SAULT COLLEGE OF APPLIED ARTS & TECHNOLOGY

SAULT STE. MARIE, ONTARIO

COURSE OUTLINE

Course Title: INTRODUCTION TO OBJECT-ORIENTED PROGRAMMING

Code No.: CSD205 *Semester:* 4

Program: COMPUTER ENGINEERING TECHNICIAN
COMPUTER PROGRAMMER

Author: Tycho Black

Date: December , 1997 *Previous Outline Dated:* December, 1996

Approved: _____ 97-01-05
Dean Date

Total Credits: 5

Prerequisites: CSD101

Length of Course: 4 Hours/Week Total Credit Hours: 75

I. COURSE DESCRIPTION:

This course introduces students to the concepts of Object-Oriented Programming and applies them in practical problem solving exercises.

II. LEARNING OUTCOMES AND ELEMENTS OF THE PERFORMANCE:

A. Learning Outcomes:

1. Compare Object-oriented programming (OOP) with procedural programming and compare various OOP languages.
2. Identify the features of C++ which make it a "better C" , and apply them to programming problems.
3. Demonstrate an understanding of classes, encapsulation and polymorphism by solving programming problems involving their use.
4. Write programs which use inheritance and exception handling to demonstrate object-oriented design.

B. Learning Outcomes and Elements of the Performance:

Upon successful completion of this course the student will demonstrate the ability to:

1. **Compare Object-oriented programming (OOP) with procedural programming and compare various OOP languages.**

Elements of the Performance:

- Identify the most important features of Object-oriented programming languages.
- Assess the strengths and weaknesses of OOP and procedural programming.
- Compare various development environments for use in OOP.
- Compare C++ with Java.

This learning outcome will constitute approximately 5% of the course.

2. **Identify the features of C++ which make it a "better C", and apply them to programming problems.**

Elements of the Performance:

- Utilize basic stream i/o.
- Organise program modules into projects and utilise the tools available in an Integrated Development Environment for program development.
- Utilise structs and unions for organising dissimilar data elements into new data types.
- Utilise enumerator (enum) variables.
- Manage memory allocation and deallocation with *new* and *delete*.
- Use pointers effectively.
- Demonstrate a knowledge of scope and storage classes.
- Utilise available methods for passing arguments to functions, including the use of default arguments.
- Use operator and function overloading (polymorphism) to provide multiple behaviours for operators and functions.

This learning outcome will constitute approximately 20% of the course.
Reference: Prata, Chap. 1, 2, 4, 7, 8

3. **Demonstrate an understanding of classes, encapsulation and polymorphism by solving programming problems involving their use.**

Elements of the Performance:

- Define classes and implement class member functions using appropriate encapsulation (data hiding) mechanisms.
- Declare and define constructors and destructors for classes.
- Use the *this* pointer to point to the invoking object.
- Implement a stack Abstract Data Type (ADT).
- Implement friend functions and friend classes appropriately.
- Use dynamic memory allocation and other improvements for an enhanced String class.
- Implement a *queue* ADT using OOP methods.

This learning outcome will constitute approximately 50% of the course.
References: Prata, Chap. 9, 10, 11

4. Write programs which use inheritance and exception handling to demonstrate object-oriented design.

Elements of the Performance:

- Use class inheritance to implement *Is-a* relationships.
- Use virtual functions (late binding) to redefine class methods for derived classes.
- Implement *Has-a* relationships with member objects (containment) or with private inheritance.
- Discuss the appropriate use of class templates.
- Use C++ exception handling in programs.

This learning outcome will constitute approximately 25% of the course.
Reference: Prata, Chap. 12, 13, 14

III. TOPICS TO BE COVERED:

1. New features of C++, and elements of C not previously covered.
2. Overview of the elements and objectives of OOP.
3. Classes, encapsulation, constructors, destructors, parameter passing.
4. Polymorphism (function overloading) and operator overloading.
5. Inheritance, friends and exception handling.

IV. REQUIRED STUDENT RESOURCES/TEXTS:

TEXT BOOK:

"C++ Primer Plus" (2nd Ed) by Stephen Prata.
Waite Group Press, 1995

V. EVALUATION PROCESS/GRADING SYSTEM:

3 WRITTEN TESTS	@ 20% each	60%
ASSIGNMENTS /QUIZZES		40%

(The percentages shown above may vary slightly if circumstances warrant.)

NOTE: *It is required to pass both the theory and the assignment part of this course. It is not possible to pass the course if a student has a failing average in the three written tests but is passing the assignment portion, (or vice versa).*

TESTS and QUIZZES

Tests will be announced about one week in advance. A zero grade will be given for tests or quizzes missed without a valid reason. Generally the only valid reasons are medical ones. Quizzes may be unannounced but warning will generally be given.

In some cases assignments will be evaluated with a quiz based on the assigned work, on the due date. Re-writes on these quizzes will not generally be possible so it is essential that assigned work is completed on time and the quizzes written at the required time.

GRADING SYSTEM

A+		90	-	100%
A		80	-	89%
B		70	-	79%
C		55	-	69%
R	Repeat	Less than 55%		
X	Incomplete			

PLAGIARISM

While it is expected that students discuss assignments with each other and share ideas, it is not acceptable that students hand in work done by someone else and claim it as their own. **Plagiarism on assignments will result in a zero grade being assigned for that assignment for everyone involved.**

UPGRADING OF INCOMPLETES

When a student's course work is incomplete or final grade is below 55%, there is the possibility of upgrading to a pass when a student meets all of the following criteria:

1. The students attendance has been satisfactory.
2. An overall average of at least 45% has been achieved.
3. The student has not failed all of the theory tests taken.
4. The student has made reasonable efforts to participate in class and complete assignments.

The nature of the upgrading requirements will be determined by the instructor and may involve one or more of the following: completion of existing labs and assignments, completion of additional assignments, re-testing on individual parts of the course or a comprehensive test on the entire course.

ATTENDANCE:
Absenteeism will affect a student's ability to succeed in this course. Absences due to medical or other unavoidable circumstances should be discussed with the instructor. Those whose attendance is poor will not be eligible for an X-grade.

VI. SPECIAL NOTES:

- **Special Needs**
  Students with special needs (e.g. physical limitations, visual or hearing impairments, or learning disabilities) are encouraged to discuss any required accommodations confidentially with the instructor and/or contact the Special Needs Office so that support services can be arranged.

- **Retention of Course Outlines**
  It is the responsibility of the student to retain all course outlines for possible future use in acquiring advanced standing at other post-secondary institutions.

- **Course Modifications**
  Your instructor reserves the right to make reasonable modifications to the course as deemed necessary to meet the needs of students or take advantage of new or different learning opportunities.

VII. PRIOR LEARNING ASSESSMENT:

Students who wish to apply for advanced standing in the course should consult the instructor. This course is not eligible for challenge at the present time.