COURSE DESCRIPTION CARD - SYLLABUS

Course name

Introduction to Computing

Course

Field of study

Computing

Year/Semester

1/1

Area of study (specialization)

Profile of study

general academic

Level of study

First-cycle studies

Course offered in

English

Form of study

full-time

Requirements

compulsory

Number of hours

Lecture

30

Laboratory classes

16

Other (e.g. online)

0

Tutorials

0

Projects/seminars

0

Number of credit points

4

Lecturers

Responsible for the course/lecturer:

Prof. Dr Habil. Jerzy Nawrocki

email: Jerzy.Nawrocki@cs.put.poznan.pl

tel. (0-61) 665-3422

Faculty of Computing and Telecommunications

Piotrowo 3, 60-965 Poznań

Responsible for the course/lecturer:

Ph. D. Wojciech Complak

email: Wojciech.Complak@cs.put.poznan.pl

tel. (0-61) 665-2983

Faculty of Computing and Telecommunications

Piotrowo 3, 60-965 Poznań

Prerequisites

Student starting this course should have basic knowledge of mathematics in accordance with the core curriculum of general education for second-level schools and the ability to obtain information from the indicated sources.

Students should also understand the necessity to broaden own competences/be ready to cooperate within the team. In addition, in the field of social competence, the student must present such attitudes as honesty, responsibility, perseverance, cognitive curiosity, creativity, personal culture, respect for other people.

Course objective

Familiarising students with the basic areas of computer science, as well as support in the acquisition of

basic programming skills. The course presents programming paradigms and basic models, concepts and techniques used in computer science. The mastery of the presented material provides future IT specialists with the foundations necessary for further studies and future professional work.

Course-related learning outcomes

Knowledge

1 has a structured and theoretically founded general knowledge in the field of key issues of computer science, and detailed knowledge in the field of selected issues in this discipline of science
2. has knowledge of important directions of development and the most important achievements of computer science and other related scientific disciplines, in particular electronics, telecommunications as well as automatic control and robotics
3. has basic knowledge about the life cycle of IT systems, both hardware and software, and in particular about the key processes taking place in them
4. has knowledge of ethical codes related to IT, is aware of the dangers related to electronic crime, and understands the specificity of mission-critical systems
5. has basic knowledge of patents, the copyright and related rights act and the act on the protection of personal data and technology transfer, in particular with regard to IT solutions .

Skills

1. can properly use information and communication techniques, applicable at various stages of the implementation of IT projects
2. can see in the process of formulating and solving IT tasks also non-IT aspects, in particular social, legal and economic issues
3.can organise, cooperate and work in a group, assuming different roles in it, and can properly define priorities for the implementation of a task set by himself or others

Social competences

1. understands that in computer science knowledge and skills very quickly become obsolete
2. is aware of the importance of knowledge in solving engineering problems and knows examples and understands the causes of malfunctioning information systems that have led to serious financial and social losses or to serious health conditions or even to death

Methods for verifying learning outcomes and assessment criteria

Learning outcomes presented above are verified as follows:
Formative assessment:
a) lectures:
- based on answers to questions related to subjects covered during former lectures,

b) laboratory classes:
- based on assessment of progress of implementation of assigned tasks,

Total assessment:
Verification of assumed learning objectives is based on:
- evaluation of preparation for individual laboratory sessions and assessment of skills related to solving

laboratory exercises,
-continual assessment, during each class (verbal answers) - rewarding the progress in the ability to use the principles and methods learned,
- assessment of knowledge and skills related to the implementation of laboratory tasks through test/s during the semester,
- assessment of knowledge and skills presented during the written exam after the end of the semester.

Additional elements cover:
- discussing additional aspects of the class topic,
- ability of using the acquired knowledge while solving a given problem
- remarks related to improving teaching materials,
- pointing out perceptual difficulties enabling ongoing improvement of the teaching process

Programme content

The course covers the following content:
Imperative programming and the C & Python programming languages (simple programs, declaring and processing simple variables, reading and printing variables, conditional statement, loop statements, arrays, records, modularisation concept, functions).
Digital circuits (Boolean algebra, logic gates, adder, flip-flop, register, implementation of logic operations in C). Von Neumann architecture and assembly language (elements of processor architecture, basic instructions, simple programs, hexadecimal system, integers and representation of negative numbers, von Neumann model, jump instructions, calling a subprogram using the call/ret instruction).
Text processing in C & Python (string representation, reading, processing and printing strings, elements of the standard input/output library) and in AWK (AWK language concept, simple programs, patterns, regular expressions, variables and standard functions).
Fundamentals of object-oriented programming in C++ & Python.
Concurrent and parallel programming (evolution of operating systems, processes, implementation of threads in C and Python, processor time sharing, computation interference, pthreads library, thread management, access synchronisation, semaphores, producer-consumer problem, reader-writer problem, deadlock problem).
Databases (evolution, relational data model, SQL language, queries, projection, selection, data management, patterns, query nesting).
Artificial intelligence and natural language (Turing test, ELIZA program, parts of speech and ambiguity, lexical analysis and lex generator, formal grammars, translation, context-free grammars).
Computer networks (computer network architecture, network communication, internet protocol stack, packets, data nesting, TCP sockets, web services).
Computational complexity and numerical methods (dynamic memory allocation, dynamic data structures, graphs, lists, stacks, queues, heap sort, computational complexity, real number representation, numerical stability, numerical algorithms).
Microcontrollers and computer control systems (specificity of real-time systems, control system architecture, bit manipulation, communication with external devices, Arduino architecture, basics of Arduino programming in C).

Software engineering (problem analysis, functional requirements and use cases, selected UML diagrams, initial user interface design, non-functional requirements, architecture, implementation and testing). Professionalism in IT (principles of effective operation, win-win, synergy).

Teaching methods

1. lecture: multimedia presentation, presentation illustrated with examples shown on the blackboard, solving tasks, programming tools demonstration,
2. laboratory classes: solving tasks, discussion

Bibliography

Basic

1.Język ANSI C, B. W. Kernighan, D. M. Ritchie, Wydawnictwa Naukowo-Techniczne, dowolne wydanie
2. Układy cyfrowe, B. Wilkinson, WKiŁ, Warszawa, 2000
3. Wprowadzenie do przetwarzania tekstów w języku AWK, J. Nawrocki, W. Complak, ProDialog 2, 23-46, Poznań, 1994.
4. Sieci komputerowe, J.F. Kurose, K.W. Ross, Helion, 2006

Additional

1. 7 nawyków skutecznego działania, S. Covey, Rebis, 2003
2. Język C. Szkoła programowania, Wydanie VI, Prata S., Helion, 2016

Breakdown of average student's workload

	Hours	ECTS
Total workload	100	4,0
Classes requiring direct contact with the teacher	46	2,0
Student's own work (literature studies, preparation for laboratory classes, preparation for tests/exam, project preparation) [1]	54	2,0

[1] delete or add other activities as appropriate