter code. Other languages may call this Form1\_create

b) Each statement is one line unless you use a continuation character to signal another line

**\_** -- Continuation character left at the end of one

\*Note that some multi-line control structures use Keywords to end blocks

**c) ‘** Used to insert comments

3) Add a button to Quit the program. That is the first thing to add to any program. The statement to stop your application is: **Application.exit**

4) **Data types and declarations**

dim variable\_list AS datatype

or dim variable as datatype = initvalue

Data types: Integer, Single, Double, Boolean, String

In many cases, VB will try to convert strings to numbers and vice versa – if it can. Sometimes, that won’t work in a formula and you may need to use intermediate variables or conversion functions to cast the data.

5) **Control Structures**

A) **Mathematical operators and precedence**

() Innermost first

^ Exponentiation (note wrong answer for ^ ^ ) Left to right

\*, / Left to right

\ (backslash) for Integer division – left to right

Mod Left to right

+, - - Left to right

=, <>, <=, >=, <, > Left to right

Not

And AndAlso And and Andalso same - except shortcircuits

Or OrElse Or and OrElse same – except shortcircuits

Xor One or the other but not both

B) **Decision Structures**

If condition then

true stmts

end if

or If condition then

true stmts

else

false stmts

end if

or

if condition then

true1 stmts

elseif condition then

true2 stmts

else

false stmts

endif

OR

Select Case discriminatorvariable (\*can be any type)

Case value OR value1 TO value2 OR value1, value2 OR else

End select

**C) Looping Structures**

for countervariable = initvalue to finalvalue [ step # to step]

loop stmts

next

OR

while *condition*

statements (include a condition update!)

end while

OR

do

loop stmts

loop while *condition*

D) **Functions and subroutines**

i) Functions

function function name ( parameter list) AS datatype

return …

end function

ii) Subroutines

Declared:

Sub subname (byval varname as datatype, etc)

code for sub

end sub