

	Course Revision - Computer Science Curriculum
	Part IJ. Description of Curriculum Change
	1. New Syllabus of Record
	I. Course Description
	COSC 10 Object Oriented and GHI Programming 20 Al 2cr
	Prerequisite: COSC 108 or COSC 110
	An jn-denth introduction to the Object Oriented Programming (OOP) naradigm including encapsulation
<u> </u>	
<u> </u>	
	inheritance, and polymorphism. The focus will be on designing, implementing, and using objects. This course will also include an introduction to Graphical User Interface (GUI) design and programming.
	II. Course Outcomes
	Upon successful completion of the course, the student will be able to:
	1. Demonstrate an understanding of the fundamental concepts of the OOP paradigm including
	encapsulation, inheritance, polymorphism, interfaces, and overloading/overriding. 2. Demonstrate an understanding of the fundamental concepts of GUI programming.
	District james garante amenil amenicana (a a involve avorand a dazan saures files) using an object
·	
•	
<u>. </u>	

oriented programming language. A few applications will use a GUI interface.

Course Revision - Computer Science Curriculum 3. Control Statements 3 hours **Algorithms** • if, then, while, switch, break, continue, and for • Expressions and assignments re-visited; operators, precedence. • Data type handling, object vs. primitive, identity vs. equality **Exception handling** Programming conventions part 3 4. More on Obiect Definitions 3 hours Static fields and methods Overloading • Packages (or namespaces) and encapsulation revisited • Intro to API libraries/packages Parameter passing Argument promotion and casting 3 hours 5. Arrays Declaring and creating arrays Initializing an array Arrays of Objects For each statement Multidimensional arrays

Course Revision - Computer Science Curriculum Polymorphism behavior using arrays. Final attribute on methods and classes Abotrant alanna un annota ala----10. Interfaces 3 hours Definition of an interface, a contract Implementation vs. use Programming to an interface Polymorphism behavior using arrays. Final attribute on methods and classes UML class diagrams for interfaces

- Revisit IO
- Collections APIs
- List Iterator and Man interfaces

IV. **Evaluation Method**

Grade distribution:

•	Quizzes	10%
•	Mid Term (1 hour 15 min)	15%
_	P' 1 P (O)	

Around 8 Programming Projects

50%

Participation

5 %

Grade Scale:

90 - 100%

80 - 89%

70 - 79%

60 - 69%

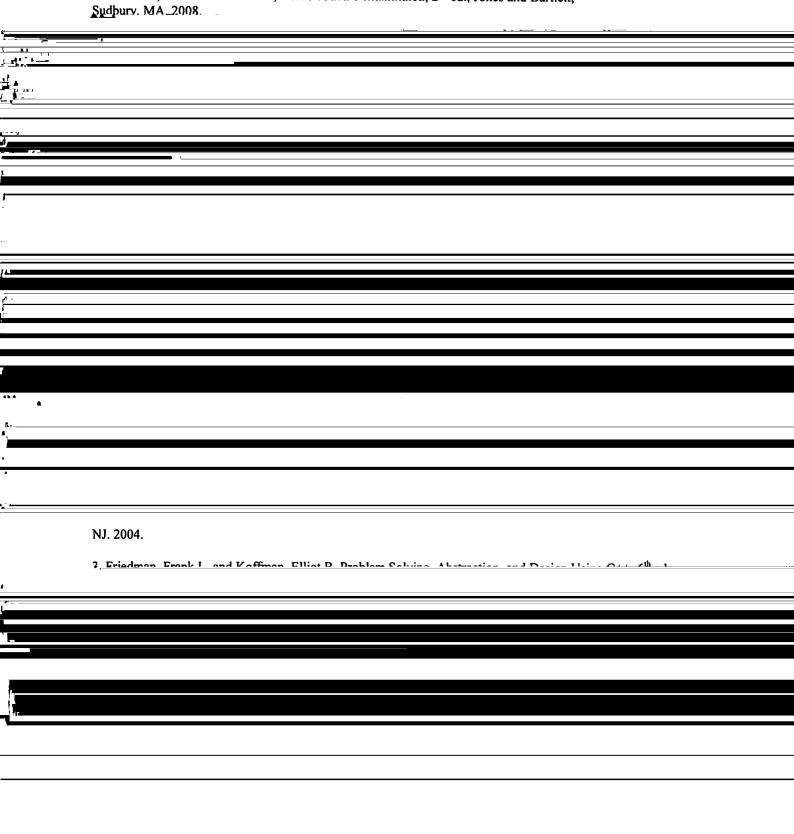
D

< 60%

Attendance Policy:

Attendance is crucial to success in this course. To encourage class attendance, the following policy will be used: Attendance will be taken at every class. For each unexcused absence, starting with the fourth, 2% will hadedinted from the overall class grade Generally excused absences involve illness with a doctor's

1. Anderson, Julie and Franceschi, Herve'. Java 6 Illuminated, 2nd ed., Jones and Bartlett,



2. Summary of Proposed Revisions

Prerequisites have been changed to COSC 108 or COSC 110.

The course outcomes are essentially the same, but have been enhanced to provide clarification and use measures targeting higher levels in Bloom's taxonomy. Two outcomes were added in relational to adapt to current trends in the industry. The additional outcomes relate to industry programming practices (outcome

4. The Old Syllabus of Record

I. Course Description

COSC 210 Object Oriented and GUI Programming

Prerequisite: COSC110

An in-depth introduction to the Object Oriented Programming (OOP) paradigm. The focus will be on

inheritance, and polymorphism. This course will also include an introduction to Graphical User Interface

II. Course Objectives

(GUI) design and programming.

The student will:

- 1. Learn the fundamental concepts of the OOP paradigm.
- 2. Implement object definitions.
- 2. Implement object definitions.

3. Operator overloading	
4. Friend functions	
E. Using Objects	4 hours
} Sconino	
2. Static and dynamic objects	
3. Arrays of objects	
F. Object Design	3 hours
1. Goals of Object Oriented Software	
G. Templates	4 hours
2. Simple sorting techniques (exchange, insertion, selection)	
2. Simple sorting techniques (exchange, insertion, selection) 3. Translate object definitions	
3 Tannote object definitions	
3 Tannota object definitions H. Inheritance	4 hours
3 Tannote object definitions H. Inheritance 1. Derived types	4 hours
H. Inheritance 1. Derived types 2. Virtual functions	4 hours
H. Inheritance 1. Derived types 2. Virtual functions 3. Declarations	4 hours
H. Inheritance 1. Derived types 2. Virtual functions 3. Declarations 4. Single and multiple inheritance	4 hours
H. Inheritance 1. Derived types 2. Virtual functions 3. Declarations 4. Single and multiple inheritance 5. Public vs. private inheritance	4 hours
H. Inheritance 1. Derived types 2. Virtual functions 3. Declarations 4. Single and multiple inheritance 5. Public vs. private inheritance 6. Virtual Derivations	
H. Inheritance 1. Derived types 2. Virtual functions 3. Declarations 4. Single and multiple inheritance 5. Public vs. private inheritance 6. Virtual Derivations I. Run-Time Type Identification	4 hours 2 hours
H. Inheritance 1. Derived types 2. Virtual functions 3. Declarations 4. Single and multiple inheritance 5. Public vs. private inheritance 6. Virtual Derivations I. Run-Time Type Identification 1. Polymorphism	2 hours
H. Inheritance 1. Derived types 2. Virtual functions 3. Declarations 4. Single and multiple inheritance 5. Public vs. private inheritance 6. Virtual Derivations I. Run-Time Type Identification 1. Polymorphism J. Exception Handling and Debugging	
H. Inheritance 1. Derived types 2. Virtual functions 3. Declarations 4. Single and multiple inheritance 5. Public vs. private inheritance 6. Virtual Derivations I. Run-Time Type Identification 1. Polymorphism J. Exception Handling and Debugging 1. Exception handling functions	2 hours
H. Inheritance 1. Derived types 2. Virtual functions 3. Declarations 4. Single and multiple inheritance 5. Public vs. private inheritance 6. Virtual Derivations I. Run-Time Type Identification 1. Polymorphism J. Exception Handling and Debugging 1. Exception handling functions 2. Expected and unexpected exceptions	2 hours
H. Inheritance 1. Derived types 2. Virtual functions 3. Declarations 4. Single and multiple inheritance 5. Public vs. private inheritance 6. Virtual Derivations I. Run-Time Type Identification 1. Polymorphism J. Exception Handling and Debugging 1. Exception handling functions	2 hours
H. Inheritance 1. Derived types 2. Virtual functions 3. Declarations 4. Single and multiple inheritance 5. Public vs. private inheritance 6. Virtual Derivations I. Run-Time Type Identification 1. Polymorphism J. Exception Handling and Debugging 1. Exception handling functions 2. Expected and unexpected exceptions	2 hours
H. Inheritance 1. Derived types 2. Virtual functions 3. Declarations 4. Single and multiple inheritance 5. Public vs. private inheritance 6. Virtual Derivations I. Run-Time Type Identification 1. Polymorphism J. Exception Handling and Debugging 1. Exception handling functions 2. Expected and unexpected exceptions	2 hours
H. Inheritance 1. Derived types 2. Virtual functions 3. Declarations 4. Single and multiple inheritance 5. Public vs. private inheritance 6. Virtual Derivations I. Run-Time Type Identification 1. Polymorphism J. Exception Handling and Debugging 1. Exception handling functions 2. Expected and unexpected exceptions	2 hours

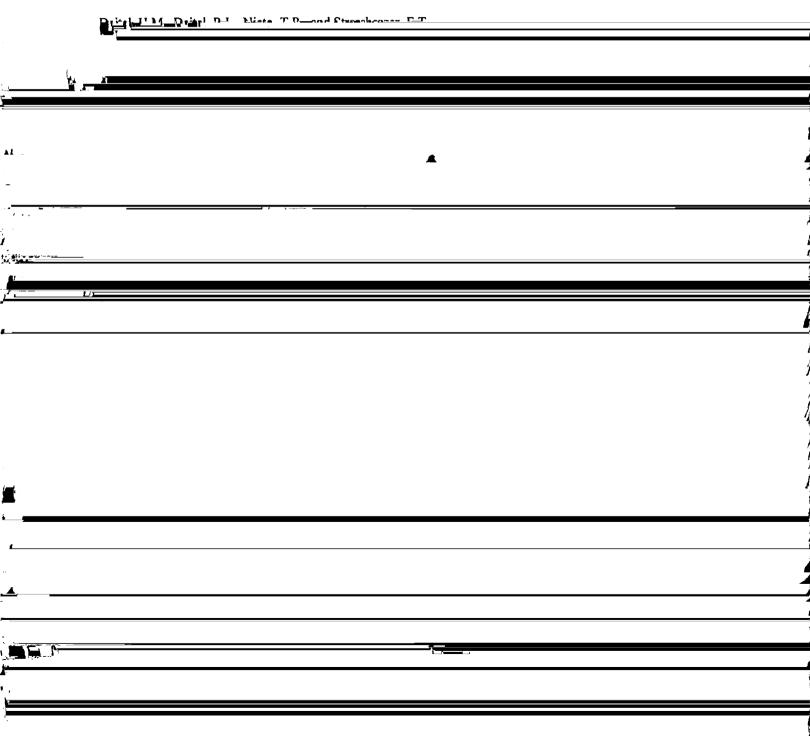
- Goals of GUI software
 L. Implementing GUIs using Object Oriented Programming
 Concepts and terminology

Project 4. Develop a hierarchical set of classes such as quadrilaterals. This project will include inheritance, polymorphism, protected attributes and methods and overloading.

Project 5. Develop a Dialog Based Graphical User Interface application. This project will introduce the concepts of GUI design and development. The project will also include the use of a large library of predefined classes.

V. Textbook(s)

Deitel, H.M. and Deitel, P.J. C++ How to Program. Second Edition. Prentice Hall PTR, Upper Saddle River, NJ. 1998.



	ell-III Berliging Completor Science Comiculus
· ·	
P	art II. Letters of Support
- '	
	Not Applicable. This is an internal change to the Computer Science program. Affected programs are all
	<u> </u>
·	

tracks in the Computer Science major, the Computer Science minor, and the Information Assurance Minor