

# Theoretical foundations of computer science - opis przedmiotu

| Informacje ogólne   |                                                                   |
|---------------------|-------------------------------------------------------------------|
| Nazwa przedmiotu    | Theoretical foundations of computer science                       |
| Kod przedmiotu      | 11.3-WE-INFP-TFCS-Er                                              |
| Wydział             | <a href="#">Wydział Informatyki, Elektrotechniki i Automatyki</a> |
| Kierunek            | Informatyka                                                       |
| Profil              | ogólnoakademicki                                                  |
| Rodzaj studiów      | Program Erasmus pierwszego stopnia                                |
| Semestr rozpoczęcia | semestr zimowy 2022/2023                                          |

| Informacje o przedmiocie        |                                                                                                                                       |
|---------------------------------|---------------------------------------------------------------------------------------------------------------------------------------|
| Semestr                         | 2                                                                                                                                     |
| Liczba punktów ECTS do zdobycia | 5                                                                                                                                     |
| Typ przedmiotu                  | obowiązkowy                                                                                                                           |
| Język nauczania                 | angielski                                                                                                                             |
| Sylabus opracował               | <ul style="list-style-type: none"><li>dr hab. inż. Piotr Borowiecki, prof. UZ</li><li>prof. dr hab. inż. Andrzej Obuchowicz</li></ul> |

| Formy zajęć |                                            |                                           |                                               |                                              |                        |
|-------------|--------------------------------------------|-------------------------------------------|-----------------------------------------------|----------------------------------------------|------------------------|
| Forma zajęć | Liczba godzin w semestrze<br>(stacjonarne) | Liczba godzin w tygodniu<br>(stacjonarne) | Liczba godzin w semestrze<br>(niestacjonarne) | Liczba godzin w tygodniu<br>(niestacjonarne) | Forma zaliczenia       |
| Wykład      | 30                                         | 2                                         | -                                             | -                                            | Egzamin                |
| Ćwiczenia   | 30                                         | 2                                         | -                                             | -                                            | Zaliczenie na<br>ocenę |

## Cel przedmiotu

- familiarize students with the basic concepts of theoretical computer science, like algorithm, computational complexity, formal language and grammar, deterministic and nondeterministic finite-state automata.
- gaining basic skills in determining the complexity of algorithms, proving halting property and partial correctness of algorithms.
- intruducing main concepts of automata, languages and computation theory, e.g. Chomsky hierarchy, Church-Turing thesis, P and NP complexity classes.
- familiarize students with parallel computation theory.

## Wymagania wstępne

Algorithms and Data Structures, Computational Logic, Linear Algebra, Mathematical Analysis.

## Zakres tematyczny

Algorithms and their properties: correctness, partial correctness, halting property; proving correctnes and the halting property.

Computational complexity of algorithms (worst and average case complexity). Upper and lower bound on the compexity of an algorithmic problem, inherent complexity and the complexity gap. Determining the complexity of algorithms. Notation for asymptotic growth rates.

Foundations of automata theory, languages and computation: grammars and languages, finite automata, regular languages and regular expressions, context-free grammars and context-free languages, normal forms of grammars, derivation trees, pushdown automata, context-sensitive languages and linear bounded automata, Turing machines and recursively enumerable languages, the Chomsky hierarchy.

Counter automatons and Turing machines. Accepting vs deciding. Time and space complexity. Random access machine. Classes of algorithmic problems: P, NP NP-complete and PSPACE. Proving NP-completeness. Open problems on computational complexity. Limits of computabiliity and decidability. Church-Turing thesis.

Parallel and probabilistic algorithms. Product complexity. Chandra-Stockmeyer parallel computation thesis. Nick's class. RNC algorithms.

## Metody kształcenia

**Lectures:** conventional lectures

**Exercises:** whiteboard problem solving, presentations, discussions

## Efekty uczenia się i metody weryfikacji osiągnięcia efektów uczenia się

| Opis efektu                                                     | Symbole efektów | Metody weryfikacji                                   | Forma zajęć                                            |
|-----------------------------------------------------------------|-----------------|------------------------------------------------------|--------------------------------------------------------|
| Students can estimate the time complexity of a given algorithm. |                 | <ul style="list-style-type: none"><li>exam</li></ul> | <ul style="list-style-type: none"><li>Wykład</li></ul> |

| Opis efektu                                                                                                                                                                                           | Symbole efektów | Metody weryfikacji | Forma zajęć |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------|--------------------|-------------|
| Students are creative in solving problems, taking into account available techniques and resources                                                                                                     |                 | • test             | • Ćwiczenia |
| Students are able to carry out an analysis of the correctness of the sequential algorithm written in block form and in a structured programming language                                              |                 | • test             | • Ćwiczenia |
| Students know the concept of computational complexity regarding parallel algorithms.                                                                                                                  |                 | • exam             | • Wykład    |
| Students are able to define and characterize the classes of the complexity of algorithmic problems.<br>Knows the basic concepts of algorithmic correctness theory, automata, grammars, and languages. |                 | • exam             | • Wykład    |

## Warunki zaliczenia

**Lectures:** the passing condition is to obtaining a positive grade from the final exam.

**Exercises:** the passing condition is to obtain a positive grade from all tests.

**Course:** it is necessary to pass both lectures and exercises.

**Calculation of the final grade:** lecture 50% + exercises 50%

## Literatura podstawowa

1. Cormen T. H., Leiserson C. E., Rivest R. L.: Introduction to algorithms, MIT Press, Boston, 1994
2. Aho A. V., Hopcroft J. E., Ullman J.D.: Algorithms and Data Structures, Addison-Wesley Longman Publishing Co., Inc. Boston 1983
3. Dasgupta S., Papadimitriou Ch., Vazirani U.: Algorithms, McGraw-Hill, New York 2008
4. Harel D.: Algorithmics, the Spirit of Computing, Addison-Wesley Publ., Reading, 1987
5. Hopcroft J. E., Ullmann J.D.: Introduction to Automata Theory, Languages, and Computation, Addison-Wesley Publ., Reading, 2003

## Literatura uzupełniająca

1. Graham R.L., Knuth D.E., Patashnik O.: Matematyka konkretna, PWN, Warszawa, 2002
2. Papadimitriou C. H.: Złożoność obliczeniowa, Helion, Gliwice, 2012
3. Ross K. A., Wright C. R. B.: Matematyka dyskretna, PWN, Warszawa, 2000

## Uwagi

Zmodyfikowane przez dr hab. inż. Piotr Borowiecki, prof. UZ (ostatnia modyfikacja: 21-04-2022 01:01)

Wygenerowano automatycznie z systemu SylabUZ