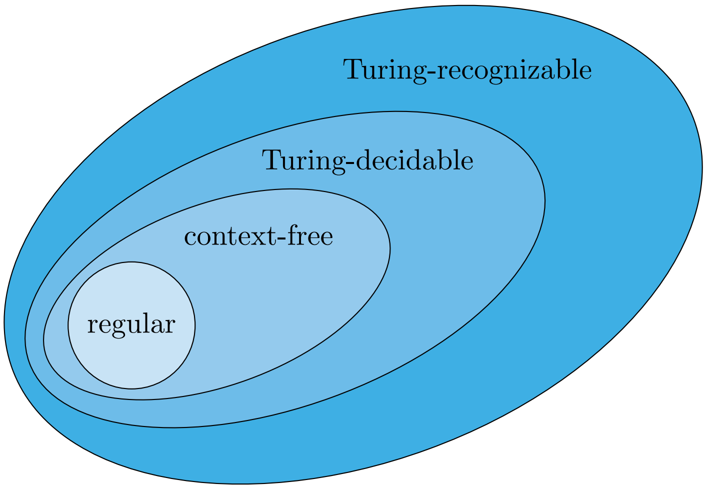
# Institute of Business Administration

## CSE 309: Theory of Automata

(Tentative Course Outline and Syllabus)




IBA SMCS
School of Mathematics and Computer Science

**Spring 2025**

# Institute of Business Administration

School of Mathematics and Computer Science

# CSE 309: Theory of Automata

*"Those who can imagine anything, can create the impossible."*
– **Alan Turing**

## 1 Logistics

| | Section-1 | Section-2 and Section-3 |
|---|---|---|
| **Instructors:** | **Shahid Hussain** | **Imran Rauf** |
| **Email:** | shahidhussain@iba.edu.pk | irauf@iba.edu.pk |
| **Phone** | 3044 | 1645 |
| **Office:** | 204 | 206 |
| **Office hours:** | Mondays and Wednesday<br>11:30 AM – 12:30 PM<br>or by appointment | TBA |
| **Discord:** | https://discord.gg/RZPKZUHk | |
| **Teaching Assistants:** | Zuha Aqib (z.aqib.26106@khi.iba.edu.pk)<br>Zara Masood (z.masood.26928@khi.iba.edu.pk) | |

Last updated: *January 20, 2025*

## 2 Course Objectives

This course is about the beautiful contributions and insights that the field computer science has afforded us. We shall specifically focus on the interplay between notions of logical proof and how they link with limits on computation; which in turn has implications on what can be computed and how efficiently in the physical world. The perspective we gain by looking at our world through the computational lens is perhaps the main contribution of theoretical computer science to the human endeavor.

The course starts from basic math preliminaries, works its way through historical breakthrough results in the field and then branches off into an investigation of fundamental challenges at the frontiers of theoretical computer science. The course provides opportunities for both:

(i) practice in developing a rigorous mathematical argument, and

(ii) practice in developing a clear intuitive explanation of a complex argument.

During Introduction to Programming and Discrete Math students learn a new language, i.e., Discrete Math. They obtain some practice in reading, writing, speaking and listening to this new language. In a human languages class, a logical progression is to read and understand the existing classic literature in the language. Similarly, we will engage ourselves with some of the classics in theoretical computer science.

The course content is spread over three distinct modules. The first deals with what can be computed in principle: computability, the second deals with how efficiently we can compute: complexity and finally we branch off into an investigation of fundamental challenges at the frontiers of theoretical computer science.

## 3 Program Learning Outcomes/Graduate Attributes

Graduate attributes (program learning outcomes - PLO's) taken from `https://www.seoulaccord.org/document.php?id=79`.

**PLO-1. Academic Education**

*[Educational depth and breadth]*

Completion of an accredited program of study designed to prepare graduates as computing professionals

**PLO-2. Knowledge for Solving Computing Problems**

*[Breadth and depth of education and type of knowledge, both theoretical and practical]*

Apply knowledge of computing fundamentals, knowledge of a computing specialization, and mathematics, science, and domain knowledge appropriate for the computing specialization to the abstraction and conceptualization of computing models from defined problems and requirements

**PLO-3. Problem Analysis**

*[Complexity of analysis]*

Identify, formulate, research literature, and solve complex computing problems reaching substantiated conclusions using fundamental principles of mathematics, computing sciences, and relevant domain disciplines

**PLO-4. Design / Development of Solutions**

*[Breadth and uniqueness of computing problems, i.e., the extent to which problems are original and to which solutions have previously been identified or codified]*

Design and evaluate solutions for complex computing problems, and design and evaluate systems, components, or processes that meet specified needs with appropriate consideration for public health and safety, cultural, societal, and environmental considerations

**PLO-5. Modern Tool Usage**

*[Level and appropriateness of the tool to the type of activities performed]*

Create, select, adapt and apply appropriate techniques, resources, and modern computing tools to complex computing activities, with an understanding of the limitations

**PLO-6. Individual and Team Work**

*[Role in, and diversity of, the team]*

Function effectively as an individual and as a member or leader in diverse teams and in multi-disciplinary settings

**PLO-7. Communication**

*[Level of communication according to type of activities performed]*

Communicate effectively with the computing community and with society at large about complex computing activities by being able to comprehend and write effective reports, design documentation, make effective presentations, and give and understand clear instructions

**PLO-8. Computing Professionalism and Society**

*[No differentiation in this characteristic except level of practice]*

Understand and assess societal, health, safety, legal, and cultural issues within local and global contexts, and the consequential responsibilities relevant to professional computing practice

**PLO-9. Ethics**

*[No differentiation in this characteristic except level of practice]*

Understand and commit to professional ethics, responsibilities, and norms of professional computing practice

**PLO-10. Life-long Learning**

*[No differentiation in this characteristic except level of practice]*

Recognize the need, and have the ability, to engage in independent learning for continual development as a computing professional

## 4 Course Learning Outcomes (CLO's)

| Course Learning Outcomes (CLO's) | |
|---|---|
| **CLO-1** | Design automata, context-free grammars, and Turing machines accepting or generating certain languages |
| **CLO-2** | Construct arguments to show that certain computational questions are uncomputable |
| **CLO-3** | Classify different computational problems in their appropriate complexity classes |

### 4.1 CLO's to PLO's Mapping

| | Academic Education | Knowledge for Solving computing Problems | Problem Analysis | Design / Development of Solutions | Modern Tool Usage | Individual and Team Work | Communication | Computing Professionalism and Society | Ethics | Life-long Learning |
|---|---|---|---|---|---|---|---|---|---|---|
| | **PLO-1** | **PLO-2** | **PLO-3** | **PLO-4** | **PLO-5** | **PLO-6** | **PLO-7** | **PLO-8** | **PLO-9** | **PLO-10** |
| **CLO-1** | | ✓ | | | | | | | | |
| **CLO-2** | | | ✓ | | | | | | | |
| **CLO-3** | | | | ✓ | | | | | | |

| | CLO-1 | CLO-2 | CLO-3 |
|---|---|---|---|
| | Design automata, context-free grammars, and Turing machines accepting or generating certain languages | Construct arguments to show that certain computational questions are uncomputable | Classify different computational problems in their appropriate complexity classes |
| Problem sets | ✓ | | |
| Quizzes | ✓ | ✓ | |
| Presentation | ✓ | ✓ | |
| Midterm | ✓ | ✓ | ✓ |
| Final | ✓ | ✓ | ✓ |

## 5 Course Texts

The main text for the course is

- *Introduction to the Theory of Computation* by **Michael Sipser**, 3rd edition.

Based on our choice of topics, we may refer to following texts during the course.

- *Introduction to Automata Theory, Languages, and Computation* by **John Hopcroft, Rajeev Motwani, and Jeffrey Ullman**.
- *Feynman Lectures on Computation* by **Richard P. Feynman**
- *People and Ideas in Theoretical Computer Science* (edited) by **Cristian Calude**.
- *Selected Papers on Computer Science* by **Donald Knuth**.

If necessary, additional readings will be made available online.

## 6 Grading Procedures (Tentative)

Grades will be computed as follows.[1]

| Assessments | Weight |
|---|---:|
| Problem sets (homework) $\times 4$ | 16% |
| Quizzes $\times 5$ $(n-1)$ | 16% |
| Topic presentation | 8% |
| Midterm | 25% |
| Final | 35% |

[1] This weight distribution is subject to change.

### 6.1 Problem Sets

One of the main tools for learning the concepts we discuss in class are the problem sets. The 'personal struggle' you engage in with these problem sets will allow you to develop the skills necessary for success as a theoretical computer scientist. Always spend some time thinking about these problems on your own before asking for hints, looking up solutions etc. Do not go in search of solutions online; learning the material happens when you are working on problems rather than looking up complete solutions.

You are welcome to collaborate on problem sets, provided that (1) you write up your solutions individually, and (2) you clearly cite the names of all collaborators and sources. Failure to do so will result in *zero credit.* An additional key requirement is that you should be able to explain what you submit. Inability to do so will result again in zero credit.

### 6.2 Quizzes

We will have five quizzes in this course and quizzes are one of the three ways to test individual and independent understanding of the contents of this course.

### 6.3 Exams

We will have two exams in this course (a midterm and a final). One around the mid of the semester and another (*a comprehensive exam*) at the end of the semester.

## 7 Late Work and Makeup Policy

No late solutions will be accepted and no make-up for exams or any of the quizzes will be given. If you have a valid medical excuse (for any of the quizzes, problems set, exams, etc.), the percentage of your grade corresponding to the missed work will be shifted to the final exam. Valid excuses require supporting documentation.

## 8 Attendance Policy

IBA attendance policy applies.

## 9 Academic Integrity

Each student in this course is expected to abide by the IBA Code of Conduct. Scholastic dishonesty shall be considered a serious violation of these rules and regulations and is subject to strict disciplinary action as prescribed by IBA regulations and policies. Scholastic dishonesty includes, but is not limited to, cheating on exams, plagiarism on assignments, and collusion.

- **PLAGIARISM:** Plagiarism is the act of taking the work created by another person or entity and presenting it as ones own for the purpose of personal gain or of obtaining academic credit. Plagiarism includes the submission of or incorporation of the work of others without acknowledging its provenance or giving due credit according to established academic practices. This includes the submission of material that has been appropriated, bought, received as a gift, downloaded, or obtained by any other means. Students must not, unless they have been granted permission from all faculty members concerned, submit the same assignment or project for academic credit for different courses.

- **CHEATING:** The term cheating shall refer to the use of or obtaining of unauthorized in- formation in order to obtain personal benefit or academic credit.
- **COLLUSION:** Collusion is the act of providing unauthorized assistance to one or more person or of not taking the appropriate precautions against doing so. Any student violating academic integrity a second time in this course will receive a failing grade for the course, and additional disciplinary sanctions may be administered.

## 10 Course Schedule (tentative)

It is a work in progress!!

| | | |
|---|---|---|
| **Week 1:** | **January 20 – January 26** | |
| | **Introduction and preliminaries** | |
| | Alphabet, strings, languages | |
| **Week 2:** | **January 27 – February 2** | Quiz 1 |
| | **Finite automata** | |
| | Deterministic finite automata | |
| | Regular languages and regular expressions | |
| **Week 3:** | **February 3 – February 9** | Homework 1 |
| | **Context-free languages** | |
| | Context-free grammars | |
| | Ambiguity | |
| | Chomsky normal form | |
| **Week 4:** | **February 10 – February 16** | Quiz 2 |
| | **Turing machines** | |
| | Basic definitions | |
| | Multi-tape Turing machines | |
| | Nondeterministic Turing machines | |
| | Equivalence with other models | |
| **Week 5:** | **February 17 – February 23** | Homework 2 |
| | **Decidability** | |
| | Undecidability | |
| | The diagonalization method | |
| | The halting problem | |
| | Turing-unrecognizable language | |
| **Week 6:** | **February 24 – March 2** | Quiz 3 |
| | **Reducibility** | |
| | Mapping reducibility | |
| | Post correspondence problem | |
| | Rice's theorem | |
| **Week 7:** | **March 3 – March 9** | Homework 3 |
| | **Advanced topics in computability theory-I** | |
| **Week 8:** | **March 10 – March 16** | |
| | **Advanced topics in computability theory-II** | |

| | | |
|---|---|---|
| **Week 9:** | **March 17 – March 23** | |
| | **Midterm exam** | |
| **Week 10:** | **March 24 – March 30** | |
| | **Mid semester break** | |
| **Week 11:** | **March 31 – April 6** | |
| | **Time Complexity** | |
| | The class P | |
| | The class NP | |
| | Polynomial time reducibility | |
| **Week 12:** | **April 7 – April 13** | Quiz 4 |
| | **NP-completeness** | |
| | The Cook-Levin theorem | |
| | More on NP-completeness | |
| **Week 13:** | **April 14 – April 20** | Homework 4 |
| | **Space complexity** | |
| | Savitch's theorem | |
| | The class PSPACE | |
| | PSPACE-completeness | |
| **Week 14:** | **April 21 – April 27** | Quiz 5 |
| | **More on space complexity** | |
| **Week 15:** | **April 28 – May 4** | |
| | **Randomization** | |
| | The class BPP | |
| | Alterations | |
| | Interactive proof systems | |
| **Week 16:** | **May 5 – May 11** | |
| | **Presentations-I** | |
| **Week 17:** | **May 12 – May 18** | |
| | **Presentations-II** | |
| **Week 18:** | **May 19 – May 20** | |
| | ***Nothing*** | |
| | **May 21 – June 3** | Final Exams |