Elementary Programming in Visual Basic
CS 142 - Fall 2009

Instructor: George Thomas **Office:** Halsey 218
Email: thomasg@uwosh.edu **Phone:** 424-2069

Office Hours: MWF 10:30AM -11:30AM,
TR 10:00AM -11:30AM, or **by appointment**.

Section 1:	Lectures:	MF	1:50PM-2:50PM,	HS 212
	Labs:	W	1:50PM-2:50PM,	HS101C
Section 2:	Lectures:	T	8:00AM-9:30AM,	HS 208
	Labs:	W	8:00AM-9:30AM,	HS101C

Prerequisites: A grade of C or better in Math 104 or Math 108 or Math 206 or CS 142, or qualifying for Math 171 via the Mathematics Placement Exam.

Required Textbook: *An Introduction to Programming Using Visual Basic 2008.* Seventh Edition, David I. Schneider, Prentice Hall, 2009.

Course Website: UWO D2L

Note: If you have special needs, please come and talk to me at the end of the first class.

Course Overview
This course is aimed at learning the fundamentals of programming in a business oriented language. Students concentrate on the development of programming logic to solve business problems. Design tools such as flow charts and pseudocode are introduced and used as a basis for developing programs. The basics of graphical user interfaces and the Visual Basic programming environment are included.

Course Objectives: The main objectives are, for a student, upon completion of this course, to be able to:

- Apply the program development cycle to solve computational problems.
- Represent algorithms using flowcharts, pseudocode, and hierarchy charts.
- Design and implement a graphical user interface using VB.NET forms and controls.
- Write algorithms that use loops and make decisions.
- Manipulate arrays in VB.NET programs.
- Design modular code with Sub Procedures and Functions.
- To write event handlers that react to user input.
- To create, open, read from, and write to sequential files.
- To handle exceptional cases using “Try…Catch” blocks.

Topic Coverage: We will cover the following topics:

- Introduction to computers and the Windows interface.
- Problem solving
- Fundamentals of Visual Basic .NET
- Procedures: Subprograms and Functions
- Decisions: "If" and Select Case" blocks
- Repetition: "Do" and "For…Next" loops
- Arrays
- Sequential files
- Additional controls

Course Grading Policy: Your final grade for this course will be based on four components, namely exams, programming assignments, labs, and unannounced quizzes. Within each component (for example, the *labs* component), all items (for example, all labs) will have equal weight. In other words, each component grade will be computed as the average of all items in the component (however, see important note below). Your overall numerical grade for the course will be computed as the weighted sum of the component grades using the following weights:

Component	Weight
Exams (3)	60%
Assignments (4 to 6)	20%
Weekly labs	10%
Quizzes	10%

Important note: If you receive a zero on more than 20% of your lab grades, your **overall lab grade** will automatically be reduced to 0. The same rule applies to your **overall quiz grade**.

Finally, your letter grade for the course will be computed as follows:

Numerical Score	Grade	Numerical Score	Grade
>=92	A	72-78	C
90-92	A-	70-72	C-
88-90	B+	68-70	D+
82-88	B	62-68	D
80-82	B-	60-62	D-
78-80	C+	<60	F

While this overall grading scheme is fixed, I will be happy to discuss any issue you may have with individual grades. If you notice a mistake or have a question regarding a specific grade, please come and talk to me *as soon as possible*. Do not wait until the end of the semester to bring up grading issues.

Attendance and Participation: You are expected to not only attend **every** class meeting but also to come **prepared** for and **participate** actively in it. Necessary preparation requires you to have studied and assimilated the material covered in previous sessions, to have met with me or the tutors outside of class to discuss any questions you may have, to have done the assigned

reading (when applicable), and to have completed the programming assignments on time. I have high expectations for my students and will always support and encourage you. I **strongly encourage you to ask any question** or raise any issue you have with the course either during or at the end of class, or during my office hours. I will also gladly meet with you by appointment.

Late Submissions: The submission procedure for your assignments and labs will be described later. However, please note that each assignment and lab will come with a deadline (day and time) after which any submission is considered late. The late-submission policy works as follows:

Turned in	Penalty
on due date but after deadline	10%
One day late	25%
Two days late	50%
Three days late	75%
Four days late	100%

Weekend days and holidays count as "regular days" when computing late penalties. Extensions on assignments may be granted at the discretion of the instructor if you provide a valid justification (in the form of a written excuse from a medical doctor or the Dean of Students Office) **before** the due date.

If you miss a scheduled exam, you **may** be able to take a make-up exam provided you give the instructor a valid justification (see above) ahead of time if possible. Only one make-up exam will be given. It will be a comprehensive exam scheduled at the end of the semester. It will take place at the testing center. Similarly, there will be no make-up quizzes unless the instructor is provided with a valid justification (see above) for your absence on the day of the quiz.

Collaborating versus Cheating: Unless otherwise stated in the assignment or lab, all submissions must be the work of a single student (the one whose name appears on the submission, that is). While it is acceptable to discuss the assignments at a high level (for example, at the design level) with others, you must submit your own work. You may not "borrow" any piece of code or design of any length from someone else, unless you can live with a zero and the other potential academic sanctions of cheating (see UWO Student Discipline Code 2007, Chapter UWS 14).

In conclusion, be aware that Computer science classes require significant work. It takes considerable practice to develop the programming and analytical skills outlined in this syllabus. I expect that you will need to spend at least two hours of effort outside of class for each in-class hour. Having said this, I expect every hardworking student to do well in this course.