SOUTH SEATTLE COMMUNITY COLLEGE
(Technical Education Division)

COURSE OUTLINE
Revision: Harold Mitchell – Date: January, 2009

DEPARTMENT:	Technical Education
CURRICULUM:	Computing Technology
COURSE TITLE:	Introduction to Computer Programming Network Administration
COURSE NUMBER:	CTN 131
TYPE OF COURSE:	Introduction to Computer Programming
COURSE LENGTH:	1 quarter
CREDIT HOURS:	5
LECTURE HOURS:	55
LAB HOURS:	0
CLASS SIZE:	24

PREREQUISITES:

CSC 100 (Beginning Computers) or Instructor's permission

COURSE DESCRIPTION:

Programming in a structured modular language with emphasis on program design and style. Includes understanding a problem, formal definition, simple graphic design methodologies and program specification through pseudo-coding, elementary searching and sorting algorithms. Programming structures are taught along with sequential file I/O techniques.

STUDENT LEARNING OUTCOMES ADDRESSED:

1. Critical Thinking and Problem Solving - Use skills for analysis of programming problems and selection of algorithms.
2. Computation - Use mathematical skills to develop algorithms and verify program outputs
3.

STUDENT LEARNING OUTCOMES ADDRESSED (CONT'D):

3. Technology - Select and use appropriate programming constructs to solve problems.
4. Information Literacy - Use textbook, programming references and online help to access necessary information.

PROGRAM OUTCOMES ADDRESSED:

3a Select, implement appropriate troubleshooting tools and methods for problem solving.
3g Make use of software applications for utilitarian or presentation purposes.
3h Design a non-complex program in a structured modular/visual language.
4a Use critical thinking for analysis of hardware, OS, or network problems.
4c Work effectively with others to accomplish complex tasks.
4d Develop logical thinking skills.
4e Develop effective communication skills.
4f Be able to explain and communicate problems accurately and the related solutions effectively.

GENERAL COURSE OBJECTIVES:

At the end of the course the student will:

1. Apply the concepts of structured programming.
2. Code before test, after test and fixed iteration loops.
3. Code simple and compound condition tests.
4. Code simple arrays and sorts.
5. Understand and use subroutines and Functions
6. Code simple data entry and inquiry routines.
7. Understand and use simple file Input/Output processing

TOPICAL OUTLINE:	APPROX. HOURS
I. File I/O	5
II. Problem definition	3
III. Solution design	10
IV. Data types defined	2
V. Decisions	5
VI. Functions	5
VII. Iterations	10
VIII. Arrays	10
IX. Sorting	5
Total Hours	55

Written By: T. Phillips
REVISED BY: Harold Mitchell
DATE: January 2009