

Course #:	INFO-I 211
Course Title:	Informatics Infrastructure II
Course Type:	Required core
Prerequisites:	INFO-I 210
Credits:	4
Text Book:	Absolute C++, Second Edition, Walter Savitch, Addison Wesley
References:	Handouts
Current Catalog Description:	P: INFO I210. The systems architecture of distributed applications. Advanced programming, including and introduction to the programming of graphical systems. Cross listed with CSCI C201.
Course Goals	<p>The student who completes this course:</p> <ol style="list-style-type: none"> 1. Will be proficient in advanced programming techniques such as those covered in advanced C++ . 2. Will be proficient in constructing program that use elementary data structures, use of pointers, use of dynamically allocated memory such as linked list 3. Will be proficient in the use of data files 4. Will be proficient in the construction of classes and other object oriented facilities such as inheritance, overloading, overriding.
Major Topics Covered in the Course	<ol style="list-style-type: none"> 1. Structs 2. Bitwise operators {and the internal representation of data} 3. More on pointers 4. Multidimensional arrays 5. Dynamically allocated arrays: new and delete operators 6. Register storage class, How memory is allocated for a C/C++ program. 7. Linked lists 8. C style I/O, scanf and print 9. More standard library functions, qsort, bsearch, clock, memmove, etc. 10. Inline functions 11. Function overloading 12. Function templates 13. More on parameter passing: const parameters, function parameters. 14. Command line parameters. 15. Typedef 16. Conditional compilation; 17. Separate compilation. 18. Files 19. Classes <ul style="list-style-type: none"> - member functions, - templates - implementing abstract data types

	<ul style="list-style-type: none"> - dynamic classes - constructors and destructors - inheritance - virtual functions 																		
Laboratory projects (specify number of weeks on each)	No closed laboratory																		
Estimate Curriculum Category Content (Semester hours)	<table border="1" style="margin-left: auto; margin-right: auto;"> <thead> <tr> <th>Area</th> <th>Core</th> <th>Advanced</th> </tr> </thead> <tbody> <tr> <td>Algorithms</td> <td>8</td> <td></td> </tr> <tr> <td>Software Design</td> <td>25</td> <td></td> </tr> <tr> <td>Comp. Arch.</td> <td>2</td> <td></td> </tr> <tr> <td>Data Structures</td> <td>5</td> <td></td> </tr> <tr> <td>Prog. Languages</td> <td>30</td> <td></td> </tr> </tbody> </table> <p>Additional hours may be dedicated to curriculum categories not listed above. For example explanation of concepts and theories. Discussion of social and ethical issues, discussion of inter personal relationships and working within groups.</p>	Area	Core	Advanced	Algorithms	8		Software Design	25		Comp. Arch.	2		Data Structures	5		Prog. Languages	30	
Area	Core	Advanced																	
Algorithms	8																		
Software Design	25																		
Comp. Arch.	2																		
Data Structures	5																		
Prog. Languages	30																		
Oral and Written Communications	Not a course objective.																		
Social and Ethical Issues	Issues of software license, the difference between commercial, public, and open source software, the GNU project and associated license, 30 minutes.																		
Theoretical Content	Not a course objective.																		
Problem Analysis	The student is taught how to analyze a problem and the correct steps to solving a problem and writing an algorithm.																		
Solution Design	<p>The design steps would be the following:</p> <ol style="list-style-type: none"> 1) Understand what the question or problem is asking. 2) Identify what is needed by the problem; all inputs, outputs and equations. 3) Determine the constraints or limits on the problem. 4) Write an algorithm to solve problem. 5) Translate algorithm into a program. 6) Run and test program 																		
Prepared By	Hakimzadeh																		