Computer Science

Ashoka University

NEP Recommended Curriculum and Guidelines

Contents

1 Computer Science Programs

The Computer Science department offers the following programs

Major/Minor in Computer Science
1 | 4-year BSc Hons in Computer Science (See 1.1)
2 | 4-year BSc Hons with Research in Computer Science (See 1.2)
3 | 3-year BSc in Computer Science (See 1.3)
4 | Minor in Computer Science (See 1.4)
5 | Concentration in Computer Science (See 1.5)
Interdisciplinary Majors: CS + X
6 | 4-year BSc Hons in Computer Science and Mathematics (See 1.6)
7 | 4-year BSc Hons in Computer Science and Entrepreneurial Leadership (See 1.7)
8 | 4-year BSc Hons in Philosophy and Computer Science (See 1.8)

1.1 4 Year BSc Hons

The 4-year BSc Hons degree in Computer Science mandates a minimum of 150 credits for completion, ensuring a well-rounded education encompassing both core computer science knowledge and broader academic experiences.

Total Credits 150						
Academic Credits 144					Non-academic Credits 6	
FC Credits 36	CS Credits 80			Open Credits 28	Co-curricular Credits 4	Internship Credits 2
	Core 64	Project 4	Electives 12			

Credits Breakup for the 4 Year BSc Hons Program in CS

These credits are classified into two categories:

- Academic Credits (144): A minimum of 144 academic credits. These credits are divided into three categories - FC credits (36), CS credits (80), and Open credits (28).
  - FC Credits (36): A total of 36 credits dedicated to foundational courses (https://www.ashoka.edu.in/programme/foundation-courses/)
  - CS Credits (80): A minimum of 80 credits from the Computer Science Department, divided as follows:
    - The student must complete 68 credits of CS core courses given in §1.1.1
    - Additionally, students must complete a minimum of 12 credits in CS elective courses. Independent Study Modules offered by CS faculty are eligible as CS electives, with a maximum of 4 credits allowed through ISMs.
    - To fulfill the major requirements, students must achieve a minimum grade of "B" in both the Introduction to Computer Science and Discrete Mathematics courses.
  - Open Credits (28): The remaining 28 academic credits can be earned by taking courses from any department within the university, including the Computer Science Department.
- Non-Academic Credits (6): A minimum of 6 non-academic credits. They include
  - 4 credits designated for co-curricular courses
  - 2 credits allocated for internship experience

1.1.1 Computer Science Core for 4 Year BSc Hons

Core Courses	Credits
Basic Science and Maths	
P&S	4
Linear Algebra	4
Calculus (Calculus Enabler)	4
4-credits from Physics Primer (See §3.1.1)	4
4-credits from Biology Primer (See §3.1.1)	4
Computational thinking	
Introduction to Computer Science	4
Discrete Mathematics	4
Data Structures and Algorithms	4
Introduction to Machine Learning	4
Data Science and Management	4
Theory of Computation	4
Design and Analysis of Algorithms	4
Programming Languages and Translation	4
Information Security	2
Systems and software	
Computer Organisation and Systems	4
Design Practices in CS	4
Computer Networks	2
Capstone Project	4
Total	68

1.1.2 Example Path for 4 Year BSc Hons.

Monsoon Semester	**Spring Semester**
1st Semester	2nd Semester
Calculus	Introduction to Computer Science
	Discrete Mathematics
3rd Semester	4th Semester
Probability and Statistics	Design and Analysis of Algorithms
Linear Algebra	Computer Organisation and Systems
Data Structures and Algorithms	Computer Networks (2 credits)
	Information Security (2 credits)
5th Semester	6th Semester
Design Practices in CS	Theory of Computation
Introduction to Machine Learning	Data Science and Management
7th Semester	8th Semester
Capstone Project	
Programming Languages and Translation	

** Incorporate one Physics and one Biology course, along with 12 credits of Computer Science electives, within your four-year curriculum in addition to the mentioned courses

1.2 4 Year BSc Hons with Research

The 4-year BSc Hons with Research degree in Computer Science mandates a minimum of 150 credits for completion, ensuring a well-rounded education encompassing both core computer science knowledge and broader academic experiences.

Total Credits 150						
Academic Credits 144					Non-academic Credits 6	
FC Credits 36	CS Credits 92			Open Credits 16	Co-curricular Credits 4	Internship Credits 2
	Core 64	Thesis 12	Electives 16			

Credits Breakup for the 4 Year BSc Hons with Research Program in CS

These credits are classified into two categories:

- Academic Credits (144): A minimum of 144 academic credits. These credits are divided into three categories - FC credits (36), CS credits (92), and Open credits (16).
  - FC Credits (36): A total of 36 credits dedicated to foundational courses (https://www.ashoka.edu.in/programme/foundation-courses/)
  - CS Credits (92): A minimum of 92 credits from the Computer Science Department, divided as follows:
    - 64 credits of CS core courses and a 12-credit capstone thesis, as specified in §1.2.1
    - Additionally, students must complete a minimum of 16 credits in CS elective courses. Independent Study Modules offered by CS faculty are eligible as CS electives, with a maximum of 4 credits allowed through ISMs.
    - To fulfill the major requirements, students must achieve a minimum grade of ”B” in both the Introduction to Computer Science and Discrete Mathematics courses.
  - Open Credits (16): The remaining 16 academic credits can be earned by taking courses from any department within the university, including the Computer Science Department.
- Non-Academic Credits (6): A minimum of 6 non-academic credits. They include
  - 4 credits designated for co-curricular courses
  - 2 credits allocated for internship experience

1.2.1 CS Core for 4 Year BSc Hons with Research

Core Courses	Credits
Basic Science and Maths	
P&S	4
Linear Algebra	4
Calculus (Calculus Enabler)	4
4-credits from Physics Primer (See §3.1.1)	4
4-credits from Biology Primer (See §3.1.1)	4
Computational thinking	
Introduction to Computer Science	4
Discrete Mathematics	4
Data Structures and Algorithms	4
Introduction to Machine Learning	4
Data Science and Management	4
Theory of Computation	4
Design and Analysis of Algorithms	4
Programming Languages and Translation	4
Information Security	2
Systems and software	
Computer Organisation and Systems	4
Design Practices in CS	4
Computer Networks	2
Capstone Thesis	12
Total	76

1.2.2 Example Path for 4 Year BSc Hons with Research

Monsoon Semester	**Spring Semester**
1st Semester	2nd Semester
Calculus	Introduction to Computer Science Discrete Mathematics
3rd Semester	4th Semester
Probability and Statistics Linear Algebra Data Structures and Algorithms	Design and Analysis of Algorithms Computer Organisation and Systems Computer Networks (2 credits) Information Security (2 credits)
5th Semester	6th Semester
Design Practices in CS Introduction to Machine Learning	Theory of Computation Data Science and Management
7th Semester	8th Semester
Capstone Thesis (4 Credits) Programming Languages and Translation	Capstone Thesis (8 Credits)
** Incorporate one Physics and one Biology course, along with 16 credits of Computer Science electives, within your four-year curriculum in addition to the mentioned courses	

Capstone Thesis: A total of 12 credits for the capstone thesis, with evaluation as follows:

- 4 credits in the 7th semester, evaluating the work done over that semester
- The remaining 8 credits in the final semester, evaluating the entire year of work

1.3 3 Year BSc

For students who exit at the end of 3 years, they receive a 3-year BSc Hons degree in Computer Science. The later mandates a minimum of 114 credits for completion.

Total Credits 114					
Academic Credits 108				Non-academic Credits 6	
FC Credits 36	CS Credits 60		Open Credits 12	Co-curricular Credits 4	Internship Credits 2
	Core 52	Electives 8			

Credits Breakup for the 3 Year BSc Hons Program in CS

These credits are classified into two categories:

- Academic Credits: A minimum of 108 academic credits. The academic credits includes
  - 36 credits dedicated to foundational courses (https://www.ashoka.edu.in/programme/foundation-courses/)
  - A minimum of 60 credits from the Computer Science Department, divided as follows:
    - 52 credits of CS core courses, as specified in §1.3.1
    - Additionally, students must complete a minimum of 8 credits in CS elective courses. Independent Study Modules offered by CS faculty are eligible as CS electives, with a maximum of 4 credits allowed through ISMs.
    - To fulfill the major requirements, students must achieve a minimum grade of ”B” in both the Introduction to Computer Science and Discrete Mathematics courses.
  - The remaining 12 academic credits can be earned by taking courses from any department within the university, including the Computer Science Department.
- Non-Academic Credits: A minimum of 6 non-academic credits. They include
  - 4 credits designated for co-curricular courses
  - 2 credits allocated for internship experience

1.3.1 CS Core for 3 Year BSc

Core Courses	Credits
Basic Science and Maths	
P&S	4
Linear Algebra	4
Calculus (Calculus Enabler)	4
4-credits from Physics Primer (See §3.1.1)	4
4-credits from Biology Primer (See §3.1.1)	4
Computational thinking	
Introduction to Computer Science	4
Discrete Mathematics	4
Data Structures and Algorithms	4
Introduction to Machine Learning	4
Data Science and Management	4
Information Security	2
Systems and software	
Computer Organisation and Systems	4
Design Practices in CS	4
Computer Networks	2
Total	52

1.3.2 Example Path for 3 Year BSc

Monsoon Semester	Spring Semester
1st Semester	2nd Semester
Calculus	Introduction to Computer Science
	Discrete Mathematics
3rd Semester	4th Semester
Probability and Statistics	Computer Organisation and Systems
Linear Algebra	Computer Networks (2 credits)
Data Structures and Algorithms	Information Security (2 credits)
5th Semester	6th Semester
Design Practices in CS	Data Science and Management
Introduction to Machine Learning	
** Incorporate one Physics and one Biology course, along with 8 credits of Computer Science electives, within your four-year curriculum in addition to the mentioned courses	

1.4 Minor in Computer Science

To obtain a Minor in Computer Science, you are required to successfully complete 24 academic credits, typically equivalent to six courses, offered by the Computer Science department. The following courses are mandatory:

- Introduction to Computer Science
- Discrete Mathematics
- Data Structures and Algorithms

In addition, you must select three more courses from the Computer Science department. ISMs are not considered as valid CS electives for Minor requirements.

1.5 Concentration in Computer Science

To obtain a Concentration in Computer Science, you are required to successfully complete 16 academic credits, typically equivalent to 4 courses, offered by the Computer Science department. The following courses are mandatory:

- Introduction to Computer Science,
- Discrete Mathematics,
- Data Structures and Algorithms

In addition, you must select one more course from the Computer Science department. ISMs are not considered as valid CS electives for concentration requirements.

1.6 Computer Science and Mathematics Interdisciplinary Major

For the Computer Science and Mathematics Interdisciplinary major we offer the following two streams of synergistic specialisations.

1. **Cryptography, Number theory, Algebra**
2. **Data Science and Machine Learning**

1.6.1 4-year BSc Hons in Computer Science and Mathematics

The 4-year BSc Hons in "Computer Science and Mathematics Interdisciplinary Major" mandates a minimum of 150 credits for completion. The credit requirements are as follows.

Total Credits 150							
Academic Credits 144						Non-academic Credits 6	
FC Credits 36	CS+Math Credits 80			Open Credits 28		Co-curricular Credits 4	Internship Credits 2
	Core 66	Project 4	Electives 10				

Credits Breakup for the 4 Year BSc Hons Program in CS + Math

These credits are classified into two categories:

- Academic Credits (144): A minimum of 144 academic credits. These credits are divided into three categories - FC credits (36), CS credits (80), and Open credits (28).
  - FC Credits (36): A total of 36 credits dedicated to foundational courses (https://www.ashoka.edu.in/programme/foundation-courses/)
  - CS+Math Credits (80): A minimum of 80 credits, divided as follows:
    - The student must complete 70 credits of CS+Math core courses given in §1.6.2
    - Additionally, take a minimum of 10 credits in elective courses, offered by CS and Mathematics department. ISMs offered by CS/Math faculty are eligible as electives, with a maximum of 4 credits allowed through ISMs.
    - To fulfill the major requirements, students must achieve a minimum grade of "B" in both the Introduction to Computer Science and Discrete Mathematics courses.
  - Open Credits (28): The remaining 28 academic credits can be earned by taking courses from any department within the university, including the Computer Science department and the Mathematics department.
- Non-Academic Credits (6): A minimum of 6 non-academic credits. They include
  - 4 credits designated for co-curricular courses
  - 2 credits allocated for internship experience

1.6.2 Computer Science and Mathamatics Core for 4 Year BSc Hons

Cryptography, number theory, algebra

Core Courses	Credits
Basic Science	
4-credits from Physics Primer (See §3.1.1)	4
4-credits from Biology Primer (See §3.1.1)	4
Computer Science	
Computational thinking	
Introduction to Computer Science	4
Discrete Mathematics	4
Data Structures and Algorithms	4
Theory of Computation	4
Design and Analysis of Algorithms	4
Cryptography	4
Information Security	2
Systems and software	
Computer Organisation and Systems	4
Mathematics	
P&S	4
Linear Algebra	4
Calculus (Calculus Enabler)	4
Algebra 1	4
Multivariate Calculus	4
Statistical Inference	4
Real Analysis	4
Capstone Project	4
Total	70

Data Science and Machine Learning

Core Courses	Credits
Basic Science	
4-credits from Physics Primer (See §3.1.1)	4
4-credits from Biology Primer (See §3.1.1)	4
Computer Science	
Computational thinking	
Introduction to Computer Science	4
Discrete Mathematics	4
Data Structures and Algorithms	4
Introduction to Machine Learning	4
Data Science and Management	4
Design and Analysis of Algorithms	4
Information Security	2
Systems and software	
Computer Organisation and Systems	4
Mathematics	
P&S	4
Linear Algebra	4
Calculus (Calculus Enabler)	4
Algebra 1	4
Multivariate Calculus	4
Statistical Inference	4
Real Analysis	4
Capstone Project	4
Total	70

1.6.3 Example Path: CS+Math 4 year BSc (To be completed)

Monsoon Semester	**Spring Semester**
1st Semester	2nd Semester
Calculus	Introduction to Computer Science Discrete Mathematics
3rd Semester	4th Semester
Probability and Statistics Linear Algebra Data Structures and Algorithms	Theory of Computation Computer Organisation and Systems
5th Semester	6th Semester
Cryptography Introduction to Machine Learning Design and Analysis of Algorithms	Data Science and Management
7th Semester	8th Semester
Capstone Project	
** Incorporate one Physics and one Biology course, along with 12 credits of Computer Science electives, within your four-year curriculum in addition to the mentioned courses	

1.6.4 4-year BSc Hons with Research in Computer Science and Mathematics

The 4-year BSc Hons with Research in "Computer Science and Mathematics Interdisciplinary Major" mandates a minimum of 150 credits for completion. The credit requirements are as follows.

Total Credits 150						
Academic Credits 144					Non-academic Credits 6	
FC Credits 36	CS+Math Credits 92			Open Credits 16	Co-curricular Credits 4	Internship Credits 2
	Core 66	Project 12	Electives 14			

Credits Breakup for the 4 Year BSc Hons Program in CS + Math

These credits are classified into two categories:

- Academic Credits (144): A minimum of 144 academic credits. These credits are divided into three categories - FC credits (36), CS+Math credits (92), and Open credits (16).
  - FC Credits (36): A total of 36 credits dedicated to foundational courses (https://www.ashoka.edu.in/programme/foundation-courses/)
  - CS+Math Credits (92): A minimum of 92 credits, divided as follows:
    - The student must complete 78 credits of CS+Math core courses given in §1.6.5
    - Additionally, take a minimum of 14 credits in elective courses, offered by CS and Mathematics department. ISMs offered by CS/Math faculty are eligible as electives, with a maximum of 4 credits allowed through ISMs.
    - To fulfil the major requirements, students must achieve a minimum grade of "B" in both the Introduction to Computer Science and Discrete Mathematics courses.
  - Open Credits (16): The remaining 16 academic credits can be earned by taking courses from any department within the university, including the department of Computer Science and Mathematics Department.
- Non-Academic Credits (6): A minimum of 6 non-academic credits. They include

- 4 credits designated for co-curricular courses
- 2 credits allocated for internship experience

1.6.5 Computer Science and Mathematics Core for 4 Year BSc Hons with Research

Cryptography, number theory, algebra

Core Courses	Credits
Basic Science	
4-credits from Physics Primer (See §3.1.1)	4
4-credits from Biology Primer (See §3.1.1)	4
Computer Science	
Computational thinking	
Introduction to Computer Science	4
Discrete Mathematics	4
Data Structures and Algorithms	4
Theory of Computation	4
Design and Analysis of Algorithms	4
Cryptography	4
Information Security	2
Systems and software	
Computer Organisation and Systems	4
Mathematics	
P&S	4
Linear Algebra	4
Calculus (Calculus Enabler)	4
Algebra 1	4
Multivariate Calculus	4
Statistical Inference	4
Real Analysis	4
Capstone Thesis	12
Total	78

Data Science and Machine Learning

Core Courses	Credits
Basic Science	
4-credits from Physics Primer (See §3.1.1)	4
4-credits from Biology Primer (See §3.1.1)	4
Computer Science	
Computational thinking	
Introduction to Computer Science	4
Discrete Mathematics	4
Data Structures and Algorithms	4
Introduction to Machine Learning	4
Data Science and Management	4
Design and Analysis of Algorithms	4
Information Security	2
Systems and software	
Computer Organisation and Systems	4
Mathematics	
P&S	4
Linear Algebra	4
Calculus (Calculus Enabler)	4
Algebra 1	4
Multivariate Calculus	4
Statistical Inference	4
Real Analysis	4
Capstone Thesis	12
Total	78

1.7 Computer Science and Entrepreneurial Leadership Major

1.7.1 4 Year BSc Hons in Computer Science and Entrepreneurship

The 4-year BSc Hons in "Computer Science and Entrepreneurship Interdisciplinary Major" mandates a minimum of 150 credits for completion. The credit requirements are as follows.

Total Credits 150						
Academic Credits 144					Non-academic Credits 6	
FC Credits 36	CS+ENT Credits 80			Open Credits 28	Co-curricular Credits 4	Internship Credits 2
	Core 64	Project 4	Electives 12			

Credits Breakup for the 4 Year BSc Hons Program in CS + ENT

These credits are classified into two categories:

- Academic Credits (144): A minimum of 144 academic credits. These credits are divided into three categories - FC credits (36), CS+ENT credits (80), and Open credits (28).
  - FC Credits (36): A total of 36 credits dedicated to foundational courses (https://www.ashoka.edu.in/programme/foundation-courses/)
  - CS+ENT Credits (80): A minimum of 80 credits, divided as follows:

    - The student must complete 68 credits of CS+ENT core courses given in §1.7.2
    - Additionally, take a minimum of 12 credits in elective courses, offered by CS and ENT department. ISMs offered by CS/ENT faculty are eligible as electives, with a maximum of 4 credits allowed through ISMs.
    - To fulfill the major requirements, students must achieve a minimum grade of "B" in both the Introduction to Computer Science and Discrete Mathematics courses.
  - Open Credits (28): The remaining 28 academic credits can be earned by taking courses from any department within the university, including the Computer Science Department and ENT department.
- Non-Academic Credits (6): A minimum of 6 non-academic credits. They include
  - 4 credits designated for co-curricular courses
  - 2 credits allocated for internship experience

1.7.2 Computer Science and Entrepreneurship Core for 4 Year BSc Hons

Core Courses	Credits
Basic Science	
P&S	4
Linear Algebra	4
Calculus (Calculus Enabler)	4
4-credits from Physics Primer (See §3.1.1)	4
4-credits from Biology Primer (See §3.1.1)	4
Computer Science	
Computational thinking	
Introduction to Computer Science	4
Discrete Mathematics	4
Data Structures and Algorithms	4
Introduction to Machine Learning	4
Information Security	2
Systems and software	
Computer Organisation and Systems	4
Design Practices in CS	4
Computer Networks	2
Entrepreneurship	
Course 1	4
Course 2	4
Course 3	4
Course 4	4
Capstone Project	4
Total	68

1.7.3 4 Year BSc Hons with Research in Computer Science and Entrepreneurship

The 4-year BSc Hons with Research in "Computer Science and Entrepreneurship Interdisciplinary Major" mandates a minimum of 150 credits for completion. The credit requirements are as follows.

Total Credits 150							
Academic Credits 144						Non-academic Credits 6	
FC Credits 36	CS+ENT Credits 92				Open Credits 16	Co-curricular Credits 4	Internship Credits 2
	Core 64	Thesis 12	Electives 16				

Credits Breakup for the 4 Year BSc Hons Program in CS + ENT

These credits are classified into two categories:

- Academic Credits (144): A minimum of 144 academic credits. These credits are divided into three categories - FC credits (36), CS+ENT credits (80), and Open credits (28).
  - FC Credits (36): A total of 36 credits dedicated to foundational courses (https://www.ashoka.edu.in/programme/foundation-courses/)
  - CS+ENT Credits (80): A minimum of 80 credits, divided as follows:
    - The student must complete 76 credits of CS+ENT core courses given in §1.7.4
    - Additionally, take a minimum of 16 credits in elective courses, offered by CS and ENT department. ISMs offered by CS/ENT faculty are eligible as electives, with a maximum of 4 credits allowed through ISMs.
    - To fulfill the major requirements, students must achieve a minimum grade of "B" in both the Introduction to Computer Science and Discrete Mathematics courses.
  - Open Credits (16): The remaining 16 academic credits can be earned by taking courses from any department within the university, including the Computer Science Department and ENT department.
- Non-Academic Credits (6): A minimum of 6 non-academic credits. They include
  - 4 credits designated for co-curricular courses
  - 2 credits allocated for internship experience

1.7.4 Computer Science and Entrepreneurship Core for 4 Year BSc Hons with Research

Core Courses	Credits
Basic Science	
P&S	4
Linear Algebra	4
Calculus (Calculus Enabler)	4
4-credits from Physics Primer (See §3.1.1)	4
4-credits from Biology Primer (See §3.1.1)	4
Computer Science	
Computational thinking	
Introduction to Computer Science	4
Discrete Mathematics	4
Data Structures and Algorithms	4
Introduction to Machine Learning	4
Information Security	2
Systems and software	
Computer Organisation and Systems	4
Design Practices in CS	4
Computer Networks	2
Entrepreneurship	
Course 1	4
Course 2	4
Course 3	4
Course 4	4
Capstone Thesis	12
Total	76

1.8 Philosophy and Computer Science Interdisciplinary Major

1.8.1 4-year BSc Hons in Computer Science and Philosophy

The 4-year BSc Hons in "Computer Science and Philosophy Interdisciplinary Major" mandates a minimum of 150 credits for completion. The credit requirements are as follows.

These credits are classified into two categories:

Total Credits 150							
Academic Credits 144						Non-academic Credits 6	
FC Credits 36	CS+Phil Credits 80			Open Credits 28		Co-curricular Credits 4	Internship Credits 2
	Core 68	Project 4	Electives 8				

Credits Breakup for the 4 Year BSc Hons Program in CS + Math

- Academic Credits (144): A minimum of 144 academic credits. These credits are divided into three categories - FC credits (36), CS+Phil credits (80), and Open credits (28).
  - FC Credits (36): A total of 36 credits dedicated to foundational courses (https://www.ashoka.edu.in/programme/foundation-courses/)
  - CS+Phil Credits (80): A minimum of 80 credits, divided as follows:
    - The student must complete 72 credits of CS+Phil core courses given in §1.8.2
    - Additionally, take a minimum of 8 credits in elective courses, offered by CS and Philosophy department. ISMs offered by CS/Philosophy faculty are eligible as electives, with a maximum of 4 credits allowed through ISMs.
    - To fulfill the major requirements, students must achieve a minimum grade of "B" in both the Introduction to Computer Science and Discrete Mathematics courses.
  - Open Credits (28): The remaining 28 academic credits can be earned by taking courses from any department within the university, including the Computer Science department and the Philosophy department.
- Non-Academic Credits (6): A minimum of 6 non-academic credits. They include
  - 4 credits designated for co-curricular courses
  - 2 credits allocated for internship experience

1.8.2 Computer Science and Philosophy Core for 4 Year BSc Hons

Core Courses	Credits
Basic Science	
Probability and Statistics	4
Linear Algebra	4
Calculus (Calculus Enabler)	4
4-credits from Physics Primer (See §3.1.1)	4
4-credits from Biology Primer (See §3.1.1)	4
Computer Science	
Computational thinking	
Introduction to Computer Science	4
Discrete Mathematics	4
Data Structures and Algorithms	4
Introduction to Machine Learning	4
Theory of Computation	4
Systems and software	
Computer Organisation and Systems	4
Philosophy	
PHI-1000 (Introduction to Philosophy)	4
PHI-1060 (Symbolic Logic)	4
One course from the range [PHI-x600 – PHI-x829]* = Ethics	4
One course from the range [PHI-x240 – PHI-x299] = Philosophy of Science (including Philosophy of Math) or [PHI-x300 – PHI-x359] =Philosophy of Language	4
Any 2 courses from the PHI offerings (strongly recommended: Philosophy of Mind)	8
Capstone Project	4
Total	72

1.8.3 4-year BSc Hons with Research in Computer Science and Philosophy

The 4-year BSc Hons with Research in ”Computer Science and Philosophy Interdisciplinary Major” mandates a minimum of 150 credits for completion. The credit requirements are as follows.

Total Credits 150						
Academic Credits 144					Non-academic Credits 6	
FC Credits 36	CS+Phil Credits 92			Open Credits 16	Co-curricular Credits 4	Internship Credits 2
	Core 68	Project 12	Electives 12			

Credits Breakup for the 4 Year BSc Hons Program in CS + Math

These credits are classified into two categories:

- Academic Credits (144): A minimum of 144 academic credits. These credits are divided into three categories - FC credits (36), CS+PHIL credits (92), and Open credits (16).
  - FC Credits (36): A total of 36 credits dedicated to foundational courses (https://www.ashoka.edu.in/programme/foundation-courses/)
  - CS+PHIL Credits (92): A minimum of 92 credits, divided as follows:

    - The student must complete 80 credits of CS+PHIL core courses given in §1.8.4
    - Additionally, take a minimum of 12 credits in elective courses, offered by CS and Philosophy department. ISMs offered by CS/Philosophy faculty are eligible as electives, with a maximum of 4 credits allowed through ISMs.
    - To fulfill the major requirements, students must achieve a minimum grade of "B" in both the Introduction to Computer Science and Discrete Mathematics courses.
  - Open Credits (16): The remaining 16 academic credits can be earned by taking courses from any department within the university, including the department of Computer Science and Philosophy Department.
- Non-Academic Credits (6): A minimum of 6 non-academic credits. They include
  - 4 credits designated for co-curricular courses
  - 2 credits allocated for internship experience

1.8.4 Computer Science and Philosophy Core for 4 Year BSc Hons with Research

Core Courses	Credits
Basic Science	
Probability and Statistics	4
Linear Algebra	4
Calculus (Calculus Enabler)	4
4-credits from Physics Primer (See §3.1.1)	4
4-credits from Biology Primer (See §3.1.1)	4
Computer Science	
Computational thinking	
Introduction to Computer Science	4
Discrete Mathematics	4
Data Structures and Algorithms	4
Introduction to Machine Learning	4
Theory of Computation	4
Systems and software	
Computer Organisation and Systems	4
Philosophy	
PHI-1000 (Introduction to Philosophy)	4
PHI-1060 (Symbolic Logic)	4
One course from the range [PHI-x600 – PHI-x829]* = Ethics	4
One course from the range [PHI-x240 – PHI-x299] = Philosophy of Science (including Philosophy of Math) or [PHI-x300 – PHI-x359] =Philosophy of Language	4
Any 2 courses from the PHI offerings (strongly recommended: Philosophy of Mind)	8
Capstone Thesis	12
Total	80

2 Foundational Courses

Ashoka University requires each student to take 9 Foundation Courses. Each of these courses is mandatory. These courses are not formal gateways into the Major programmes, but distinctive courses that introduce students to the foundations of thought and various styles of thinking, and also to inter- and transdisciplinary approaches.

These are Introduction to Critical Thinking, Great Books, Literature and the World, Indian Civilizations, Environmental Studies, Mind and Behaviour, Economy, Politics and Society, Mathematical Thinking and Principles of Science. Further details on these courses are available at https://www.ashoka.edu.in/programme/foundation-courses/.

3 Computer Science Curriculum

At the core of the computer science curriculum lie three foundational pillars: *Basic Science and Maths*, *Computational Thinking*, and *System and Software*. These pillars represent the essential aspects of computer science education that every aspiring CS major must master. We've designed our core courses within each of these areas to provide you with a strong and well-rounded foundation in computer science. Beyond the core courses, the department also offer a wide range of elective courses - spanning various knowledge areas of computer science education.

3.1 Computer Science Core Courses

3.1.1 Basic Science and Maths

1. Probability and Statistics
2. Linear Algebra (LA)
3. Calculus (Calculus Enabler)
4. Physics Primer
   (a) Physics of the Universe
5. Biology Primer
   (a) Introduction to Biology I: Genetics and Evolution

3.1.2 Computational thinking

1. Introduction to Computer Science (ICS)
2. Discrete Mathematics (DM)
3. Data Structures and Algorithms (DSA)
4. Design and Analysis of Algorithms (DAA)
5. Introduction to Machine Learning (IML)
6. Data Science and Management (DSM)
7. Information Security (2 credits) (InfoSec)
8. Theory of Computation (ToC)
9. Programming Languages and Translation (PLT)

3.1.3 Systems and software

1. Computer Organisation and Systems (COS)
2. Design Practices in CS (DP)
3. Computer Networks (CN)
4. Embedded Systems (ES)

3.2 Computer Science Electives

3.2.1 Algorithms and Computational Thinking

1. Formal Logic

2. Numerical Algorithms and Optimization
3. Advanced Algorithm
4. Graph Algorithms
5. Approximation Algorithms
6. Randomized Algorithms
7. Quantum algorithms

3.2.2 Machine learning and AI

1. Advanced Machine Learning
2. Natural Language Processing
3. Computer Vision
4. AI for Social Good
5. Machine learning in Economics
6. Machine learning in Healthcare
7. Computational Radiology
8. Machine learning in Natural Sciences

3.2.3 Security and Privacy

1. System Security and Privacy
2. Cryptography
3. Post-Quantum Cryptography
4. Digitalisation and Privacy

3.2.4 Systems

1. Database Management Systems
2. Operating Systems
3. Computer Architecture
4. Advanced Networks

3.2.5 Applications and software

1. Software engineering
2. Embedded systems

3.2.6 Ethics of Computing

1. Ethics of Digitalisation
2. Ethics of AI
3. Science Technology and Society

3.3 Core Courses Prerequisites in Computer Science

Course	Pre-requisites	Remarks
P&S	Check with the Mathematics Department	
Linear Algebra	Check with the Mathematics Department	
Calculus	Check with the Mathematics Department	
Physics		
Biology		
Introduction to Computer Science	Mathematics in Grades XI and XII. Alternatively, a minimum of B grade in Calculus (MAT 1000)	
Discrete Mathematics	Mathematics in Grades XI and XII. Alternatively, a minimum of B grade in Calculus (MAT 1000)	
Data Structures and Algorithms	Introduction to Computer Science; Discrete Mathematics	
Theory of Computation (ToC)	Data Structures and Algorithms	
Computer Organisation and Systems	Introduction to Computer Science	
Introduction to Machine Learning	Probability and Statistics; Linear Algebra; Data Structures and Algorithms	
Design Practices in CS	Data Structures and Algorithms; Computer Organisation and Systems	
Computer Networks	Data Structures and Algorithms	
Information Security	Data Structures and Algorithms; Probability and Statistics	
Design and Analysis of Algorithms	Data Structure and Algorithms; Linear Algebra	
Data Science and Management	Data Structures and Algorithms; Introduction to Machine Learning	
Programming Languages and Translations	Data Structures and Algorithms; Theory of Computation	

4 Core Course Syllabus

4.1 Introduction to Computer Science

- **Course objective:** To introduce students to algorithmic and computational thinking, and programming in the small.

  The course will address the following issues:

  1. Problem formulation in a precise and concise fashion and independent of language considerations.
  2. Design of simple algorithms from the problem specification – their correctness and analysis of efficiency.
  3. The intermediate steps in the design of a program from an algorithm through a process of step-wise refinement. Language dependent considerations may be used in this process, but not elsewhere.

  The emphasis throughout the course will be on the analysis required while designing correct and efficient algorithms. The course is intended to teach a student a systematic process of design - beginning with problem formulation from an informal specification, through convincing arguments to algorithms, the analysis of their correctness and efficiency, and finally arriving at programs through a process of step-wise refinement. A programming language bias will be avoided and programs will be developed in both imperative and functional styles.

- **Pre-requisite:** The course will expect familiarity with class 12 level mathematics (sets, relations, functions, basic logic and truth tables, basic counting and the principle of mathematical induction), but not necessarily with programming.

- **Coverage:** Concept of an algorithm; recursion and principle of mathematical induction; correctness of algorithms; efficiency of algorithms - time and space measures; algorithms to programs; specification, top-down development and step-wise refinement; the notion of state and finite state machine; imperative programs; correctness and loop invariants; basic algorithm design techniques; encapsulation, abstractions and modularity; basics of object-oriented programming; basic logic, soundness and completeness, example of a propositional resolution; an example of concurrency; an intro to numerical computation.

  The course will be programming language independent, and will involve programming in multiple programming languages, in both functional and imperative styles.

- **References:**

4.2 Discrete Mathematics

- **Course objective:** To lay the mathematical foundations for computational thinking for the core and elective courses including Algorithms and Data Structures, Compilers, Principle of Program Languages and Theory of Computation, and even Security to build on.

  The course:

  1. introduces students to mathematical structures, tools, techniques, and frameworks for algorithmic and computational thinking.
  2. illustrates ways for problem formulation, representation, and analysis using mathematical models
  3. elucidates how mathematical rigour and formalism ensure soundness and completeness of computation

  The emphasis throughout the course is on teaching how various concepts and results of mathematics are used in correct, efficient, and robust computations. It trains the students to think and reason about computation in a programming language-agnostic manner. It complements the course on Introduction to Computer Science from mathematical perspective and thinking.

- **Pre-requisite:** The course will expect familiarity with class 12 level mathematics (sets, relations, functions, basic logic and truth tables, basic counting and the principle of mathematical induction).
- **Coverage:** Propositional and Predicate Logic – Arguments in Logic; Proof Techniques – What is a proof, Proof Methods and Strategy; Basic Structures: Sets, Functions, Relations, Countable sets vs uncountable, Cardinality, Sequences and Matrices; Problem Solving with Recursive Decomposition – Analysis of elementary algorithms for sorting and searching, Growth of Functions, Recursion and Iteration, Induction and Recursion; Graphs and Trees; Counting – Counting Principles like pigeonhole; Recurrence Relations and basic techniques for solving them like generating functions; Modular Arithmetic; Probability - expectation, probabilistic inequlaities ; Information Theory –
- **References:**

4.3 Data Structures and Algorithms

- **Course objective:** Introduce students to a wide variety of data structures, mechanisms of analysis, implementation, along with when and how to use them. Students will:
  1. Understand and implement the wide spectrum of available data structures and how they influence algorithm design to achieve efficiency of space and time complexity..
  2. Design and analyze out-of-the-box data structures for new scenarios that regularly arise in practice and require understanding of basic data types and their internal representation models like RAM and pointers.
  3. Be able to clearly understand and distinguish between static and dynamic versions where there are trade-offs between space and time complexity.
  4. Implement and analyze the real-world performance of these data structures.
- **Pre-requisite:** Discrete Mathematics, ICS.
- **Coverage:** Basic data structures (linear lists, arrays, stacks, queues) and analysis, trees, heaps, hash-tables and related randomized constructions like universal hash functions, skiplists, self-organizing lists, graphs, strings and tries, integers, geometric data-structures like $k - d$ trees, segment trees, range trees; The significance of abstract and primitive data types; some background on word RAM vs pointer machines. Application to sorting and searching algorithms whose performance depend on the data structures taught in class as well as some elementary graph algorithms like DFS, BFS and MST. Some exposure to notions like amortization, deferred data-structuring and filtering search using suitable examples.
- **References:** Introduction to Algorithms. Cormen, Leiserson, Rivest and Stein. MIT Press

4.4 Theory of Computation

- **Course Objective:** Theory of computation deals with encapsulation and abstraction of diverse computational processes, be it hardware or software that enables us to compare them and understand their basic capabilities and limitations. It is intrinsically related to algorithmic models, where we seek to determine what can and cannot be computed, how quickly, with how much memory without being bogged down by low-level implementation details. The students will
  1. learn conceptual tools that practitioner use and argue about many fundamental computational questions in a rigorous framework.
  2. be introduced to Chomsky hierarchy of formal languages that involve understanding of core techniques like simulation, reductions and non-determinism.
  3. get a deeper understanding of what is hard to compute efficiently and trade-offs between resources like time and space even for the fastest computing machines.
- **Pre-requisites:** Discrete Mathematics, Data Structures and algorithms and Design and Analysis of Algorithms (highly recommended)

- **Coverage:** The course will cover key topics in Automata and Languages, Computability Theory, and Complexity Theory. The topics include:
  - Automata and Languages Regular Languages: Finite Automata, Non-determinism, Regular expressions, Non-regular languages, Myhill Nerode theorem and DFA Minimization
    Context Free Languages: Context free grammar and normal forms, Push down automata, Equivalence of CFL and PDA, Pumping lemma for proving non-regularity and Non-CFL, Deterministic CFL and non-CFL (optional)
  - Computability Theory
    Turning Machine and variations including Stack and counter machines Decidability: Decidable Language, Undecidability, basic Rice's theorem, Reducibility and its application to proving undecidability
  - Complexity Theory
    Time and Space Complexity and their relationship in the context of DTM and NDTM, Complexity Classes: P, NP, PSPACE, Randomized complexity classes (optional) Polynomial Reducibility NP Hardness, NP-completeness, Cook-Levin theorem for Satisfiability and some common NP complete problems
- **References:**
  - Introduction to the Theory of Computation. Michael Sipser. Cengage India Private Limited
  - Introduction to Automata Theory, Languages, and Computation. John E. Hopcroft and Rajeev Motwani and Jeffrey D. Ullman.

4.5 Computer Organisation and Systems

Basic computer organisation. Elementary concepts of an OS (multiprocessing, interprocess communication, semaphores and deadlocks, memory hierarchy).

4.6 Design Practices in CS

- **Course objective:** To get familiar with methods for doing (large?) projects in collaborative mode with special emphasis on software development.
- **Pre-requisite:** Data structures and computer organisation.
- **Coverage:**
  - Concepts: Design, analysis and modelling; software requirement analysis and detailed specifications; design documentation; standard software development tools; testing; report.
  - Tools: Including, but not limited to: Standard Unix utilities, Latex, gcc/g++ - the C preprocessor, header files, #defines; object files, linking, static and run-time libraries; symbol tables; `Makefile`, Using and creating libraries, `ar` and `ranlib`, Object oriented style, classes, namespaces etc., `git` and version control, debugging and profiling tools, Macros and in-line code, loop transformations and unrolling, speed-up issues, Documentation tools like `Doxygen`, Auto-configuration, Makefile generation and porting issues.
- **References:**

4.7 Design and Analysis of Algorithms

- **Course objective:** This course will introduce students to the mathematical modeling and solution of computational problems, and teach them how to design and analyze efficient algorithms. Students will:

1. Cover the standard algorithmic paradigms like induction, divide-and-conquer, prune-and-search, greedy, dynamic programming and a toolkit of common algorithms including randomized techniques for design, correctness proofs and rigorous analysis of time and space complexity.
2. Explore different ways of thinking about and representing problems coming from diverse domains like sets, graphs, geometry, algebra and number-theory.
3. Develop an understanding of performance metrics like worst-case, average-case and expected-case and how the underlying computational framework is intrinsically tied to the design of efficient algorithms and the notion of optimality.

- **Pre-requisite:** Data Structures, Probability and Statistics.
- **Coverage:** Techniques for the design and analysis of efficient algorithms, emphasizing methods useful in practice. Topics include divide-and-conquer; dynamic programming; greedy algorithms; branch and bound; backtracking; amortized analysis; randomized techniques and randomized data structures; graph algorithm including flows and matching. Advanced topics may include computational geometry including dimension-reduction techniques, number-theoretic algorithms, polynomial and matrix calculations, external-memory and caching, and some flavor of distributed and parallel computing. Introduction to NP-Completeness, reductions and approximation algorithms.
- **References:** Introduction to Algorithms. Cormen, Leiserson, Rivest and Stein. MIT Press

4.8 Introduction to Machine Learning

- **Course Objectives:** To understand the basic theory underlying machine learning. To be able to formulate machine learning problems corresponding to different real life applications. To understand a range of machine learning algorithms along with their strengths and weaknesses. To be able to apply machine learning algorithms, optimize models, evaluate their performance.
- **Pre-requisites:** Probability and Statistics; Linear Algebra; Data Structures and Algorithms
- **Coverage:**

  Introduction to Machine Learning; Analytics lifecycle; Overview of the different types of data: structured, unstructured, and semi-structured. Data Distributions Supervised Learning and Linear Regression; Classification and Logistic Regression; Evaluation metrics for classification models: Accuracy, Precision, Recall, F1-score, and ROC curves Classification models - Naïve Bayes; Decision Tree and Random Forest; Support Vector Machine ML and MAP estimates. Bayes' Optimal Classifier. PAC learnability and generalisation. Introduction to Graphical Models. Graph-cuts and spectral methods. Generative Vs. Discriminative Models. Unsupervised Learning; K-means clustering, Expectation Maximization, GMM Dimensionality reduction - PCA and Feature Selection Introduction to Neural Networks and Deep Learning - CNN, LSTM Reinforcement Learning
- **References:**

  1. Pattern Recognition and Machine Learning; Christopher M. Bishop

  2. Pattern Classification (2nd ed.) Richard O. Duda, Peter E. Hart and David G. Stork; 1997

  3. An Introduction to Statistical Machine Learning; Gareth James, Daniela Witten, Trebor Hastie, Robert Trebshirani, Jonathan Taylor; 2023

  4. Machine Learning; Tom Mitchel, 1997

  5. Deep Learning; Ian Goodfellow and Yoshua Bengio and Aaron Courville; MIT Press; 2016

4.9 Computer Networks

- **Course Objective:** The dynamic field of computer networking involves many concepts, protocols, and technologies that are woven together in an intricate manner. The Internet is the largest computer network that interconnects hundreds of millions of computing devices throughout the world.

This course will use the Internet as the principal vehicle for discussing computer networks and their protocols. Despite the fact that the Internet is extremely large and has numerous diverse components and uses, there exist guiding principles and structures that can provide a foundation for understanding such an amazingly large and complex system. This course will ensure that students grasp a comprehensive understanding of the guiding principles and overall structure, empowering them to navigate the intricacies of computer networking with confidence and expertise.

- **Pre-requisite:** Students should have finished the introductory CS sequence
  - Introduction to Computer Programming, and
  - Computer Organization and Systems.
- **Coverage:**
  - **Network architecture, layering, and protocols.**
  - **Physical Layer:** Overview of the Physical Layer's role and its primary responsibilities, which involve transmitting raw bit streams over physical media. Data encoding, Transmission modes, Physical layer devices.
  - **The Link Layer and Local Area Networks:**
    - Link Layer Introduction and Services
    - Encoding, Synchronization, and Framing
    - Error Detection: parity checks, checksum methods, cyclic redundancy check (CRC)
    - Reliable Transmission: Stop-and-wait Protocols. Sliding-window Protocols.
    - Multiple Access Protocols: Random Access Protocols (ALOHA)
    - Link Layer Addressing: MAC addresses
    - Local Area Networks (LANs)
    - Ethernet. Topology, Frame structure, and CSMA/CD MAC protocol.
    - LAN Extension: Link layer switches, packet switching.
  - **Network Layer: Internetworking:**
    - The Internet Protocol (IP).
    - The IP service model.
    - Routers, Subnets, Domain.
    - IP Addressing.
    - ARP Protocol
    - Dynamic Host configuration Protocol (DHCP)
    - Routing Algorithms.
    - Intra-domain routing: Link-state routing, Distance-vector routing, RIP, OSPF.
    - Inter-domain routing: BGP.
  - **Transport and Application Layers Protocols:**
    - Transport layer service model.
    - Transport layer protocols: Connectionless transport (UDP), Connection-oriented transport (TCP)
    - Network Address Translation (NAT)
    - Application-level protocols
    - Web and HTTP
    - Domain name service (DNS)
    - Socket Programming
- **References:**
  - Computer Networking: A Top-Down Approach. James F. Kurose and Keith W. Ross. Pearson Publication.
  - Computer Networks: A Systems Approach. Larry L. Peterson and Bruce S. Davie. Morgan Kaufmann Publishers.

4.10 Data Science and Management

- **Course Objectives:** The main objective of the course is to understand the life cycle of data, starting from collecting the data to storing and analyzing it and to present the results of the analysis in an intuitive and visually appealing way.
- **Pre-requisites:** Data Structures, Machine Learning
- **Coverage:** The broad topics that will be covered in this course include data collection from online and offline sources, data cleaning problems and methods, data processing and storage for relational, non-relational, unstructured and graph-structured data, data analysis with various tools, including statistics, machine learning models and SQL, and visualization.
- **References:**

4.11 Information Security

- **Course Objectives:** This course will help students understand and model information security in the modern world. It covers various definitional basics of cryptography, models of trust, adversaries, etc., helping students reason concretely about real-world problems.
- **Pre-requisites:** Data Structures and Algorithms, Probability and Statistics
- **Coverage:** Introduction to security. Trust vs verifiability. Adversarial threat models. Notions of secrecy, privacy, and security. Basics of cryptography - symmetric and public key encryption, authentication, hash functions, digital signatures, certificates, cryptographic protocols, and applications. Cryptographic security definitions. Models of authentication and authorisation; biometrics; identity. Issues with verification of hardware and software integrity; the frameworks of formal verification and model checking. Trust assumptions, distributing trust with secure multiparty computations, hardware trust models, trusted computing environments and remote attestation. Elements of OS and Network security.
- **References:**

4.12 Programming Languages and Translation

- **Course Objective:** Starting 1950's (FORTAN & LISP) innumerable programming languages have been designed, implemented, deployed, and re-engineered. Some are targeted for specific domains while many serve to be 'general purpose' and cross-domain. Yet all of them have one thing in common – the ability to express the computational needs for some real world problem, to act as a bridge between the man and the machine. In this evolution, several paradigms of computing emerged — functional, procedural, object-oriented, generic (meta programming), logic — to name the major few. And languages have been functional (LISP), procedural (C), logic (Prolog), or simply multi-paradigm (C++).

  Programming Languages are living entities of computation. They are born (designed and implemented), grow in their use, and fade out to others. Hence, long serving languages need regular re-births and / or re-purposing (C++98 $\rightarrow$ C++11 $\rightarrow$ C++14 $\rightarrow \cdots$). Understandably lot of languages (actually most) die when they stop serving their computing role. And a few are resurrected (like LISP), when we rediscover their worth after decades.

  As languages take their journey, we need to take care of two major aspects for every language – how does the semantics of the language work for the 'man' and how is it efficiently translated to the 'machine' for the ever-evolving processor paradigms and architectures.

  This course on language translation attempts to address these two aspects through a few illustrative languages. It is practice oriented. So you will need to write a complete compiler for a tiny language.
- **Pre-requisites:**
  1. Programming in C / C++ / Python

2. Data Structures
3. Algorithms

- **Coverage:**

1. **Introduction**: Why study Programming Languages?; Overview of Programming Paradigms and Languages: Imperative (Algorithms + Data): Fortran, Pascal, Algol, C, Cobol, Ada, Perl, Python, Object Oriented (Data + Model): C++, Java, C#, SmallTalk, Eiffel, Logic (Facts + Rules + Queries): Prolog, Functional (Functions): Haskell, Scheme, Lisp, ML; Commonality and Contrast of Semantics
2. **Framework for Language Translation**: Phases of a Compiler: Overview of Compilation Process, Compiler Front-end, Compiler Back-end; Sample Translation
3. **Lexical Analysis**: Fundamentals; Theory of Lexical Analysis: RE → NFA → DFA
4. **Flex: Lexical Analyzer Generator**: Flex (Fast Lexical Analyzer) Specification; Interactive Flex and Flex-Bison Flow
5. **Syntax Analysis**: Fundamentals: Grammar Rules, Derivations and Parsing, Parse Tree, Abstract Syntax Tree (AST); Top-Down Parsers: Recursive Descent / LL Parsers, Left-Recursion, Ambiguous Grammar; Bottom-Up Parsers: SR, LR(0), SLR(1), LR(1), LALR(1), LR(k) Parsers, Ambiguous Grammar
6. **Bison: Syntax Analyzer / Parser Generator**: Bison Specification; Example Specifications and Handling Ambiguous Grammars
7. **Machine Independent Translation**: Intermediate Representations: AST, DAG, SSA, RTL, CDFG, Three Address Code; Symbol Table: Scope, Interface, Implementation; Translation of C Features: Arithmetic Expressions, Boolean Expressions, Control Constructs, Types and Declarations, Type in Translation, Arrays, Type Expressions, Functions, Scopes, Structures, C Pre-Processor (CPP) Directives; Translation of C++ Features (Optional): Namespace, Class, Inheritance, Templates, Exceptions
8. **Run-time Environments**: Memory Organization and Binding Protocol: Symbol and Address; Symbol Table; Activation Record (AR) / Stack Frame; Function Call Protocol; Optimization and I/O; Types at Run Time: `int`, `double`, Pointer, `struct`, Array, Function Pointer, Blocks, Global, Static
9. **Target Code Generation**: Converting Three Address Code to Target Code: Three Address Code Optimization, Memory Binding, Register Allocation and Assignment, Code Translation, Target Code Optimization; Three Address Code to Assembly: Simple Code Mapping by Table Lookup
10. **Control Flow Graph (CFG) and Local Optimization**: Optimization Issues; Basic Block and Control Flow Graph; Optimizing Basic Blocks; Extended Basic Blocks
11. **Global Register Allocation (GRA)**: Issues and Problem Formulation; Simple GRA by Usage Count; Chaitin's Algorithm: GRA by Graph Coloring
12. **Simple Code Generators**: Issues in Code Generation; Simple and Optimal Code Generation Algorithms; Peephole Optimizations

- **References:**

1. Compilers: Principles, Techniques, and Tools by Alfred V. Aho, Monica S Lam, Ravi Sethi, Jeffrey D. Ullman, 2nd Edition, Pearson / Addison-Wesley, 2006
2. Concepts of Programming Languages by Robert W. Sebesta, 11th Edition.
3. Flex and Bison by John Levine, O'Reilly, 2009

4.13 Cryptography

- **Course Objective:** This course serves as an introduction to modern cryptography, emphasizing its classical goals: data privacy, authenticity, and integrity. It covers the fundamentals of both symmetric-key and public-key cryptography. Additionally, the course introduces advanced cryptography concepts, including commitment schemes, secret sharing, oblivious transfer, zero-knowledge proofs, and multi-party computation. We will implement practically everything we learn.

  Privacy, secrecy, and security are central to our emerging "information society", and cryptography is a key technology for achieving them; it is also a fascinating field of study in its own right. Cryptography lies at the center of this course: on the one end, we'll look at problems of computer and information security, and then understand and implement cryptographic tools to solve them. We'll also touch on some social issues surrounding the use of cryptography. At the other end, we'll explore the mathematical structures from which cryptographic primitives are built, and learn how to use some of these techniques in real-world scenarios.

- **Pre-requisite:** Discrete Mathematics, Probability and Statistics, Introduction to Computer Science

- **Coverage:**

  - Symmetric key cryptography: Symmetric Key Encryption, Pseudorandom number generators (PRG), Pseudorandom functions (PRF), Stream ciphers, Block ciphers, Modes of Operations, Security definitions for Encryption, Message integrity, Message authentication codes, Hash Functions.
  - Public key cryptography: Diffie-Hellman Key Exchange, Discrete Logarithm Problem. Public key encryption, Security definitions, Factoring Problem, ElGamal Encryption, RSA Encryption. Digital Signatures, RSA Signature, DSA Signature. Introduction to Elliptic Curve, ECDSA Signature.
  - Cryptography Applications: Public key infrastructure (PKI), Brief Introduction to TLS.
  - Advanced Cryptography Concepts: Commitment Schemes, Secret Sharing, Oblivious Transfer, Zero knowledge Proofs, Multiparty Computation.

- **References:**

  - Introduction to Modern Cryptography. Katz, Jonathan and Lindell, Yehuda. CRC Press.
  - A Graduate Course in Applied Cryptography. Dan Boneh and Victor Shoup.
  - Cryptography: Theory and Practice. Douglas R. Stinson and Maura B. Paterson. CRC Press.
  - Goldwasser and Bellare's notes (`https://cseweb.ucsd.edu/~mihir/papers/gb.pdf`).