A. COURSE OBJECTIVES

This module covers many of the features and techniques needed for computer programming. The first half of the module is structured so that the necessary components of procedural programming are introduced. The concepts covered are applicable to a whole host of different programming languages. Emphasis is placed on writing correct, efficient and maintainable programs. The second half of the module develops on the earlier techniques but with particular application to object oriented programming. The construction of well-designed interfaces, and program encapsulation and abstraction are discussed. The module is based on a number of example programs and emphasis is placed on coursework with the aim of ensuring that the theory covered in lectures is reinforced by practical programming exercises.

Learning Outcomes

At the end of the module, a student will:

- design, implement, test, debug and document web based computer programs using a variety of current tools and technologies.
- design, implement, test, debug and document at least one other type of computer program such as: game program, database program, object oriented program.
- understand the relationship between computer programs and organizational processes.
- interpret the mathematical concepts of a programming related problem-solving task and translate them into programming logic and expressions.
- identify and predict a program behaviour favoured by a certain microarchitecture of a processor

B. COURSE CONTENT

1. Introduction to programming:
   - Programming paradigms;
   - From Specification through Algorithms to Implementation;
   - Program compilation and testing.
2. Building Elements:
   - Preconditions and post conditions;
Basic data types;
Variables, identifiers and scope.

3. Control structures:
  - Conditionals;
  - Case statements and loops;
  - Correctness issues when programming with loops.

4. Methods:
  - Comparison between iteration and recursion.

5. Programming with objects and classes:
  - Complex data types:
  - Parameter passing by reference and by value;
  - Encapsulation.

6. Arrays and strings
  - Dynamic binding;
  - Multiple inheritance;
  - Interfaces and abstract classes.

7. Program specifications

8. Error handling

9. Methods of testing

C. LABORATORY

1. Applications with articles and crowd.
2. Develop a program menu.
3. Transmission parameters of functions and procedures.
4. Applications of the method breakdowns in successive steps
5. Recursive procedure and functions.
6. Simple linked lists, specific operations
7. Double-linked list
8. Greedy method: connecting steps with minimal cost
9. Buck tracking
10. Files
11. Graphics

D. BIBLIOGRAPHY


E. ASSESSMENT

Type of assessment: exam at the end of semester
Written work to end the app. I Laboratory tests I a computer quiz.
K1 = 40% work during semester laboratory
K2 = 60% practical exam

F. INTERNATIONAL COOPERATION
1. Technical University of Munich
2. University of Trento (Italy)
SYLLABUS
DIGITAL ELECTRONICS

ENGINEERING FACULTY
FIELD / Specialization: INFORMATION TECHNOLOGY
Year of study: I
Semester II

A. COURSE OBJECTIVES

Digital Electronics Technology is a course of study in applied digital logic that encompasses the design and application of electronic circuits and devices found in video games, watches, calculators, digital cameras, and thousands of other devices. Instruction includes the application of engineering and scientific principles as well as the use of Boolean algebra to solve design problems. Using computer software that reflects current industry standards, activities should provide opportunities for students to design, construct, test, and analyze simple and complex digital circuits.

Learning outcomes

On successful completing this module, students will be able to:
- analyse simple combinational logic circuits, using standard logic gates;
- use Boolean algebra and related techniques to simplify logical expressions;
- use the binary number system to carry out basic arithmetic operations, and to implement these operations using digital circuits;
- Develop solutions to specified requirements using processor systems

B. COURSE CONTENT

1. Boolean algebra elements:
   - Axioms and rules of calculation of Boole algebra
   - Algebraic expression and representation of Boolean functions.
   - Minimize Boolean functions
2. Analysis of stochastic processes:
   - Probability theory, random variables
   - Sums and limits of random variable sequences
   - Time and frequency domain, modeling of continuous and discrete random signals, least-squares estimation
3. High-Speed Electronic Devices
   - Compound semiconductors, heterojunctions and their application to electronic
   - Opto-electronic devices such as HBTs, MODFETs, resonant tunnel transistors, injection lasers, detectors and modulators
4. Combinational logic circuits.

- Decoding.
- Memory ROM, PROM, REPROM. Extensions.

5. Sequential logic circuits.

- CBB-SR asynchronous, synchronous and Master-Slave
- Latch's addressable.
- RAM.
- CBB-D Master-Slave. Registers. CBB-T.

6. Advanced topics in semiconductor devices

C. LABORATORY:

1. Basic logic gates
2. Encoder and decoder study
3. Topics of current interest in the field of solid state electron devices
4. Boolean Algebra Applications
5. CBB-D asynchronous and synchronous
6. Counters study
7. Applications

BIBLIOGRAPHY

1. R. J. Tocci, DIGITAL SYSTEMS. Principles and Applications; Prentice Hall, 2002

ASSESSMENT

Written examination with three subjects: two of theory and a problem.
The final mark consists of the exam mark ratio of $2 / 3$ and note from the laboratory to the share of one third of the final mark.

F. INTERNATIONAL COOPERATION
1. University of Massachusetts: Digital Logic Design;
2. Dublin City University: Digital Electronics;
A. COURSE OBJECTIVES

The course seeks acquisition by students of the mechanisms of functioning and structure of the architecture of computing components. It also aims at highlighting the performance gap between functional parts of the computer to search for optimum use.

Learning outcomes

At the end of the course, students will be able to:
- Explain and demonstrate the concept of top down approach
- Make a summary on basic operation and state the significant components in computer.
- Explain and demonstrate the basic concept of computer organization and computer architecture.

B. COURSE CONTENT

1. The computer system
   - Computer Components:
   - Computer function, Interconnection Structures Interconnection, PCI.
   - Internal Memory: Computer Memory System Overview, Semiconductor Main memory, Cache Memory, Advanced DRAM Organization.

2. Summary devices and subtraction

3. Summary devices binary multiplication
   - Characteristics of the computer implementation of binary multiplication operation
   - Summary of device sequential multiplication of numbers represented in sign-magnitude binary
4. Synthesis of binary devices division

- Characteristics of implementation in computer binary division operation.
- Fundamental procedures for binary division.
- Summary of device sharing sequence binary integers without restoring the balance method.
- Combinational array structure for the division.
- Speeding up the operation of binary division.

5. Organization of data processing units

- Organization units in fixed-point arithmetic and logic.
- Organizing the floating point arithmetic units.
- CPU Arithmetic.

C. LABORATORY

1. Representation of information in computer. Sign-magnitude binary Assembly, a complement, supplement.
2. Representation in binary floating point numbers in IEEE 754 formats and IBM S 360 / 370.
3. VHDL simulation of a binary adder with serial transmission.
4. VHDL simulation of a binary adder-low.
5. VHDL simulation of a Carry Lookahead adder.
6. VHDL simulation of a Carry Skip adder.
7. VHDL simulation of an adder Carry Select.
8. VHDL simulation of a BCD adder.
9. VHDL simulation of a combinational device binary multiplication.
10. VHDL simulation of a binary control units for multiplying by the "paper and pencil" technique summarized by the delay element "
11. VHDL simulation of a binary control units for multiplying by the "paper and pencil" technique synthesized using "sequence counter".
12. VHDL simulation of a binary control units for multiplying by Booth's algorithm, synthesized using the technique "member boards".
13. VHDL simulation of a device sharing binary matrix.
D. BIBLIOGRAPHY


E. PROCEDURE FOR ASSESSMENT

Type of assessment: Final exam
K1 = 70% written examination
K2 = 30% lab work during semester

F. F. INTERNATIONAL COOPERATION

1. Politecnico di Torino, Facolta di Ingegneria (Ingegneria dell'Informazione): Processing systems of Microprocessors;
2. Technische Universität Hamburg-Harburg: Microsystem Design;
SYLLABUS
ARTIFICIAL INTELLIGENCE

ENGINEERING FACULTY
FIELD / Specialization: INFORMATION TECHNOLOGY
Year of study: II
Semester II

A. COURSE OBJECTIVES

Artificial Intelligence aims at developing computer applications, which encompasses perception, reasoning and learning and to provide an in-depth understanding of major techniques used to simulate intelligence. This course covers methods for allowing computer based systems (agents) to sense their environment, learn from experience, plan future actions, and infer new facts from their existing knowledge.

Learning Outcomes

The student will develop: an understanding of the scope and methodology of artificial intelligence through the idea of an intelligent agent. They will learn how to design an intelligent agent, giving it (the agent) the skills to sense, act, plan, learn, and reason classification of problem types and recognition of logical structures. A student will be able to:

- Demonstrate understanding of the nature of AI tasks and solutions;
- Recognize and apply suitable knowledge representation schemes and AI methods appropriate to particular problem areas

COURSE CONTENT

1. Intelligent Agents:
   - Agents and environments:
     - Good behavior
     - The nature of environments – structure of agents

2. Learning from observations:
   - Knowledge in learning

3. Communicating, perceiving, acting- uniformed search strategies
4. Problem Solving

- problem solving agents
- example problems
- searching for solutions

5. Neural networks

- Reinforcement learning
- Passive reinforcement learning
- Active reinforcement learning

B. LABORATORY

1. Fuzzy logic:
   - Definition of fuzzy set
   - Membership function
   - Notation of fuzzy set
   - Operations of fuzzy set
   - Fuzzy number and operations
   - Extension principle
   - Fuzzy rule
   - Fuzzy control

2. Production system
   - Production system
   - Inference engine
     - Knowledge base
     - Pattern matching
     - Forward inference

3. Machine translation

4. Working memory

5. Conflict resolution

BIBLIOGRAPHY

1. Toth-Taşcău M., Cinematica şi dinamica roboţilor inteligenţi, Editura Politehnica Timişoara, 2001
4. Slavici, Inteligenţa Artificială, Editura Fundaţiei Ioan Slavici,Timisoara 2009

E. ASSESSMENT

K1 = 70% final examination, two written topics and a case study
K2 = 30% class participation
F. INTERNATIONAL COOPERATION

1. Politecnico di Torino, Facolta di Ingegneria (Ingegneria dell'Informazione): Processing systems of Microprocessors
2. Technische Universität Hamburg-Harburg: Microsystem Design;
SYLLABUS
OPERATING SYSTEMS

ENGINEERING FACULTY
FIELD / Specialization: INFORMATION TECHNOLOGY
Year of study: IV
Semester I

COURSE GOAL

Theoretical understanding of operating systems, knowledge of the role and functions which have components operating systems Understanding interactions between user and operating system and the operating system and hardware, the theoretical illustration in current operating systems (Linux, Windows XP/2003).

Using the Linux operating system and Windows XP/2003, implementation of operating system components and implementations obtained using algorithms to assess the diverse use, development of applications using system calls.

Learning outcomes:

Upon successful completion of this course, the student shall be able to:

- exhibit familiarity with the fundamental concepts of operating systems;
- exhibit competence in recognizing operating systems features and issues; and
- apply a mature understanding of operating system design and how it impacts application systems design and performance.

CONTENT

1. Systems- general view
   - Mainframe Systems - Multiprocessor Systems
   - Distributed Systems – Clustered Systems - Real Time Systems
   - Hardware Protection – System Components – Handheld Systems
   - Operating System Services – System Calls – System Programs

2. Visual Machines - System Design and Implementation

3. System Model
   - Deadlock Characterization – Methods for handling Deadlocks – Deadlock prevention – Deadlock avoidance
   - Threading issues

5. CPU Scheduling
   - Basic Concepts – Scheduling Criteria – Scheduling Algorithms
   - Processor Scheduling – Real Time Scheduling
   - The Critical-Section Problem – Synchronization Hardware
   - Semaphores – Classic problems of Synchronization – Critical regions
B. LABORATORY

1. **Mass Storage Structure**
   - Physical characteristics & performance determinants of disk Technology
   - Connecting disks and processors
   - Disk scheduling
   - Different algorithms with their pros & cons
   - Overcoming disk failures by the use of RAID-technology
   - General idea
   - Different levels of protection

2. **I/O Systems**
   - Transforming application-level commands into device operations
   - Analyze the steps necessary & point out major OS tasks
   - Many details on OS implementation of I/O interface
   - Ports, polling, interrupts, DMA, device drivers, kernel I/O Structure
   - Show how interrupt handling is done in the OS!
   - Why several levels (priorities), masking, etc.?
   - Blocking vs. non-blocking vs. asynchronous I/O
   - Give an example, show principal steps of interaction between user process and OS components

3. **Memory management**
   - Memory partitioning
   - Swapping
   - paging
   - segmentation
   - virtual memory - Concepts, Overlays, Demand paging, Performance of demand paging
   - Allocation algorithms

---

**BIOGRAPHY**

ASSESSMENT

Type of assessment: exam at the end of semester
  K1 = 70% final exam
  K2 = 30% activity during the year

F. INTERNATIONAL COOPERATION
Stanford University
University of Cambridge
University of Massachusetts
SYLLABUS
SOFTWARE ENGINEERING

ENGINEERING FACULTY
FIELD / Specialization: INFORMATION TECHNOLOGY
Year of study: IV
Semester II

COURSE OBJECTIVES

To cover Requirements Analysis, including use cases, traditional requirements gathering techniques, and user interface prototypes as a tool for obtaining. Experience the application of formal specifications, software modeling and development tools, in a collaborative team environment, toward large-scale software development.

Learning Outcomes:

At the end of the course, the student is able to:

- Describe principles, concepts and practice of software engineering.
- Explain the methods and processes of constructing the different types of software systems.
- Apply techniques and tools of software engineering within the context of systematic construction of quality software

B. COURSE CONTENT

1. Introduction to Software Engineering
2. Software Processes
3. Defining requirements for a software product
4. Source Code Management
5. Specification requirements: formal and informal methods
6. Elements of algebraic specifications
7. Software Design: Object-oriented design.
8. Competing systems design, distributed and real time.
9. Modeling the system architecture

C. LABORATORY

1. User interface design.
2. Programming techniques for large systems.
3. Verification and validation software.

4. Information strategy planning, business area analysis

5. Emergent system properties

6. Software Management: the software lifecycle, functional specification

7. Designing a project using the information provided by the course support

D. BIBLIOGRAPHY


E. ASSESSMENT

Type of assessment: Exam at end of semester
K1 = 70% written examination
K2 = 30% lab work

F. INTERNATIONAL COOPERATION

Stanford University, University of Cambridge, University of Massachusetts