

14

14

# Glassboro State College Senate Curriculum Committee

## Approval Form

0704.548

Proposal Title: Programming Languages: Theory, Implementation and Application

Sponsor(s) Don Stone Dept.: Math/Computer Sci. Ext. 7323

Check one:  Course  Specialization  Concentration  Minor  Achievement Certificate  
 Certification Program  Major Program  Minor Change (please name: deletion or credit/title/catalog change)

Undergraduate  Graduate 3 Credit Hours

<p><b>Step 1 (Department)</b></p> <p><input checked="" type="checkbox"/> Approved <u>10/19/87</u> Date</p> <p><input type="checkbox"/> Not Approved</p> <p><u>Ronald J. Gocher</u> Dept. CC Chairperson</p> <p><input checked="" type="checkbox"/> Reviewed <u>10-21-87</u> Date</p> <p><u>[Signature]</u> Dept. Chairperson</p>	<p><b>Step 2 (Receipt)</b></p> <p><input type="checkbox"/> SCC# <u>87-88-26</u></p> <p>Proposal Received <u>11/18/87</u> Date</p> <p><u>Brenda A. Belay</u> SCC Chairperson</p>	<p><b>Step 3 (School CC)</b></p> <p>Reviewed <u>2/15/89</u></p> <p><input checked="" type="checkbox"/> Approved <input type="checkbox"/> Not Approved</p> <p><b>Comments:</b></p> <p><u>Ronald J. Gocher</u> School Ctr. Comm. Chairperson</p>
--	---	--

**Step 4 (Academic Dean) Comments:**

Recommend  
 Not Recommend  
 Conditionally Recommend (see comments)

Reviewed 2-16-89  
Date

[Signature]  
Signature, Dean of School

**Step 5 (SCC)**

Open Hearing 3/16/89  Approved by Senate Curriculum Committee 3/16/89  
Date Date

Returned to sponsor(s) for the following reasons:  
Pre-reg. clarified OK. Pass  
Edit cat. desc OK.

**Step 6 (Senate)**

Presented to Senate 3/17/89  Approved  Not Approved  
Date

Notification to Vice-President for Academic Affairs 3/31/89 Brenda A. Belay  
Date Signature, SCC Chairperson

**Step 6 (Senate)**

Received 4/3/89  
Date

Approved  YES  No

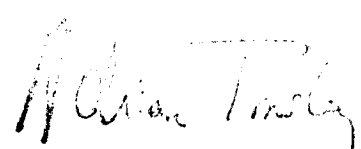
If no, reasons are as follows:

Student credit hours 3

Faculty load hours 2

Equalized credit hours 3

Official copy and approval sheet filed 4/27/89  
Date



Signature, Vice-President for Academic Affairs

**Registrar**

Approved course description received \_\_\_\_\_  
Date

Hegis Taxonomy and Course Number assigned 0704, 548 Rec. KRC 4/10/89

Signature, Registrar

Date

**Notification forwarded:**

- Senate Curriculum Committee Chairperson
- Department Chairperson(s)
- Academic Dean(s)
- Registrar
- Sponsor(s)

Glassboro State College  
Department of Mathematics and Computer Science

Course Proposal

Programming Languages: Theory, Implementation and Application

1. Details:

- a. Course title: Programming Languages: Theory, Implementation and Application
- b. Sponsor: Don Stone, Department of Mathematics and Computer Science
- c. Credit hours: 3
- d. Course level: Graduate
- e. Curricular effect: This course is intended to be part of the Computer Science Academic Specialization for the master's degree in community college teaching.
- f. Prerequisite: C794-542 (Computer Science III - Data Structures).
- g. Suggested time and scale of implementation: Fall 1989, if feasible to be offered initially once a year.
- h. Adequacy of the present staff, resources and library facilities: the present staff, resources (including computer hardware and software) and library facilities are more than adequate for the proposed course.

2. Rationale: The proposed course would cover material that, as well as being important in its own right, is also useful background for topics covered in early Computer Science courses, such as Pascal or Data Structures and Algorithms, courses which would be likely to be taught by people obtaining the community college teaching degree with specialization in Computer Science. The Mathematics/Computer Science Department is also considering proposing a master's degree program in applied Computer Science, and the proposed course could be a part of that program as well.

3. Essence of the course:

- a. Objectives: This is an intermediate graduate course in Computer Science intended to acquaint the student with the major categories of programming languages and to familiarize the student with one or two languages in each category. The student will demonstrate a working knowledge of all (or almost all) of the languages studied by completing programming projects in those languages. In addition, the student will learn formal mechanisms for specifying the syntax and semantics of programming languages and techniques for implementing data structures and control structures in them. The student will also investigate software development environments. On a space-availability basis, undergraduate students may be admitted to sections of

this course (for undergraduate credit). There will be exceptions of them; for example, some test questions may be labeled as optional or extra credit for undergraduates.

1. Typical outline:
  1. Introduction
    - 1.1 Survey of language design issues
    - 1.2 Historical perspective
    - 1.3 Data abstraction
    - 1.4 Control abstraction
  2. The structure of programming languages
    - 2.1 Formal specification of syntax
    - 2.2 Semantics
    - 2.3 Language processing: interpretation and translation
    - 2.4 Binding
    - 2.5 Variables
    - 2.6 Program units
    - 2.7 Run-time structure of programming languages
  3. Data types
    - 3.1 Built-in types
    - 3.2 User-defined types
    - 3.3 Issues relating to types
  4. Control structures
    - 4.1 structured programming constructs
    - 4.2 Unit-level control structures
  5. Imperative programming languages
    - 5.1 Pascal (focusing on implementation issues)
    - 5.2 C
    - 5.3 Ada
  6. Object-oriented programming languages
    - 6.1 Smalltalk
  7. Functional programming languages
    - 7.1 Lisp
    - 7.2 APL
  8. Logic programming languages
    - 8.1 PROLOG
  9. Software development methodologies and environments
2. Evaluation and grading procedure: the student's grade will be based on programming projects and examinations. We anticipate giving at least four programming projects; a typical set of projects would be three C programs, one Lisp program and one PROLOG program. The C programs would be of progressively greater difficulty, involving progressively more features of the language and increasing in scope and complexity. The examinations will include:
  1. questions about language theory and implementation issues,
  2. questions about sample programs,
  3. problems where the students write programs to satisfy given specifications.

- d. Course evaluation: we plan to evaluate the course by means of student evaluations. In addition, formal review of the program or programs in which the course is used will include evaluations of the course.

4. Consultations:

- a. Internal to the Department of Mathematics and Computer Science.
- b. Dr. Richard K. Smith, Program Coordinator of the M.A. in Community College Education.

5. Additional Information:

Possible textbooks:

- Allen E. Tucker, Programming Languages, McGraw-Hill, 1986.
- Russell J. MacLennan, Principles of Programming Language, Second Edition, Holt, Rinehart and Winston, 1987.

5. Catalog Description:

0704.520 [suggested number]

Programming Languages, Theory, Implementation and Application

(Prerequisite: 0704.542; [Note that the prerequisite number will change if the early Computer Science graduate courses are renumbered.]

This ~~is an intermediate graduate course in computer science intended to~~ **will** acquaint the student with the major categories of programming languages and ~~to~~ familiarize the student with one or two languages in each category. The student will demonstrate a working knowledge of most of the languages studied by completing programming projects in those languages. ~~In addition,~~ **Students** will learn formal mechanisms for specifying the syntax and semantics of programming languages and techniques for implementing data structures and control structures in them. ~~The student will also investigate software development environments.~~

Glassboro State College  
Department of Mathematics and Computer Science

Course Proposal

Programming Languages: Theory, Implementation and Application

1. Details:

- a. Course title: Programming Languages: Theory, Implementation and Application
- b. Sponsor: Don Stone, Department of Mathematics and Computer Science
- c. Credit hours: 3
- d. Course level: Graduate
- e. Curricular effect: This course is designed to be part of the Computer Science academic specialization for the master's degree in community college teaching.
- f. Prerequisite: 0704.542 (Computer Science III - Data Structures).
- g. Suggested time and mode of implementation: Fall 1982. If feasible to be offered initially once a year.
- h. Adequacy of the present staff, resources and library facilities: the present staff, resources (including computer hardware and software) and library facilities are more than adequate for the proposed course.

2. Rationale: The proposed course would cover material that, as well as being important in its own right, is also useful background for topics covered in early Computer Science courses, such as Present on Data Structures and Algorithms, courses which would be likely to be taught by people obtaining the community college teaching degree with specialization in Computer Science. The Mathematics/Computer Science Department is also considering proposing a master's degree program in applied Computer Science, and the proposed course could be a part of that program as well.

3. Essence of the Course:

- a. Objectives: This is an intermediate graduate course in Computer Science intended to acquaint the student with the major categories of programming languages and to familiarize the student with one or two languages in each category. The student will demonstrate a working knowledge of all (or almost all) of the languages studied by completing programming projects in those languages. In addition, the student will learn formal mechanisms for specifying the syntax and semantics of programming languages and techniques for implementing data structures and control structures in them. The student will also investigate software development environments. In a space available, he/she will refer to other students may be admitted to a team of

this course than undergraduate credit. Less will be expected of them; for example, some test questions may be labeled as optional or extra credit for undergraduates.

- b. Typical outline:
  1. Introduction
    - 1.1 Survey of language design issues
    - 1.2 Historical perspective
    - 1.3 Data abstraction
    - 1.4 Control abstraction
  2. The structure of programming languages
    - 2.1 Formal specification of syntax
    - 2.2 Semantics
    - 2.3 Language processing: interpretation and translation
    - 2.4 Binding
    - 2.5 Variables
    - 2.6 Program units
    - 2.7 Run-time structure of programming languages
  3. Data types
    - 3.1 Built-in types
    - 3.2 User-defined types
    - 3.3 Issues relating to types
  4. Control structures
    - 4.1 Structured programming constructs
    - 4.2 Unit-level control structures
  5. Imperative programming languages
    - 5.1 Pascal (focusing on implementation issues)
    - 5.2 C
    - 5.3 Ada
  6. Object-oriented programming languages
    - 6.1 Smalltalk
  7. Functional programming languages
    - 7.1 Lisp
    - 7.2 APL
  8. Logic programming languages
    - 8.1 PROLOG
  9. Software development methodologies and environments
- c. Evaluation and grading procedure: the student's grade will be based on programming projects and examinations. We anticipate giving at least four programming projects: a typical set of projects would be three C programs, one Lisp program and one PROLOG program. The C programs would be of progressively greater difficulty, involving progressively more features of the language and increasing in scope and complexity. The examinations will include:
  1. questions about language theory and implementation issues.
  2. questions about sample programs.
  3. problem where the students write programs to satisfy given specifications.

d. Course evaluation: We plan to evaluate the course by means of student evaluations. In addition, formal reviews of the program or programs in which the course is used will include evaluations of the course.

4. Consultations:

- a. Internal to the Department of Mathematics and Computer Science.
- b. Dr. Richard E. Smith, Program Coordinator of the M.A. in Community College Education.

5. Additional information:

Possible textbooks:

- Allen B. Tucker, Programming Languages, McGraw-Hill, 1986.
- Bruce J. MacLennan, Principles of Programming Languages, Second Edition, Holt, Rinehart and Winston, 1987.