Effective Semester / Session: Fall 2006

Type of Action:  
- New  
- Modification  
- Cancellation  

Course Alpha and Number: CS 227

Course Title: Introduction to Programming

Reason for initiating, revising, or canceling:  
This course will serve as the introductory programming course in the computer technology curriculum. It will consolidate the existing introductory CS programming courses CS223, CS224, CS225, and CS229.
1. **Department:** Business

2. **Purpose**
   This introductory computer programming course is the equivalent to the PRG1 (Programming 1) course recommended by the ACM (Association of Computing Machinery) as part of a Computer Information Systems associate degree program. Students will be taught the fundamentals of programming that are applicable to any programming language. All fundamental programming concepts will be taught in this course and the selection of an appropriate computer language will be at the discretion of the instructor.

   The primary target population is students interested in computers that currently consist of the students in the A.A.S. in Business Administration degree with the Computer Applications emphasis.

3. **Description**

   **A. Required/Recommended Textbook(s) and Related Materials**

   Readability level: Grade 11

   **B. Contact Hours**
   Lecture: 3 hours per week / 45 hours per semester

   **C. Credits**
   1. **Number:** 3
   2. **Type:** Regular Degree Credits

   **D. Catalogue Course Description**
   This course introduces computer programming with emphasis on program design, coding, debugging, testing, documentation, and algorithm design. The course presents such fundamentals of programming as data types, operators, control structures, arrays, strings, and functions. Prerequisite: CS222, MA132, or concurrent enrollment. English and Math Placement Levels: EN 093/094, MA 151.
E. Degree or Certificate Requirements Met by Course
This course is a required course for the A.A.S. degree in Business Administration with an emphasis in Computer Applications, and several associated Certificates of Achievement in the computer field. It also serves as a computer elective course for other degree options.

F. Course Activities and Design
Course activities include lectures and demonstrations using a computer and screen projector. Assignments involve creating simple programs to demonstrate proficiency. Individual coaching is used to discover and eliminate syntax errors and other program bugs. Exams assess students' understanding of the materials learned. A final project assesses the students' ability to apply the programming concepts.

4. Course Prerequisite(s); Concurrent Course Enrollment; Required English/Mathematics Placement Level(s)
Prerequisites: CS222, MA132, or concurrent enrollment
English Placement Level: EN 093/094

5. Estimated Cost of Course; Instructional Resources Needed
Cost to the Student: Tuition for a three-credit course; cost of the textbook.
Cost to the College: Instructor's salary, computer hardware and software, computer supplies, computer maintenance, lab aides' stipends and reproduction of teaching supplies. Instructional resources needed for this course include instructor's computer console, computer projector and projection screen, sound card and speakers, whiteboard, whiteboard markers, photocopied handouts, and appropriate reference materials.

6. Method of Evaluation
Student grades will be based on the regular letter grade system as described below:

A: Excellent – grade points: 4.0;
B: Above average – grade points: 3.0;
C: Average – grade points: 2.0;
D: Below average – grade points: 1.0;
F: Failure – grade points: 0.0.

NMC's grading and attendance policies will be followed.
7. **Course Outline**
This is a topical outline and does not necessarily indicate the sequence in which the material will be presented.

1.0 Background and history of programming language
2.0 Information coding: integer numbers, floating point numbers, character strings
3.0 Editors, compilers, interpreters, and debuggers
4.0 Variable declaration, initialization, and data types
5.0 Operators: assignment, arithmetic, relational, logical, and precedence
6.0 Selection control structures: if else, else if, switch case
7.0 Repetition control structures: while, do while, for
8.0 Functions
9.0 Arrays
10.0 Objects
11.0 Document object model
12.0 Event driven programming
13.0 Problem solving and algorithm design
14.0 Testing and debugging
15.0 Proper program documentation
16.0 Program development life-cycle

8. **Instructional Goals**
This course will introduce students to:

1.0 The differences between integer, floating point, and character string data;
2.0 Problem analysis and designing a solution algorithm that will correctly solve the problem;
3.0 Testing and debugging the algorithm using known test data;
4.0 Implementing the algorithm by coding it into a programming language;
5.0 Debugging programs by removing syntax and logic errors;
6.0 The parts of the program development life cycle;
7.0 Declaring and initializing variables using self-documenting identifiers;
8.0 Selection control structures using if else and else if;
9.0 Repetition control structures using while, do while, for;
10.0 Code, and debug functions;
11.0 Function parameters, arguments, definitions, function calls, and return values;
12.0 Arrays in a program;
13.0 Creating user defined functions and calling these functions in a program;
14.0 Accessing a component object model to create an event-driven program;
15.0 Properly documenting programs using comments.

9. Student Learning Outcomes
Upon successful completion of this course, students will be able to:

1.0 Understand the difference between integer, floating point, and character string data;
2.0 Analyze a problem and design a solution algorithm that will correctly solve the problem;
3.0 Test and debug the algorithm using known test data;
4.0 Implement the algorithm by coding it into a programming language;
5.0 Debug the program by removing syntax and logic errors;
6.0 Identify the parts of the program development life cycle;
7.0 Declare and initialize variables using self-documenting identifiers;
8.0 Implement selection control structures using if else and else if;
9.0 Implement repetition control structures using while, do while, for;
10.0 Understand, code, and debug functions;
11.0 Identify function parameters, arguments, definitions, function calls, and return values;
12.0 Implement arrays in a program;
13.0 Create user defined function and call these functions in a program;
14.0 Access a component object model to create an event-driven program;
15.0 Properly document programs using comments.

10. Assessment Measures
Assessment of student learning may include, but not be limited to, the following:

1.0 Class attendance and participation
2.0 Practical project assignments
3.0 Periodic exams