14-162
Uwucc App - 4/21/15
Senate App - 4/28/15

Template B

Course Revision/Deletion Template

Steps to the approval process:

1. Complete the applicable template(s) and email them to the departmental or program curriculum committee chair.
2. The curriculum chair emails the proposal to the curriculum committee, then to the department/program faculty for a vote and finally to the department/program chair.
3. The department/program chair emails the proposal to curriculum-approval@iup.edu; this email will also serve as an electronic signature.
4. Curriculum committee staff will log the proposal, forward it to the appropriate dean's office(s) for review within 14 days and post it on the X Drive for review by all IUP faculty and administrators. Following the dean's review the proposal goes to the UWUCC/UWGC and the Senate.
5. Questions? Email curriculum-approval@iup.edu.

Contact Person:	Dr. David T. Smith	Email Address:	dtsmith@iup.edu
Proposing Depart/Unit:	Computer Science	Phone:	724-357-4478

Course Revisions *(Check all that apply; fill out categories below as specified; i.e. if only changing a course title, only need to complete Category A information; if Category B need information in both A and B; For Category C, complete entire form)*:

Category A: ☐ Course Prefix/Number Change ☐ Course Title Change ☐ Course Deletion

Category B: ☒ Catalog Description Change ☐ Modify Prerequisite(s)

Category C: ☐ Add Dual Level ☐ Add Liberal Studies *(Complete Template C)* ☐ Change in Class/Lab Hours

☐ Add Distance Education *(Complete Template E)* ☐ Add/Revise TECC *(Complete Template D)* ☒ Course Revision

☐ Credit Hour Change ☐ Other - Click here to enter text.

Current Course Information		Proposed Changes	
Category A *(if not changed leave blank)*			
Current Prefix	Click here to enter text.	Proposed Prefix	Click here to enter text.
Current Number	Click here to enter text.	Proposed Number	Click here to enter text.
Current Course Title	Click here to enter text.	Proposed Course Title	Click here to enter text.
Prerequisite(s)	Click here to enter text.	Proposed Prerequisite(s)	Click here to enter text.
Category B *(if not changed leave blank)*			

Template B

Current Catalog Description	Prerequisite: Grade of C or better in COSC 300 and 310, or permission of instructor Concepts and techniques of systems programming with an emphasis on assembly, linking, loading, and macro processing for user programs. Overviews of higher-level language translation and system control. Programming and research projects.	Proposed Catalog Description	Prerequisite: Grade of C or better in COSC 300 and 310, or permission of instructor An in-depth introduction to a systems programming, system programming language(s) and application of those language(s) to systems level problems. The focus will be on programming constructs that are closely aligned with the architecture of a digital computer including those providing portability between platforms, dynamic allocation and management of virtual memory, complex in-memory data structures, reading/writing binary data using sequential and random access, pointer arithmetic/manipulation, and interaction between threads/processes
Category C *(if not changed leave blank)*			
Number of Credits	(UG) Class Hours – Click here to enter text. (UG) Lab Hours – Click here to enter text. Credits - Click here to enter text.	Number of Credits	(UG) Class Hours – Click here to enter text. (UG) Lab Hours – Click here to enter text. Credits - Click here to enter text.
Current Course (Student Learning) Outcomes	A. Analyze an assembly language to determine the complexity of the assembler and the architecture of the machine that it runs on. B. Describe several ways the passes of an assembler can be constructed to accommodate the architecture and loading. C. Write a two-pass assembler that creates object modules for programs written in a simple assembly language. D. Describe how a linker/loader creates an executable program from an object module and write such a linker/loader for a simple two-pass assembler. E. Describe how a macro processor works and write a simple one. F. Describe the phases of compilation. G. Describe the functioning of the principal parts of an operating system. H. Use the systems programming problem solving approach.	Proposed Course (Student Learning) Outcomes	1. Enumerate and explain the function of the common operating system kernel routines that are provided by an operating system and accessible from a systems programming language. 2. Design, write, and test moderately complicated low-level programs using a systems programming language. 3. Proficiently use a preprocessor to implement code that is portable between different computing platforms. 4. Implement routines that read and write structured binary files such as word processing documents, index systems, or serialized hierarchical data. 5. Use operating system kernel calls from within a programming language to allocate/free virtual memory, initiate and synchronized multiple threads/processes, interact with the file system, set and respond to timers/interrupts. 6. Implement routines that implement complex data structures which superimpose arrays, records, and references on unstructured blocks of memory. 7. Implement programs that exploit the use of pointers to improve efficiency.

Template B

Brief Course Outline *(it is acceptable to copy this from the old syllabus)*	1. What is System Programming?	(4 hrs)
	A. Discussions of the assignments	
	B. Explanations of specific system features	
	C. Suggestions on use of resources	
	2. Operating system functions	(3 hrs)
	A. Device management	
	B. Memory management	
	C. CPU management	
	D. File system management	
	E. Accounting and security	
	F. User services	
	3. Machine Considerations for Assemblers	(3 hrs)
	A. Instruction Formats	
	B. Instruction Types	
	C. Addressing Modes	
	D. Addressable Units and Address Spaces	
	E. Registers	
	F. Data Types and Representations	
	4. Assembly Language Forms	(1 hr)
	A. Statement Formats	
	B. Directives	
	C. Literals	
	5. Assembly Process	
	A. Data Bases	(2 hrs)
	(1) Fixed Tables (Opcodes and Directives)	
	(2) Dynamic Tables (Symbols and Literals)	
	(3) Organization and Access	
	B. Pass 1 Actions	(4 hrs)
	(1) Building Tables	
	(2) Keeping the Location Counter	
	(3) Error Handling	
	(4) Handling Multiple CSECTs	
	C. Pass 2 Actions	(3 hrs)
	(1) Error Handling	
	(2) Parsing Operands and Expressions	
	(3) Code Generation	
	D. One-Pass Assemblers	(1 hr)

Brief Course Outline *(Give sufficient detail to communicate the content to faculty across campus. It is not necessary to include specific readings, calendar, or assignments.)*	1. System Programming and what languages are used?	2 hrs
	A. What is Systems Programming	
	B. Explanations of specific system features	
	C. Assembly for systems programming	
	D. Overview of high level system programming languages	
	2. Operating system functions	3 hrs
	A. Device management	
	B. Memory management	
	C. Process management	
	D. File system management	
	E. Accounting and security	
	F. User services	
	3. Case study of a high level systems programming language	6 hrs
	A. Data types, operators, expressions	
	B. Flow of control	
	C. Functions and program structure	
	D. Scopes	
	E. Pointers, arrays, structures, and unions	
	F. Basic input and output	
	4. Machine considerations and portability	3 hrs
	A. Instruction formats/types between platforms	
	B. Addressing modes and address spaces between platforms	
	C. Registers between platforms	
	D. Data representations between platforms	
	E. Preprocessor directives and portability	
	F. Macros, inline assembly, and typdefs	
	Exam 1	1.5 hrs
	5. Modularization and program assembly	2 hrs
	A. Multi-file development (interfaces, APIs, header files, make files)	
	A. Libraries, archives, and shared objects	
	B. Dynamic and static linking	
	6. Memory Management	3 hrs
	A. Arrays, records, unions from a memory perspective	
	B. Allocation and de-allocation of memory from the operating system	
	C. Pointer casts, arithmetic, navigation, and field references	
	D. Memory corruption issues, detection and resolution	

Template B

6. Linking and Loading
   - A. Object Files (4 hrs)
     - (1) Relocation Information
     - (2) Unresolved Forward References
     - (3) Inter-PSECT References
     - (4) External References
     - (5) Record Forms
   - B. Linking Tasks

     (1 hr)

     (Allocation, Relocation, Resolution and Loading)
   - C. Approaches/Implementations (4 hrs)
     - (1) Absolute Loader
     - (2) Transfer Vectors
     - (3) Direct Linking Loaders
     - (4) Linkage Editor
     - (5) Dynamic Loading
     - (6) Dynamic Linking
   - D. Effects of Memory Management Approaches (1 hr)

7. Macro Processing (6 hrs)
   - A. Types of Macros
   - B. Definitions (including nesting)
   - C. Invocation (including nesting)
   - D. Expansion, Substitution, and Labels
   - E. Macro Processor Organizations
   - F. Interaction with the Assembler
   - G. Functions and Loops

8. Compilation process (3 hrs)
   - A. Lexical analysis
   - B. Syntax analysis (parsing)
   - C. Semantic analysis
   - D. Storage allocation
   - E. Code generation and optimization

7. Input/Output at a systems level 3 hrs
   - A. Binary input and output
   - B. Sequential and random access
   - C. On-disk data structures
   - D. Indexes and other organizational structures

8. Device drivers 2 hrs

9. Files systems and directories 3 hrs
   - A. File system and directory architecture
   - B. Disk architecture
   - C. Access and update of directory attributes
   - D. File Permissions

Exam 2 1.5 hrs

10. Process management 3 hrs
    - A. Threads
    - B. Spawning processes
    - C. Sleep, wait, and nap
    - D. Synchronization

11. Inter-process communication 3 hrs
    - A. Pipes
    - B. Sockets
    - C. Signals and signal handlers
    - D. Shared memory
    - E. Secure Sockets
    - F. Certificates

12. Object-Oriented extensions of a systems programming language 6 hrs
    - A. Class definition including constructors and destructors
    - B. Encapsulation, inheritance, polymorphism
    - C. Derived classes, abstract classes, multiple inheritance
    - D. Generics/templates
    - E. Operator overloading
    - F. Exception handling

Template B

			Total	42 hrs
			Final	2 hrs

Rationale for Proposed Changes (All Categories)

Why is the course being revised/deleted:	This course has not been taught in over fifteen years and has not been updated in over twenty years. The course as described in the last known syllabus of record is archaic with respect to today's state of the computing. The original focus on developing an assembler is now obsolete. Therefore the description, course outcomes, and objectives have been significantly revised to be consistent with the "Systems Programming" courses offered by a dominant portion of universities including Rutgers, California Polytechnic State University, University of Pittsburgh, University of Georgia, and University of Birmingham.
Implication of the Change on: - Program - Other programs - Students	Implications on the Computer Science Program: This change will allow the department to offer the course in a format that provides up-to-date content, therefore, it can again be offered as an elective in all 3 tracks.
For Dual Listed Courses	*List additional learning objectives for the higher-level course* N/A

For Dean's Review

- Are resources available/sufficient for this course? ☐ Yes ☐ No ☐ NA
- Is the proposal congruent with college mission? ☐ Yes ☐ No ☐ NA
- Has the proposer attempted to resolve potential conflicts with other academic units? ☐ Yes ☐ No ☐ NA

Comments: Click here to enter text.