COURSE NUMBER: Ve280	**COURSE TITLE:** Programming and Introductory Data Structures
CREDIT: 1	**PREREQUISITES:** Vv156 or Vv186,and Vg101 or equivalent
TEXTBOOKS/REQUIRED MATERIAL: "Problem Solving with C++, 8th Edition," W. Savitch	**INSTRUCTOR:** Weikang Qian **DATE OF PREPARATION:** May. 25, 2012 **DATE OF UC APPROVAL:** Oct. 30, 2013
INSTRUCTOR(S): Weikang Qian	**SCIENCE/DESIGN:** n/a
CATALOG DESCRIPTION: Techniques and algorithm development and effective programming, top-down analysis, structured programming, testing, and program correctness. Program language syntax and static and runtime semantics. Scope, procedure instantiation, recursion, abstract data types, and parameter passing methods. Structured data types, pointers, linked data structures, stacks, queues, arrays, records, and trees.	**COURSE TOPICS: (assume 27 lectures in total)** 1. Linux basics and compiling program on Linux (2 lectures) 2. Review of C++ Basics, such as array, pointer, etc. (1 lecture) 3. Procedural abstraction, function call mechanism, and recursion (2.5 lectures) 4. Function pointer (0.5 lecture) 5. Enum (0.5 lecture) 6. Program arguments (0.5 lecture) 7. Testing (1 lecture) 8. Debugging (including how to use GDB) (0.5 lecture) 9. IO (1.5 lectures) 10. Exception (1 lecture) 11. Abstract data type and class (3 lectures) 12. Inheritance and virtual function (2 lectures) 13. Interface (i.e., abstract base class) (0.5 lecture) 14. Representation invariant (0.5 lecture) 15. Dynamic memory allocation and dynamic arrays (1 lecture) 16. Overloaded constructor, destructor, copy constructor, and overloaded assignment operator (2 lectures) 17. Operator overloading and friend mechanism (1 lecture) 18. Linked list (including linked list traversal) (2 lectures) 19. Stack and queue (2 lectures) 20. Polymorphism, template, and STL (2 lectures)
COURSE STRUCTURE/SCHEDULE: Lecture: 2.5 times per week, 90 minutes each.	

COURSE OBJECTIVES [Course Outcomes in brackets]	1. To give an introduction to programming and to provide a foundation on data structures. [1, 6, 7, 8] 2. To provide students with experience on how to design and implement an algorithm to solve a practical problem. [1, 2, 3, 8, 9] 3. To teach students some useful techniques for developing, debugging, and testing programs. [4, 5, 9]
COURSE OUTCOMES [Program Outcomes in brackets]	After completing Ve280, students should be able to: 1. Take a problem and consider various possible approaches for solving it. [a, c, e] 2. Select an approach—or algorithm—that provides for a simple, clean, well-structured solution. [b, e] 3. Convert the algorithm into C++ code, using good design and style. [c, k] 4. Develop and debug a program on Linux operating systems. [k] 5. Test and debug the program using rigorous techniques. [b, k] 6. Understand the concepts of top-down design, data encapsulation, information hiding, and procedural and data abstraction. [a] 7. Design, implement, and use classes, including constructors, destructors, and operator overloading. [a] 8. Implement dynamic data structures for stacks, queues, and lists. [a, c] 9. Be able to quickly design, implement, test, and debug a large scale project independently (1000+ lines of code). [a, b, c]
ASSESSMENT TOOLS [Course Outcomes in brackets]	Programming Projects [1, 2, 3, 4, 5, 6, 7, 8, 9] Midterm and Final Exam [1, 2, 3, 5, 6, 7, 8]