

Informatica (2023)

11925 - Computer Architecture

[PDF](#), [ADOC](#).

Learning outcomes

At the end of the course, the student knows the basic architecture of a computer, combinatorial and sequential networks, and the correspondence between assembly and high-level languages. The student is able to design simple combinational and sequential circuits and write assembly code.

Course contents

Organisation of computer systems. Binary systems. Elements of Boolean algebra, Logic Gates, Combinational circuits, Sequential circuits. Memory, CPU and Bus. The ISA level and assembly programming. The operating systems. Mapping from high-level programming languages to lower architectural levels.

93283 - Logics for Informatic (9 CFU)

[PDF](#), [ADOC](#).

Learning outcomes

The student will know propositional calculus and first order logic. He will be able to write and understand logical propositions and to verify them. He will be able to apply generalization and instantiation processes both in mathematics and programming.

Course contents

FIRST PART: LOGICS

Paradoxes and their resolution. Applications of paradoxes to obtain negative results in computer science.

Introduction to axiomatic set theory: ZF, relations, functions, quotients, cardinality. Cantor's theorem.

Propositional languages: syntax and semantics. Satisfiability and semantic equivalence. Syntactical methods: propositional resolution and natural deduction. Soundness and completeness.

First order languages. Predicates, terms, quantifiers. Syntax: free and bound variables. Interpretations. Semantics for a predicative language. Satisfiability and semantic equivalence.

BNF to define grammars. Structural induction and recursion.

Syntactical methods for first order. Natural deduction. Soundness theorem. Completeness theorem. Compactness theorem.

SECOND PART: ALGEBRA

Generalization, abstraction, instantiation in mathematics and computer science. Higher order functions.

Introduction to algebra: semigroups, monoids, groups and their applications to generic programming.

00819 - Programming

[PDF](#), [ADOC](#).

Learning outcomes

At the end of the course, the student knows programming principles, tools and techniques. He/she is able to program in a specific programming language.

Course contents

Introduction to Programming with C++.

Imperative programming in C++: algorithms and programs, data types, assignment, input / output, conditional, iteration, functions, recursion and recursive functions, vectors, records, memory allocation, dynamic data structures (lists, queues, trees)

Object-oriented programming in C++: classes, methods, overloading, inheritance

Use of a development environment.

58414 - Algebra and Geometry

[PDF](#), [ADOC](#).

Learning outcomes

At the end of this course, students are supposed to have learnt some basic notions in abstract algebra (Euclidean algorithm and congruence relations) and in linear algebra. They will be able to solve linear systems and to study linear maps.

Course contents

Linear Systems

Matrices

Gaussian elimination

Real vector spaces and subspaces

Linear maps

Diagonalization of matrices

Orthogonality in $\mathbb{R}^n$

Modular arithmetic, congruence classes modulo n

37635 - Data Structure and Algorithms

[PDF](#), [ADOC](#).

Learning outcomes

At the end of the course the student:

- knows the algorithms to solve basic computational problems on elementary data structures;
- knows the basic techniques to compute the computational complexity of an algorithm;
- knows the computational complexity classes P, NP, and NP-hard;
- is able to design efficient algorithms to solve simple computational problems;
- is able to analyze the computational complexity of basic computational problems;
- is able to realize and present a project for solving basic computational problems.

Course contents

Data structures. Arrays, records, lists, stacks, queues. Trees. Tree visits (preorder, inorder, postorder). Sets. Dictionaries. Binary search. Hash tables. Priority queues. Heaps. Balanced search trees. MFSET. Graphs. DFS and BFS. Design and analysis of algorithms. Computational complexity. Order of growth. Recurrence equations. Lower bounds. Design techniques: divide-&-conquer, greedy, dynamic programming. Sorting: Insertion Sort, Merge Sort, Heap Sort, Quick Sort, Counting Sort, Bucket Sort, Radix Sort. Algorithms on graphs: Minimum Spanning Tree (Prim, Kruskal), Shortest Paths (Bellman-Ford, Dijkstra, Floyd-Warshall). Complexity. The P and NP classes. NP-completeness.

There are also practical lectures in which data structures are implemented and used, and the Object-Oriented paradigm as well as some Java notions are introduced.

00013 - Mathematical Analysis

[PDF](#), [ADOC](#).

Learning outcomes

At the end of the course, the student knows the basic tools of mathematical analysis such as real numbers, limits, continuity, derivatives and integrals. Moreover, she/he can use these tools for studying other disciplines.

Course contents

The number sets: $\mathbb{N}, \mathbb{Z}, \mathbb{Q}, \mathbb{R}$.

The induction principle.

Sequences of real numbers.

Differential calculus for functions of one real variable.

Exponential and logarithmic functions. Trigonometric functions.

Limits. Continuity (local and global properties).

Derivatives, monotony. Local maxima and local minima.

Infinite and infinitesimal asymptotics. Taylor's formula.

Integral calculus for functions of one real variable: primitive and integral, techniques of integration (by parts, by substitution), integral of rational functions. Generalized integrals.

An introduction to differential calculus for function of several variables. Continuity and derivatives for functions of several real variables.

Integral calculus for functions of two real variables.

02023 - Numerical Computing

[PDF](#), [ADOC](#).

Learning outcomes

At the end of the course students learn the basics of Numerical Computation as error analysis, data interpolation, numerical integration, non-linear equations, linear systems. They are able to solve problems of scientific computing.

Course contents

- Floating point numbers and finite arithmetics.
- Direct and iterative numerical methods for the solution of linear systems. The least squares formulation.
- Data and functions interpolation.
- Minimization of functions in one and more variables. Numerical algorithms for roots finding.

Descent methods for multivariable functions minimization.

- Introduction to inverse problems in imaging: denoise, deblur, super-resolution, image reconstruction from projections.
- Exercises in Python.

13477 - Combinatorial Optimisation

[PDF](#), [ADOC](#).

Learning outcomes

At the end of the course, the student knows the foundations of linear optimization and of integer linear optimization; he or she knows the simplex algorithm and knows when a problem has integer solutions. He or she can model a problem in terms of linear (or integer linear) constraints and objective function(s), or realize that this cannot be done. He or she is able to model combinatory problems on graphs as shortest paths, maximum flows and assignments as optimization problems, and solve them with the algorithms from the literature. Finally, he or she is able to recognize whether a given optimization problem is inherently intractable.

Course contents

The following are the main topics of the course: optimization problems, example models, linear programming, graphs and graph models, integer linear programming.

04642 - Probability Calculus and Statistics

[PDF](#), [ADOC](#).

Learning outcomes

At the end of the course, the student knows basic concepts and methods of probability and mathematical statistics. The student can solve simple problems of probability and statistical inference.

Course contents

- Mathematical model of a random experiment: sample space, events, axioms of probability and their consequences.
- Conditional probability and independence: chain rule, total probability rule and Bayes' rule.
- Combinatorics and discrete uniform probability spaces.
- Random variables:

Distribution (or law) and cumulative distribution function.

Discrete and (absolutely) continuous random variables: discrete and continuous probability density functions.

Expected value and variance.

Relevant probability distributions: Bernoulli, binomial, Poisson, discrete and continuous uniform, exponential, normal (or Gaussian).

- Random vectors:

Joint law, marginal laws, joint cumulative distribution function, independence of random variables, covariance.

Discrete random vectors: joint discrete and marginal discrete probability density functions.

- Descriptive statistics: population and sample, types of data, frequencies, tabular and graphical representations; measures of central tendency, measures of variability.

- Bivariate data: joint frequencies and two-way tables; scatter plot; covariance and linear correlation coefficient; method of least squares and linear regression.

- Limit theorems:

Sequence of i.i.d. random variables.

Law of large numbers: Chebyshev's inequality, Monte Carlo method.

Central limit theorem.

- Discrete-time Markov chains: transition matrix, directed graph representation, n-step transition probability, communication classes, invariant distribution.

Readings/Bibliog

88566 - Web Technology (9 ECTS)

[PDF](#), [ADOC](#).

Learning outcomes

At the end of the course, the student knows the most important technologies used in the World Wide Web context. The student is able to create web documents and simple distributed web applications, determine their visual aspects, verify their correctness and universality, and design and verify their usability and user experience.

Course contents

This course is held in the second semester of each A.Y. (February-May). The teacher is usually informed of the timetable and rooms at the very last moment, and never before January of the same year.

The course discusses the following topics:

Fundamentals: VII level protocols, character encodings, standard bodies

Basic web technologies: HTTP, URI, HTML, CSS, XML

Server-side technologies for web applications: php, python, NodeJs

Client-side technologies for web applications: JavaScript, Ajax, JSON, JavaScript frameworks.

Component-based web programming: Angular, React, Vue.

Introduction to some technologies of Semantic Web: RDF, OWL, SPARQL, ontologies.

User Experience Design for web sites: the Garrett model.

04138 - Programming Languages

[PDF](#), [ADOC](#).

Learning outcomes

At the end of the course, the student will know the principal techniques for defining the syntax and the semantics of the most common programming languages; they will also know how to implement the principal constructs.

Course contents

First module, first semester; instructor: Roberto Gorrieri

<http://www.cs.unibo.it/~gorrieri/>

The evolution of programming languages. From assembly to higher level languages. Abstract machines, interpreters and compilers. Description of a programming language: syntax, semantics, pragmatics and implementation. Syntax (BNF). Structured Operational Semantics (SOS). Regular grammars, regular expressions, and finite automata: equivalences and principal theorems (e.g., pumping lemma). Design of lexical analysers. Lex. Context free grammars and push-down automata: equivalences and principal theorems (e.g., pumping theorem). Deterministic context free grammars: algorithms for parsing; grammars LL(1), LR(0), SLR, LR(1), LALR(1). YACC.

Second module, second semester; instructors: Maurizio Gabbrielli and Saverio Giallorenzo

<http://www.unibo.it/SitoWebDocente/default.htm?UPN=maurizio.gabbrielli%40unibo.it>

<https://www.unibo.it/sitoweb/saverio.giallorenzo2>

Environment, scoping rules and their implementation. Stack of the activation records; heap. Memory management: garbage collection. Sequence control, procedures, recursion. Types and type checking. Parameters and parameter passing: by value, by reference, by result, by name. Functional parameters; closures. Exceptions. The object-oriented paradigm: classes and objects, initialization,

inheritance and late-binding. Subtyping is not inheritance. The logical paradigm. The functional paradigm (Scala). The concurrent paradigm and service-oriented computing (Jolie).

93315 - Computer Networks (12 ECTS)

[PDF](#), [ADOC](#).

Learning outcomes

The student will learn:

- the fundamentals of the computer networks;
- main technologies and communication protocols, including the TCP/IP suite for Internet and packet-based communication;
- the Internet architecture, the way Internet operates, inter-process communication and the design and development of inter-process communication protocols, including the principles and technologies for Wireless communications.

Course contents

Foundation topics: definitions, history and development of computer networks.

Topologies, network resources, and logical channels.

Computer Network performances: indexes and their meaning in different application contexts.

Circuit-switched and packet-switched networks.

Network communication protocols.

Network architectures: HW and SW.

Network Service architectures: Client/server, Peer to peer, hybrid.

ISO OSI Reference Model.

Physical layer: transmission medium, signals, encoding/decoding specifically focusing on radio communications.

Data Link layer: communication channels, Medium Access Control techniques, MAC addressing, Reliable communication, Error detection and correction.

Local Area Network technologies: hub, repeater, bridge, switch. LAN connectivity.

LAN topologies and links.

Wireless systems: design of radio communication systems, antennas, wireless propagation, link budget, Decibels, encoding, error detection and correction techniques, MAC protocols (e.g. IEEE 802.11 and Bluetooth). Mobile systems: Mobile IP, TCP variants.

Virtual channels (MPLS) and virtual networks (VLAN).

Network Layer: IPv4 protocol and addressing. IPv6. Domains and hierarchical addressing. Subnetting and supernetting, IPv4 network classes, CIDR, IP configuration. Network Address Translation (NAT). SDN e OpenFlow. ICMP. ARP e RARP. DHCP.

Design of network and subnetworks in IP domains.

Management and configuration of LANs (SNMP).

Troubleshooting and analysis of network performance and issues.

Networks of networks and inter-networking. Forwarding and routing IP (local and ISP-based - interdomain). Router.

Multicasting.

Transport layer: Transmission Control Protocol (TCP), performance of end-to-end communications, Congestion control. Flow control.

Sockets and socket programming (examples) with UDP/TCP.

Inter-process communications over Internet.

Session and Presentation layers.

Application layer: examples of protocols and services at the application layer. SMTP (email), http (WWW), DNS, streaming video, gaming, P2P, VoIP.

Quality of service. Real Time communication.

Security of communication networks. Privacy, cryptography, integrity and digital signature. Secure transport layer (TCP): SSL. Secure network layer. IPsec. Virtual Private Networks (VPN). Firewalls and Intrusion Detection.

08574 - Operating Systems

[PDF](#), [ADOC](#).

Learning outcomes

The objective of the course is to illustrate the structure and the methods to build modern multitasking operating systems. This course also explains how to install, program and administer an operating system.

Course contents

Operating Systems: definition and history

Concurrent Programming

Structure of an O.S.

Scheduling

Resource Management

Main Memory Management

Secondary Memory Management

File Systems

Security of Operating Systems

the C language

Programming Tools

Shell scripting

The Python Language

90107 - DATABASES (9 CFU)

[PDF](#), [ADOC](#).

Learning outcomes

At the end of the course the student: - knows the relational data model ed the basic constructors of SQL; - can design and develop a database; - is capable of processing a project to implement and information system.

Course contents

Databases

Relational data model

Relational algebra and calculus

SQL

Database design methodology

ER data model and quality verification

Laboratory: notions about the architecture of a DBMS, indexes, transactions and design examples

90106 - Software Engineering (9 ECTS)

[PDF](#), [ADOC](#).

Learning outcomes

At the end of the course the student knows the basics of software systems construction: methods and tools for analyzing, designing and measuring the qualities of software products. The student will be able to develop a specification starting from some requirements written in natural language. The student will be able to model a software system using UML and to develop the code using an object oriented language using collaborative versioning and code analysis tools.

Course contents

The course in Italian. •Software products and their development •Software lifecycle •Agile software development methods •Agile Scrum •Requirement engineering •Design patterns •Modeling software with UML •Software development tools •Project Management for software systems •Controlling and measuring software quality •Software maintenance •Configuration management

Readings/Bibliography

93319 - Introduction to machine learning

[PDF](#), [ADOC](#).

Learning outcomes

During the course, the student will be introduced to the complex themes pertaining to the simulation of intelligent behavior by means of machines, with practical experimentation of basic machine learning techniques for different tasks: supervised, unsupervised, with reinforcement. The course will also provide rudiments of image processing, since images will be extensively used as experimental test bench for the aforementioned learning techniques.

Course contents

The first part of the course provides a general introduction to the field of machine learning, in its typical forms: supervised, unsupervised, with reinforcement. Traditional topics such as decision tree learning, logistic regression, Bayesian networks and Support Vector Machines will be covered.

The second part of the course is focused on Neural Networks, and their typical learning mechanism: the backpropagation algorithm. We shall discuss the main types of neural nets: feed forward, convolutional, recurrent, and their practical applications. We shall also investigate techniques to visualize the effect of hidden units (tightly related to deep dreams and inceptionism) as well as several generative approaches comprising Generative Adversarial Networks. Thematics relative to Object Detection and Semantic Segmentation will be also briefly discussed too.

93466 - CYBERSECURITY

[PDF](#), [ADOC](#).

Learning outcomes

The objective of the course is to present the theory, mechanisms, techniques and tools that are effective in increasing the security of a computer system. At the end of the course, the student will be familiar with the mathematical foundations of modern cryptography, authentication, authorization and access control mechanisms that are suitable for achieving confidentiality, integrity and availability of computer systems. The student will also acquire the knowledge necessary to assess the potentials and limitations of current technologies.

Course contents

Vulnerability Assessment

Penetration testing

TOR / Dark Web

OSINT / Data Leak

SDR and radio attacks

One-time pad

Stream Ciphers

Pseudo Random Generators

Real-World Stream Ciphers

Wifi network security

Cryptographic Systems with secret key

Cryptographic Systems with public key

DES

AES

RSA Digital Signatures

Public-key infrastructures

Reverse engineering

PWN

41169 - Theoretical Computer Science (6 Credits)

[PDF](#), [ADOC](#).

Learning outcomes

The student will learn the main notions and results of Computability and Complexity Theory. At the end of course the student will be aware of the theoretical and practical limits of computation, and will be able to use and apply methodologies and techniques typical of formal methods to the study and solution of a wide range of algorithmic problems.

Course contents

Turing machines and equivalent computational models; the Church-Turing thesis

the Halting problem and other undecidable problems; Rice's theorem

Basic Complexity Theory, reductions, P vs NP and other open problems

37925 - Virtual System Design

[PDF](#), [ADOC](#).

Learning outcomes

The objective of the course is to teach the structure and implementation of virtual systems like virtual machines and view based operating systems, virtual networks, virtual file systems. This course is mainly based on laboratory projects.

Course contents

Virtuality: introduction

Virtual Networks

Virtual Machines

Virtual Devices

Virtual File Systems

Virtual Networking stacks. Internet of Threads (IoTh)

User level Operating systems: VUOS.

66860 - Mobile Applications Laboratory

[PDF](#), [ADOC](#).

Learning outcomes

At the end of the course, the student knows methodological and technological aspects, and application development tools for mobile devices both under iOS (iPhone, iPad, iPod Touch) and Android platforms. Students will understand the management of devices with innovative user interfaces, multi-touch, event management, ObjectiveC programming, Xcode and Cocoa Touch, Eclipse and Android SDK, design patterns, I/O, sensors and geo-localization/maps APIs, networking services, debugging and testing of applications. In addition, students will understand the basic issues of applications' execution in wireless mobile scenarios, and will experience the most relevant platforms for mobile applications' development, APIs of internal devices, multimedia management, iPhone and Android SDK and design of applications under a Model-View-Control pattern.

Course contents

Introduction:

overview of technologies for iPhone, iPod Touch e iPad (and iOS in general).

overview of Android Technology

iOS Module:

iOS technology layers: Core OS, Core Services, Media, Cocoa Touch.

iOS e iOS SDK. Development tools for iOS: Xcode, Storyboard, Simulator.

Swift and Swift UI language (notes on differences with ObjectiveC).

Model-View-Controller.

Target, Action, Outlets.

Foundation Framework and UIKit (Cocoa Touch), user interface, UIWindow e UIView.

UIViewController and MultiViews, controllers and views.

Touch events and Multi-touch, gestures.

Debugging and Testing of iOS apps incrementally developed in classes.

Android Module:

The history of Android versions

Installing the Android SDK

The Android Architecture

Java and Kotlin

The Android Resources System

Android Activities and Fragments

Android Intents

Android Layouts, Widgets and Events

Android Menu, Dialog and Toasts

Android Services and Background

Android Data Management

Maps Support (Google Maps, Mapbox, ...)

Android Network

Android Design guidelines and patterns

Android Navigation

Android System Services (alarms, sensors, vibration, audio)

Hybrid development framework

37459 - Business Strategy

[PDF](#), [ADOC](#).

Learning outcomes

This course aims at introducing the student to the main strategic issues at the business level. At the end of the course the student knows what a strategy is and how a competitive advantage can be achieved through the analysis of the industry and of the internal resources and capabilities. He/She understands the importance of business models, including the role of organizational structures and technological innovation. The student is able to define a business strategy and to evaluate a business strategy by situating the business in its context

Course contents

Strategy and results

Sector structure and company positioning

Sector dynamics and strategic change

Strategy evaluation

Strategy evaluation in the multi-business enterprise