Master's Degree

Computing and Programming Languages

Accreditation/Membership

Association
for Computing
Machinery

tech global university

tech global university

Master's Degree
Computing and Programming Languages

- Modality: **online**
- Duration: **12 months.**
- Certificate: **TECH Global University**
- Accreditation: **60 ECTS**
- Schedule: **at your own pace**
- Exams: **online**

Website www.techtitute.com/us/information-technology/master-degree/master-degree-computing-programming-languages

Index

01

Introduction to the Program

The constant evolution of artificial intelligence, machine learning, and language processing has transformed the global computing landscape. These advances require professionals capable of understanding both the theoretical foundations and the practical applications of computational languages. According to UNESCO, 90% of jobs in the near future will require advanced digital competencies. This reality has increased the need for specialized programs that integrate knowledge of algorithms, data structures, and language semantics. In this context, TECH presents an innovative, fully online qualification in Computing and Programming Languages, designed to provide solid and up-to-date academic preparation within a flexible digital environment.

A comprehensive and 100% online program, exclusive to TECH, with an international perspective supported by our membership with the Association for Computing Machinery"

In an increasingly technological environment shaped by automation, artificial intelligence, and natural language processing, understanding how computational languages are constructed, interpreted, and optimized has become a key requirement for progress across multiple fields of knowledge and industry. Human–machine interaction, the efficiency of computing systems, and the creation of new digital tools depend largely on mastery of these languages.

This program offers a comprehensive view of the pillars that underpin computing and programming languages, enabling the acquisition of specialized competencies that enhance logical thinking, complex problem-solving, and the design of innovative solutions. Topics such as formal semantics, language theory, automata, compilation, and other fundamental elements essential to understanding the internal functioning of computing systems are addressed in depth.

Moreover, mastery of these contents opens opportunities in highly dynamic sectors such as artificial intelligence, programming language development, quantum computing, and software engineering. This specialization represents a strategic step for those seeking to advance their careers in academic, technical, or professional settings, particularly in contexts where profiles with a strong theoretical foundation and analytical capacity are highly valued. The conceptual depth acquired enables not only the application of existing technologies, but also the design of those of the future.

The online format of this program allows access to up-to-date, high-level academic content from anywhere, without compromising quality or rigor. This flexibility facilitates compatibility with other professional or academic projects and promotes autonomous, rigorous learning oriented toward critical analysis. Through an interactive platform, multimedia materials, and expert guidance, an enriching educational experience is ensured that responds to the current needs of the digital and academic sectors. In addition, a renowned International Guest Director will deliver 10 in-depth Masterclasses.

Furthermore, thanks to TECH's membership in the **Association for Computing Machinery (ACM)**, students will have access to exclusive and up-to-date resources, such as scientific publications, specialized courses, and international conferences. Additionally, they will have the opportunity to expand their network by connecting with experts in technology, artificial intelligence, data science, and other key disciplines in the sector.

This **Master's Degree in Computing and Programming Languages** contains the most complete and up-to-date university program on the market. Its most notable features are:

- The development of practical cases presented by experts in Computing and Programming Languages
- The graphic, schematic, and practical contents with which they are created, provide scientific and practical information on the disciplines that are essential for professional practice
- Practical exercises where the self-assessment process can be carried out to improve learning
- Its special emphasis on innovative methodologies
- Theoretical lessons, questions to the expert, debate forums on controversial topics, and individual reflection assignments
- Content that is accessible from any fixed or portable device with an Internet connection

A renowned International Guest Director will deliver 10 exclusive Masterclasses on the latest trends in Computing and Programming Languages"

You will delve into automata theory and formal languages to understand the internal functioning of computational systems"

You will work with advanced data structures to develop efficient, scalable, and performance-oriented solutions.

You will design sophisticated algorithms to solve computational problems with high demand in key technology sectors.

The faculty includes professionals from the field of Computing, who contribute their practical experience to the program, as well as renowned specialists from leading professional associations and prestigious universities.

Its multimedia content, developed using the latest educational technology, enables situated and contextualized learning—that is, a simulated environment that provides an immersive learning experience designed to prepare professionals for real-world situations.

This program is designed around Problem-Based Learning, whereby the student must try to solve the different professional practice situations that arise throughout the program. For this purpose, the professional will be assisted by an innovative interactive video system created by renowned and experienced experts.

02

Why Study at TECH?

TECH is the world's largest online university. With an impressive catalog of more than 14,000 university programs, available in 11 languages, it is positioned as a leader in employability, with a 99% job placement rate. In addition, it has a huge faculty of more than 6,000 professors of the highest international prestige.

Study at the largest online university in the world and ensure your professional success. The future begins at TECH"

The world's best online university, according to FORBES

The prestigious Forbes magazine, specialized in business and finance, has highlighted TECH as "the best online university in the world" This is what they have recently stated in an article in their digital edition in which they echo the success story of this institution, "thanks to the academic offer it provides, the selection of its teaching staff, and an innovative learning method oriented to form the professionals of the future".

The best top international faculty

TECH's faculty is made up of more than 6,000 professors of the highest international prestige. Professors, researchers and top executives of multinational companies, including Isaiah Covington, performance coach of the Boston Celtics; Magda Romanska, principal investigator at Harvard MetaLAB; Ignacio Wistuba, chairman of the department of translational molecular pathology at MD Anderson Cancer Center; and D.W. Pine, creative director of TIME magazine, among others.

The world's largest online university

TECH is the world's largest online university. We are the largest educational institution, with the best and widest digital educational catalog, one hundred percent online and covering most areas of knowledge. We offer the largest selection of our own degrees and accredited online undergraduate and postgraduate degrees. In total, more than 14,000 university programs, in ten different languages, making us the largest educational institution in the world.

The most complete syllabuses on the university scene

TECH offers the most complete syllabuses on the university scene, with programs that cover fundamental concepts and, at the same time, the main scientific advances in their specific scientific areas. In addition, these programs are continuously updated to guarantee students the academic vanguard and the most demanded professional skills. and the most in-demand professional competencies. In this way, the university's qualifications provide its graduates with a significant advantage to propel their careers to success.

A unique learning method

TECH is the first university to use Relearning in all its programs. This is the best online learning methodology, accredited with international teaching quality certifications, provided by prestigious educational agencies. In addition, this innovative academic model is complemented by the "Case Method", thereby configuring a unique online teaching strategy. Innovative teaching resources are also implemented, including detailed videos, infographics and interactive summaries.

The official online university of the NBA

TECH is the official online university of the NBA. Thanks to our agreement with the biggest league in basketball, we offer our students exclusive university programs, as well as a wide variety of educational resources focused on the business of the league and other areas of the sports industry. Each program is made up of a uniquely designed syllabus and features exceptional guest hosts: professionals with a distinguished sports background who will offer their expertise on the most relevant topics.

Leaders in employability

TECH has become the leading university in employability. Ninety-nine percent of its students obtain jobs in the academic field they have studied within one year of completing any of the university's programs. A similar number achieve immediate career enhancement. All this thanks to a study methodology that bases its effectiveness on the acquisition of practical skills, which are absolutely necessary for professional development.

Google Premier Partner

The American technology giant has awarded TECH the Google Premier Partner badge. This award, which is only available to 3% of the world's companies, highlights the efficient, flexible and tailored experience that this university provides to students. The recognition not only accredits the maximum rigor, performance and investment in TECH's digital infrastructures, but also places this university as one of the world's leading technology companies.

The top-rated university by its students

Students have positioned TECH as the world's top-rated university on the main review websites, with a highest rating of 4.9 out of 5, obtained from more than 1,000 reviews. These results consolidate TECH as the benchmark university institution at an international level, reflecting the excellence and positive impact of its educational model.

03

Syllabus

In a context where advances in programming languages, automata theory, and formal semantics are driving the development of new technologies, an in-depth study of these foundations is essential for innovation in areas such as quantum computing, artificial intelligence, and cybersecurity. For this reason, this syllabus integrates up-to-date and strategically structured content that enables an understanding of the internal logic of computing systems. In this way, it fosters a critical and advanced perspective on computational functioning, aligned with the digital transformations that are redefining computer science standards on a global scale.

You will design and implement domain-specific languages tailored to specific technical needs"

Module 1. Programming Fundamentals

1.1. Introduction to Programming
- 1.1.1. Basic Structure of a Computer
- 1.1.2. Software
- 1.1.3. Programming Languages
- 1.1.4. Life Cycle of a Software Application

1.2. Algorithm Design
- 1.2.1. Problem Solving
- 1.2.2. Descriptive Techniques
- 1.2.3. Algorithm Elements and Structure

1.3. Elements of a Program
- 1.3.1. C++ Origin and Features
- 1.3.2. Development Environment
- 1.3.3. Concept of Program
- 1.3.4. Types of Fundamental Data
- 1.3.5. Operators
- 1.3.6. Expressions
- 1.3.7. Statements
- 1.3.8. Data Input and Output

1.4. Control Sentences
- 1.4.1. Statements
- 1.4.2. Branches
- 1.4.3. Loops

1.5. Abstraction and Modularity: Functions
- 1.5.1. Modular Design
- 1.5.2. Concept of Function and Utility
- 1.5.3. Definition of a Function
- 1.5.4. Execution Flow in a Function Call
- 1.5.5. Function Prototypes
- 1.5.6. Results Return
- 1.5.7. Calling a Function: Parameters
- 1.5.8. Passing Parameters by Reference and by Value
- 1.5.9. Scope of an Identifier

1.6. Static Data Structures
- 1.6.1. *Arrays*
- 1.6.2. Matrices. Polyhedra
- 1.6.3. Searching and Sorting
- 1.6.4. Strings. Input/Output Functions for Strings
- 1.6.5. Structures. Unions
- 1.6.6. New Types of Data

1.7. Dynamic Data Structures: Pointers
- 1.7.1. Concept. Definition of Pointer
- 1.7.2. Pointer Operators and Operations
- 1.7.3. Arrays of Pointers
- 1.7.4. Pointers and Arrays
- 1.7.5. Pointers to Strings
- 1.7.6. Pointers to Structures
- 1.7.7. Multiple Indirection
- 1.7.8. Pointers to Functions
- 1.7.9. Passing Functions, Structures, and Arrays as Function Parameters

1.8. Files
- 1.8.1. Basic Concepts
- 1.8.2. File Operations
- 1.8.3. Types of Files
- 1.8.4. File Organization
- 1.8.5. Introduction to C++ Files
- 1.8.6. Managing Files

1.9. Recursion
- 1.9.1. Definition of Recursion
- 1.9.2. Types of Recursion
- 1.9.3. Advantages and Disadvantages
- 1.9.4. Considerations
- 1.9.5. Recursive to Iterative Conversion
- 1.9.6. Recursion Stack

1.10. Testing and Documentation
- 1.10.1. Program Testing
- 1.10.2. White Box Testing
- 1.10.3. Black Box Testing
- 1.10.4. Testing Tools
- 1.10.5. Program Documentation

Module 2. Data Structures

2.1. Introduction to C++ Programming
- 2.1.1. Classes, Constructors, Methods and Attributes
- 2.1.2. Variables
- 2.1.3. Conditional Expressions and Loops
- 2.1.4. Objects

2.2. Abstract Data Types (ADT)
- 2.2.1. Types of Data
- 2.2.2. Basic Structures and ADTs
- 2.2.3. Vectors and Arrays

2.3. Linear Data Structures
- 2.3.1. List ADT: Definition
- 2.3.2. Linked and Doubly Linked Lists
- 2.3.3. Sorted Lists
- 2.3.4. Lists in C++
- 2.3.5. ADT Stack
- 2.3.6. ADT Queue
- 2.3.7. Stack and Queue in C++

2.4. Hierarchical Data Structures
- 2.4.1. ADT Tree
- 2.4.2. Paths
- 2.4.3. N-Ary Trees
- 2.4.4. Binary Trees
- 2.4.5. Binary Search Trees

2.5. Hierarchical Data Structures: Complex Trees
- 2.5.1. Perfectly Balanced or Minimum Height Trees
- 2.5.2. Multipath Trees
- 2.5.3. Bibliographic References

2.6. Priority Mounds and Queue
- 2.6.1. ADT Heaps
- 2.6.2. ADT Priority Queues

2.7. Hash Tables
- 2.7.1. ADT Hash Table
- 2.7.2. Hash Functions
- 2.7.3. Hash Function in Hash Tables
- 2.7.4. Rehashing
- 2.7.5. Open Hash Tables

2.8. Graphs
- 2.8.1. ADT Graph
- 2.8.2. Types of Graphs
- 2.8.3. Graphical Representation and Basic Operations
- 2.8.4. Graph Design

2.9. Algorithms and Advanced Graph Concepts
- 2.9.1. Problems about Graphs
- 2.9.2. Path Algorithms
- 2.9.3. Search or Path Algorithms
- 2.9.4. Other Algorithms

2.10. Other Data Structures
- 2.10.1. Sets
- 2.10.2. Parallel Arrays
- 2.10.3. Symbol Tables
- 2.10.4. *Tries*

Module 3. Algorithmics and Complexity

3.1. Introduction to Algorithm Design Strategies
- 3.1.1. Recursion
- 3.1.2. Divide and Conquer
- 3.1.3. Other Strategies

3.2. Efficiency and Analysis of Algorithms
- 3.2.1. Efficiency Measures
- 3.2.2. Measuring the Size of the Input
- 3.2.3. Measuring Execution Time
- 3.2.4. Worst, Best and Average Case
- 3.2.5. Asymptotic Notation
- 3.2.6. Criteria for Mathematical Analysis of Non-Recursive Algorithms
- 3.2.7. Mathematical Analysis of Recursive Algorithms
- 3.2.8. Empirical Analysis of Algorithms

3.3. Sorting Algorithms
- 3.3.1. Concept of Sorting
- 3.3.2. Bubble Sorting
- 3.3.3. Sorting by Selection
- 3.3.4. Sorting by Insertion
- 3.3.5. Merge Sort
- 3.3.6. Quick Sort

3.4. Algorithms with Trees
- 3.4.1. Tree Concept
- 3.4.2. Binary Trees
- 3.4.3. Tree Paths
- 3.4.4. Representing Expressions
- 3.4.5. Ordered Binary Trees
- 3.4.6. Balanced Binary Trees

3.5. Algorithms Using Heaps
- 3.5.1. Heaps
- 3.5.2. Heap Sort
- 3.5.3. Priority Queues

3.6. Graph Algorithms
- 3.6.1. Representation
- 3.6.2. Traversal in Width
- 3.6.3. Depth Travel
- 3.6.4. Topological Sorting

3.7. Greedy Algorithms
- 3.7.1. Greedy Strategy
- 3.7.2. Greedy Strategy Elements
- 3.7.3. Currency Exchange
- 3.7.4. Traveler's Problem
- 3.7.5. Backpack Problem

3.8. Minimal Path Finding
- 3.8.1. The Minimum Path Problem
- 3.8.2. Negative Arcs and Cycles
- 3.8.3. Dijkstra's Algorithm

3.9. Greedy Algorithms on Graphs
- 3.9.1. Minimum Spanning Tree
- 3.9.2. Prim's Algorithm
- 3.9.3. Kruskal's Algorithm
- 3.9.4. Complexity Analysis

3.10. *Backtracking*
- 3.10.1. Backtracking Algorithm
- 3.10.2. Alternative Techniques

Module 4. Advanced Algorithm Design

4.1. Analysis of Recursive and Divide and Conquer Algorithms
- 4.1.1. Posing and Solving Homogeneous and Non-Homogeneous Recurrence Equations
- 4.1.2. General Description of the Divide and Conquer Strategy

4.2. Amortized Analysis
- 4.2.1. Aggregate Analysis
- 4.2.2. The Accounting Method
- 4.2.3. The Potential Method

4.3. Dynamic Programming and Algorithms for NP Problems
  4.3.1. Characteristics of Dynamic Programming
  4.3.2. Backtracking
  4.3.3. Branch and Bound
4.4. Combinatorial Optimization
  4.4.1. Representation
  4.4.2. 1D Optimization
4.5. Randomization Algorithms
  4.5.1. Examples of Randomization Algorithms
  4.5.2. Buffon's Theorem
  4.5.3. Monte Carlo Algorithm
  4.5.4. Las Vegas Algorithm
4.6. Local Search and Candidate Search
  4.6.1. *Gradient Ascent*
  4.6.2. *Hill Climbing*
  4.6.3. *Simulated Annealing*
  4.6.4. *Tabu Search*
  4.6.5. Candidate Search
4.7. Formal Verification of Programs
  4.7.1. Specification of Functional Abstractions
  4.7.2. The Language of First-Order Logic
  4.7.3. Hoare's Formal System
4.8. Verification of Iterative Programs
  4.8.1. Rules of Hoare's Formal System
  4.8.2. Concept of Invariant Iterations
4.9. Numeric Methods
  4.9.1. The Bisection Method
  4.9.2. The Newton-Raphson Method
  4.9.3. The Secant Method
4.10. Parallel Algorithms
  4.10.1. Parallel Binary Operations
  4.10.2. Parallel Operations with Networks
  4.10.3. Parallelism in Divide and Conquer
  4.10.4. Parallelism in Dynamic Programming

Module 5. Advanced Programming

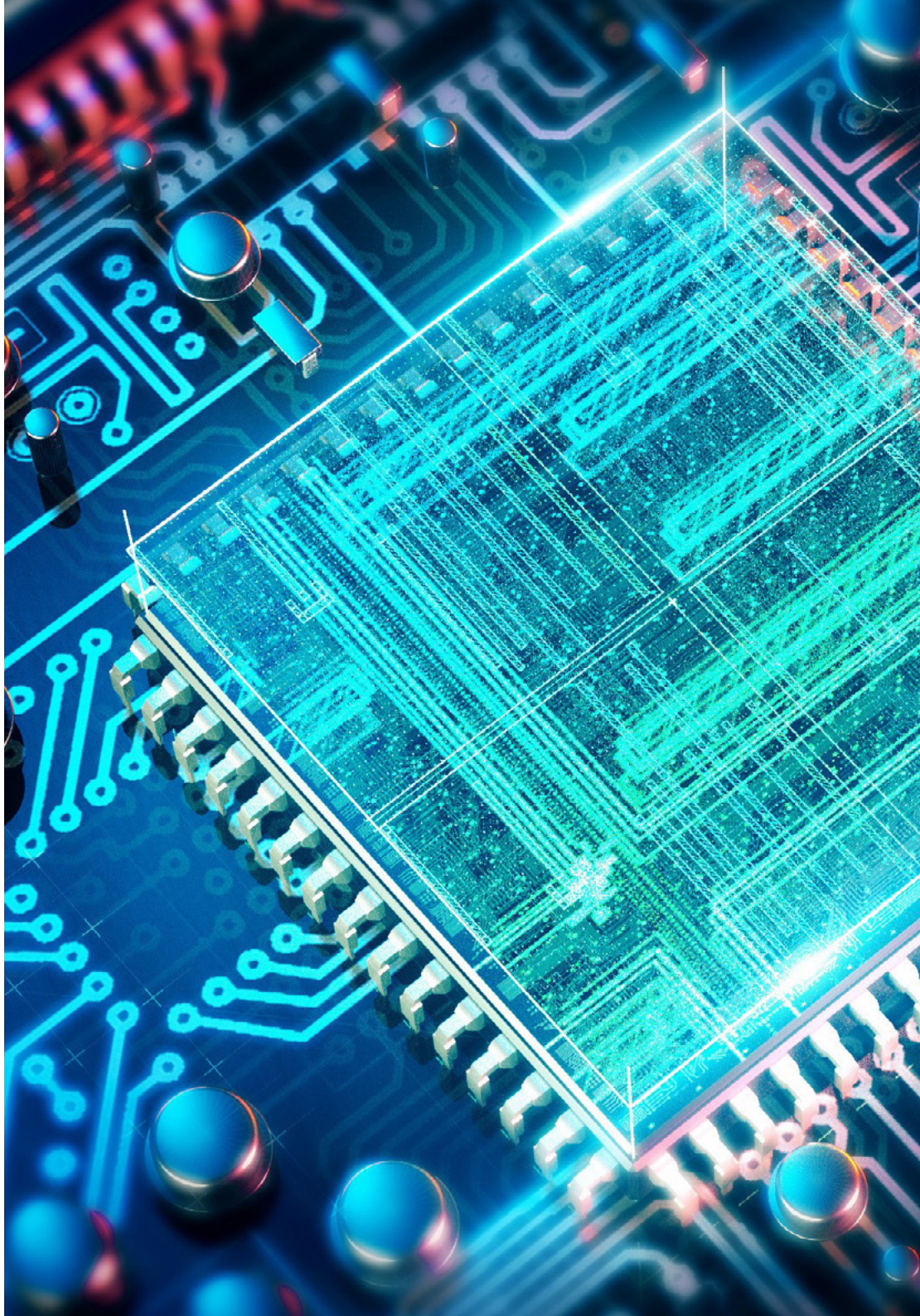
5.1. Introduction to Object-Oriented Programming
  5.1.1. Introduction to Object-Oriented Programming
  5.1.2. Class Design
  5.1.3. Introduction to UML for Problem Modeling
5.2. Relationships Between Classes
  5.2.1. Abstraction and Inheritance
  5.2.2. Advanced Inheritance Concepts
  5.2.3. Polymorphism
  5.2.4. Composition and Aggregation
5.3. Introduction to Design Patterns for Object-Oriented Problems
  5.3.1. What Are Design Patterns?
  5.3.2. Factory Pattern
  5.3.3. Singleton Pattern
  5.3.4. Observer Pattern
  5.3.5. Composite Pattern
5.4. Exceptions
  5.4.1. What Are Exceptions?
  5.4.2. Exception Catching and Handling
  5.4.3. Throwing Exceptions
  5.4.4. Exception Creation
5.5. User Interfaces
  5.5.1. Introduction to Qt
  5.5.2. Positioning
  5.5.3. What Are Events?
  5.5.4. Events: Definition and Catching
  5.5.5. User Interface Development
5.6. Introduction to Concurrent Programming
  5.6.1. Introduction to Concurrent Programming
  5.6.2. The Concept of Process and Thread
  5.6.3. Interaction Between Processes or Threads
  5.6.4. Threads in C++
  5.6.5. Advantages and Disadvantages of Concurrent Programming

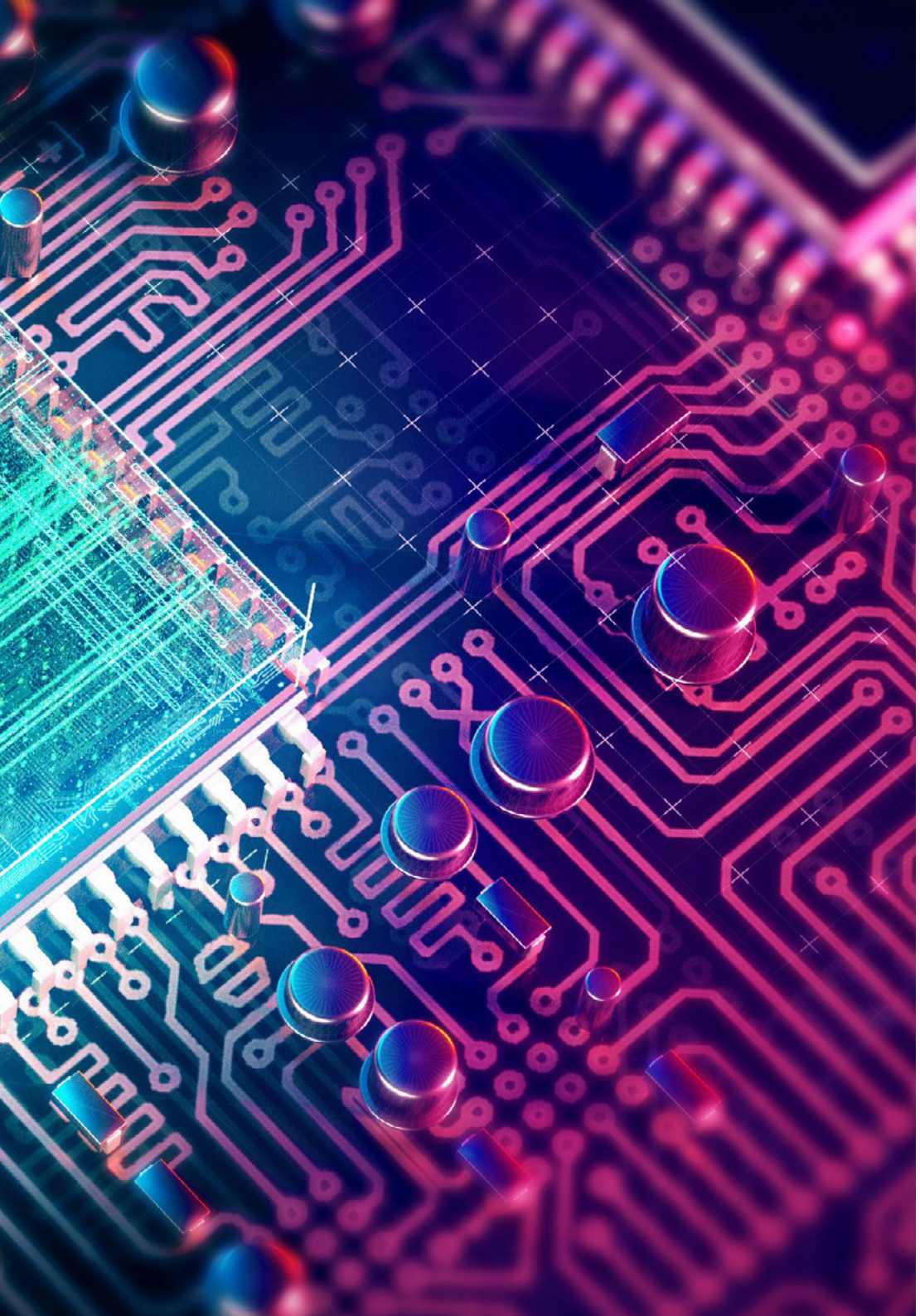
5.7. Thread Management and Synchronization
- 5.7.1. Life Cycle of a Thread
- 5.7.2. Thread Class
- 5.7.3. Thread Planning
- 5.7.4. Thread Groups
- 5.7.5. Daemon Threads
- 5.7.6. Synchronization
- 5.7.7. Locking Mechanisms
- 5.7.8. Communication Mechanisms
- 5.7.9. Monitors

5.8. Common Problems in Concurrent Programming
- 5.8.1. The Problem of Consuming Producers
- 5.8.2. The Problem of Readers and Writers
- 5.8.3. The Problem of the Philosophers' Dinner Party

5.9. Software Documentation and Testing
- 5.9.1. Why is it Important to Document Software?
- 5.9.2. Design Documentation
- 5.9.3. Documentation Tool Use

5.10. Software Testing
- 5.10.1. Introduction to Software Testing
- 5.10.2. Types of Tests
- 5.10.3. Unit Test
- 5.10.4. Integration Test
- 5.10.5. Validation Test
- 5.10.6. System Test





Module 6. Theoretical Computer Science

6.1. Mathematical Concepts Used
- 6.1.1. Introduction to Propositional Logic
- 6.1.2. Theory of Relations
- 6.1.3. Numerable and Non-Numerable Sets

6.2. Formal Languages and Grammars and Introduction to Turing Machines
- 6.2.1. Formal Languages and Grammars
- 6.2.2. Decision Problem
- 6.2.3. The Turing Machine

6.3. Extensions to Turing Machines, Constrained Turing Machines and Computers
- 6.3.1. Programming Techniques for Turing Machines
- 6.3.2. Extensions for Turing Machines
- 6.3.3. Restricted Turing Machines
- 6.3.4. Turing Machines and Computers

6.4. Indecibility
- 6.4.1. Non-Recursively Enumerable Language
- 6.4.2. A Recursively Enumerable Undecidable Problem

6.5. Other Undecidable Problems
- 6.5.1. Undecidable Problems for Turing Machines
- 6.5.2. Post Correspondence Problem (PCP)

6.6. Intractable Problems
- 6.6.1. The Classes P and NP
- 6.6.2. A NP-Complete Problem
- 6.6.3. Restricted Satisfiability Problem
- 6.6.4. Other NP-Complete Problems

6.7. *Co-NP and PSPACE* Problems
- 6.7.1. Complementary to NP Languages
- 6.7.2. Problems Solvable in Polynomial Space
- 6.7.3. Complete *PSPACE* Problems

6.8. Classes of Randomization-Based Languages
- 6.8.1. MT Model with Randomization
- 6.8.2. RP and ZPP Classes
- 6.8.3. Primality Test
- 6.8.4. Complexity of the Primality Test

6.9. Other Classes and Grammars
- 6.9.1. Probabilistic Finite Automata
- 6.9.2. Cellular Automata
- 6.9.3. McCulloch and Pitts Cells
- 6.9.4. Lindenmayer Grammars

6.10. Advanced Computing Systems
- 6.10.1. Membrane Computing: P-Systems
- 6.10.2. DNA Computing
- 6.10.3. Quantum Computing

Module 7. Automata Theory and Formal Languages

7.1. Introduction to Automata Theory
- 7.1.1. Why Study Automata Theory?
- 7.1.2. Introduction to Formal Demonstrations
- 7.1.3. Other Forms of Demonstration
- 7.1.4. Mathematical Induction
- 7.1.5. Alphabets, Strings and Languages

7.2. Deterministic Finite Automata
- 7.2.1. Introduction to Finite Automata
- 7.2.2. Deterministic Finite Automata

7.3. Non-Deterministic Finite Automata
- 7.3.1. Non-Deterministic Finite Automata
- 7.3.2. Equivalence Between DFA and NFA
- 7.3.3. Finite Automata with ε-Transitions

7.4. Languages and Regular Expressions (I)
- 7.4.1. Languages and Regular Expressions
- 7.4.2. Finite Automata and Regular Expressions

7.5. Languages and Regular Expressions (II)
- 7.5.1. Conversion of Regular Expressions into Automata
- 7.5.2. Applications of Regular Expressions
- 7.5.3. Algebra of Regular Expressions

7.6. Pumping and Closure Lemma of Regular Languages
- 7.6.1. Pumping Lemma
- 7.6.2. Closure Properties of Regular Languages

7.7. Equivalence and Minimization of Automata
- 7.7.1. Equivalence of Finite Automata
- 7.7.2. Minimization of Finite Automata

7.8. Context-Free Grammars (CFGs)
- 7.8.1. Context-Free Grammars
- 7.8.2. Derivation Trees
- 7.8.3. Applications of CFGs
- 7.8.4. Ambiguity in Grammars and Languages

7.9. Pushdown Automata and CFGs
- 7.9.1. Definition of Pushdown Automata
- 7.9.2. Languages Accepted by a Pushdown Automaton
- 7.9.3. Equivalence Between Pushdown Automata and CFGs
- 7.9.4. Deterministic Pushdown Automaton

7.10. Normal Forms, the Pumping Lemma for CFGs, and Properties of Context-Free Languages
- 7.10.1. Normal Forms for CFGs
- 7.10.2. Pumping Lemma
- 7.10.3. Closure Properties of Languages
- 7.10.4. Decision Properties of Context-Free Languages

Module 8. Language Processors

8.1. Introduction to the Compilation Process
- 8.1.1. Compilation and Interpretation
- 8.1.2. Compiler Execution Environment
- 8.1.3. Analysis Process
- 8.1.4. Synthesis Process

8.2. Lexical Analyzer
- 8.2.1. What Is a Lexical Analyzer?
- 8.2.2. Implementation of the Lexical Analyzer
- 8.2.3. Semantic Actions
- 8.2.4. Error Recovery
- 8.2.5. Implementation Issues

8.3. Syntactic Analysis
- 8.3.1. What Is a Parser?
- 8.3.2. Previous Concepts
- 8.3.3. Top-Down Analyzers
- 8.3.4. Bottom-Up Analyzers

8.4. Top-Down Parsing and Bottom-Up Parsing
- 8.4.1. LL(1) Parser
- 8.4.2. LR(0) Parser
- 8.4.3. Parser Example

8.5. Advanced Bottom-Up Parsing
- 8.5.1. SLR Parser
- 8.5.2. LR(1) Parser
- 8.5.3. LR(k) Parser
- 8.5.4. LALR Parser

8.6. Semantic Analysis (I)
- 8.6.1. Syntax-Driven Translation
- 8.6.2. Table of Symbols

8.7. Semantic Analysis (II)
- 8.7.1. Type Checking
- 8.7.2. The Type Subsystem
- 8.7.3. Type Equivalence and Conversions

8.8. Code Generation and Execution Environment
- 8.8.1. Design Aspects
- 8.8.2. Execution Environment
- 8.8.3. Memory Organization
- 8.8.4. Memory Allocation

8.9. Intermediate Code Generation
- 8.9.1. Syntax-Driven Translation
- 8.9.2. Intermediate Representations
- 8.9.3. Examples of Translations

8.10. Code Optimization
- 8.10.1. Register Allocation
- 8.10.2. Elimination of Dead Assignments
- 8.10.3. Compile-Time Execution
- 8.10.4. Expression Reordering
- 8.10.5. Loop Optimization

Module 9. Computer Graphics and Visualization

9.1. Color Theory
- 9.1.1. Properties of Light
- 9.1.2. Color Models
- 9.1.3. The CIE Standard
- 9.1.4. *Profiling*

9.2. Output Primitives
- 9.2.1. The Video Driver
- 9.2.2. Line Drawing Algorithms
- 9.2.3. Circle Drawing Algorithms
- 9.2.4. Filling Algorithms

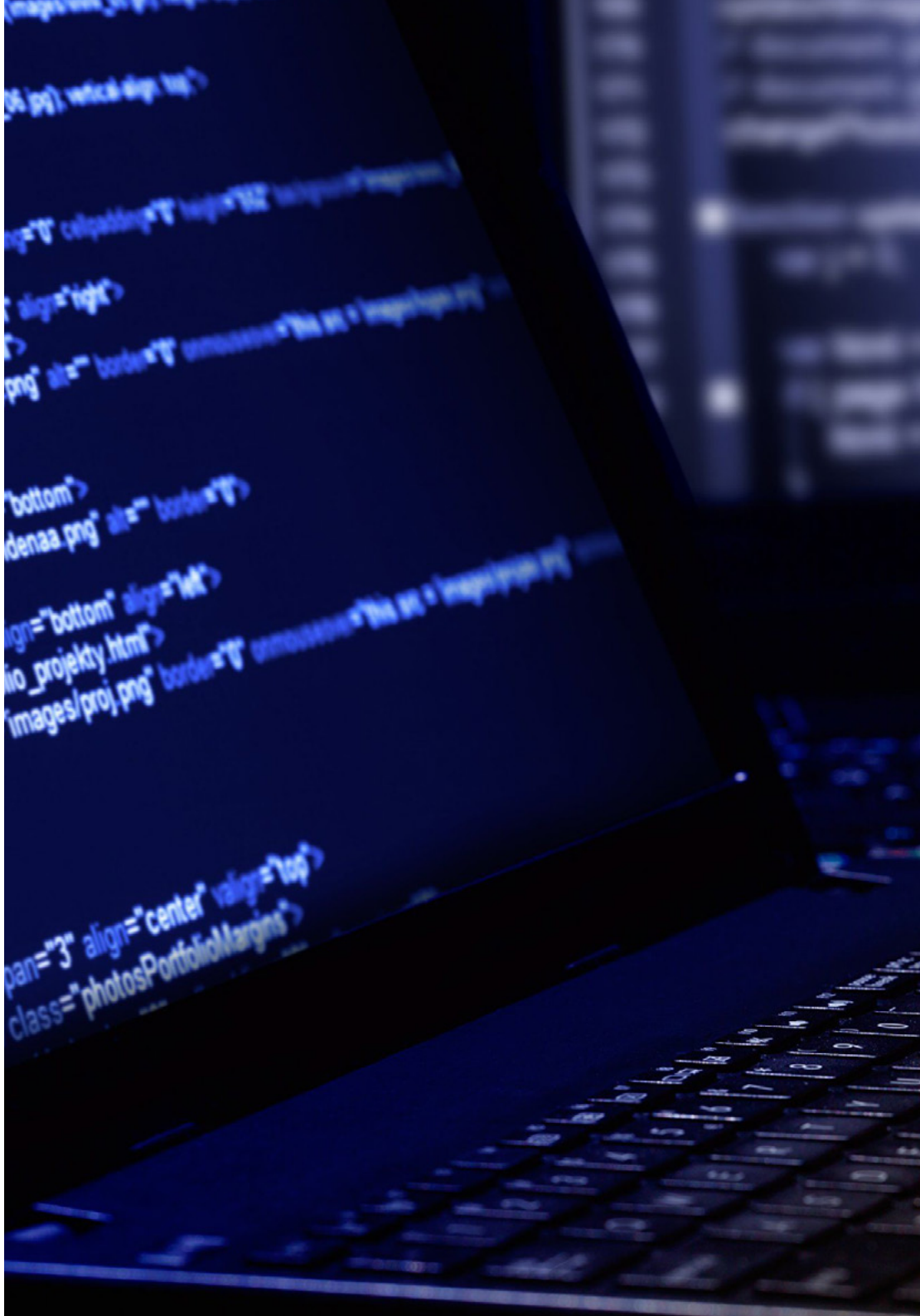
9.3. 2D Transformations and 2D Coordinate Systems and 2D Clipping
- 9.3.1. Basic Geometric Transformations
- 9.3.2. Homogeneous Coordinates
- 9.3.3. Inverse Transformation
- 9.3.4. Composition of Transformations
- 9.3.5. Other Transformations
- 9.3.6. Coordinate Change
- 9.3.7. 2D Coordinate Systems
- 9.3.8. Coordinate Change
- 9.3.9. Standardization
- 9.3.10. Trimming Algorithms

9.4. 3D Transformations
- 9.4.1. Translation
- 9.4.2. Rotation
- 9.4.3. Scaling
- 9.4.4. Reflection
- 9.4.5. Shearing

9.5. Display and Change of 3D Coordinates
- 9.5.1. 3D Coordinate Systems
- 9.5.2. Visualization
- 9.5.3. Coordinate Change
- 9.5.4. Projection and Normalization

9.6. 3D Projection and Clipping
- 9.6.1. Orthogonal Projection
- 9.6.2. Oblique Parallel Projection
- 9.6.3. Perspective Projection
- 9.6.4. 3D Clipping Algorithms

9.7. Hidden Surface Removal
- 9.7.1. *Back-Face Removal*
- 9.7.2. Z-Buffer
- 9.7.3. Painter Algorithm
- 9.7.4. Warnock Algorithm
- 9.7.5. Hidden Line Detection



9.8. Interpolation and Parametric Curves
- 9.8.1. Interpolation and Polynomial Approximation
- 9.8.2. Parametric Representation
- 9.8.3. Lagrange Polynomial
- 9.8.4. Natural Cubic *Splines*
- 9.8.5. Basic Functions
- 9.8.6. Matrix Representation

9.9. Bézier Curves
- 9.9.1. Algebraic Construction
- 9.9.2. Matrix Form
- 9.9.3. Composition
- 9.9.4. Geometric Construction
- 9.9.5. Drawing Algorithm

9.10. *B-Splines*
- 9.10.1. The Local Control Problem
- 9.10.2. Uniform Cubic B-Splines
- 9.10.3. Basis Functions and Control Points
- 9.10.4. Derivative to the Origin and Multiplicity
- 9.10.5. Matrix Representation
- 9.10.6. Non-Uniform B-Splines

Module 10. Bioinspired Computing

10.1. Introduction to Bioinspired Computing
- 10.1.1. Introduction to Bioinspired Computing

10.2. Social Adaptation Algorithms
- 10.2.1. Bioinspired Computation Based on Ant Colonies
- 10.2.2. Variants of Ant Colony Algorithms
- 10.2.3. Particle Cloud Computing

10.3. Genetic Algorithms
- 10.3.1. General Structure
- 10.3.2. Implementations of the Major Operators

10.4. Space Exploration and Exploitation Strategies for Genetic Algorithms
- 10.4.1. CHC Algorithm
- 10.4.2. Multimodal Problems

10.5. Evolutionary Computing Models (I)
- 10.5.1. Evolutionary Strategies
- 10.5.2. Evolutionary Programming
- 10.5.3. Algorithms Based on Differential Evolution

10.6. Evolutionary Computing Models (II)
- 10.6.1. Evolutionary Models Based on Estimation of Distributions (EDA)
- 10.6.2. Genetic Programming

10.7. Evolutionary Programming Applied to Learning Problems
- 10.7.1. Rules-Based Learning
- 10.7.2. Evolutionary Methods in Instance Selection Problems

10.8. Multi-Objective Problems
- 10.8.1. Concept of Dominance
- 10.8.2. Application of Evolutionary Algorithms to Multi-Objective Problems

10.9. Neural Networks (I)
- 10.9.1. Introduction to Neural Networks
- 10.9.2. Practical Example with Neural Networks

10.10. Neural Networks (II)
- 10.10.1. Use Cases of Neural Networks in Medical Research
- 10.10.2. Use Cases of Neural Networks in Economics
- 10.10.3. Use Cases of Neural Networks in Artificial Vision

You will develop skills for the formal and semantic analysis of programs, significantly improving software efficiency"

04

Teaching Objectives

This Master's Degree aims to deepen the theoretical and practical foundations that underpin the design, analysis, and optimization of computational languages. To this end, it combines the study of grammatical structures, semantics, and models of computation with tools applied to the development of intelligent systems. Likewise, it promotes abstract thinking and rigorous problem-solving, which are essential in high-demand technological environments. Thanks to this comprehensive perspective, the program fosters the capacity to innovate in emerging areas such as natural language processing, formal languages, and compiler architecture.

You will integrate computational thinking into technological innovation projects, enabling you to offer advanced solutions to different organizations"

General Objectives

- Gain an in-depth understanding of the theoretical foundations of programming languages and their application in advanced and specialized computational environments
- Critically analyze the formal structures that underpin the design and evolution of computational languages in different contexts
- Confidently apply models of computation to efficiently solve complex problems in the field of modern computer science
- Master the essential principles of formal semantics and their direct relationship with the accurate execution of computer programs
- Study automata, grammars, and formal languages rigorously from a comprehensive computational and scientific perspective
- Integrate fundamental knowledge of computation theory with current and future technological developments in the sector
- Systematically evaluate the performance and efficiency of different languages, compilers, and advanced development environments
- Design innovative computational solutions based on detailed analysis of logical, syntactic, and semantic structures
- Develop specialized competencies for applied research in the field of computing and formal languages
- Interpret with discernment the impact of computational languages on the ongoing evolution of modern intelligent systems

Specific Objectives

Module 1. Programming Fundamentals

- Identify the basic elements of a programming language
- Apply control structures in problem solving
- Use variables, operators, and functions in simple programs
- Develop structured programs following good practices

Module 2. Data Structures

- Implement linear structures such as lists and stacks
- Use trees and graphs in computational representations
- Evaluate the efficiency of different data structures
- Select appropriate structures according to the problem posed

Module 3. Algorithmics and Complexity

- Analyze the time and space efficiency of algorithms
- Classify algorithms according to their computational complexity
- Solve problems using basic algorithmic paradigms
- Compare solution alternatives based on their performance

Module 4. Advanced Algorithm Design

- Apply dynamic programming and greedy techniques
- Design algorithms for complex optimization problems
- Use advanced structures in algorithm construction
- Evaluate the performance of advanced algorithmic solutions

Module 5. Advanced Programming

- Develop programs using object-oriented programming concepts
- Integrate error handling and debugging in complex projects
- Implement design patterns in scalable applications
- Use libraries and frameworks to accelerate development

Module 6. Theoretical Computer Science

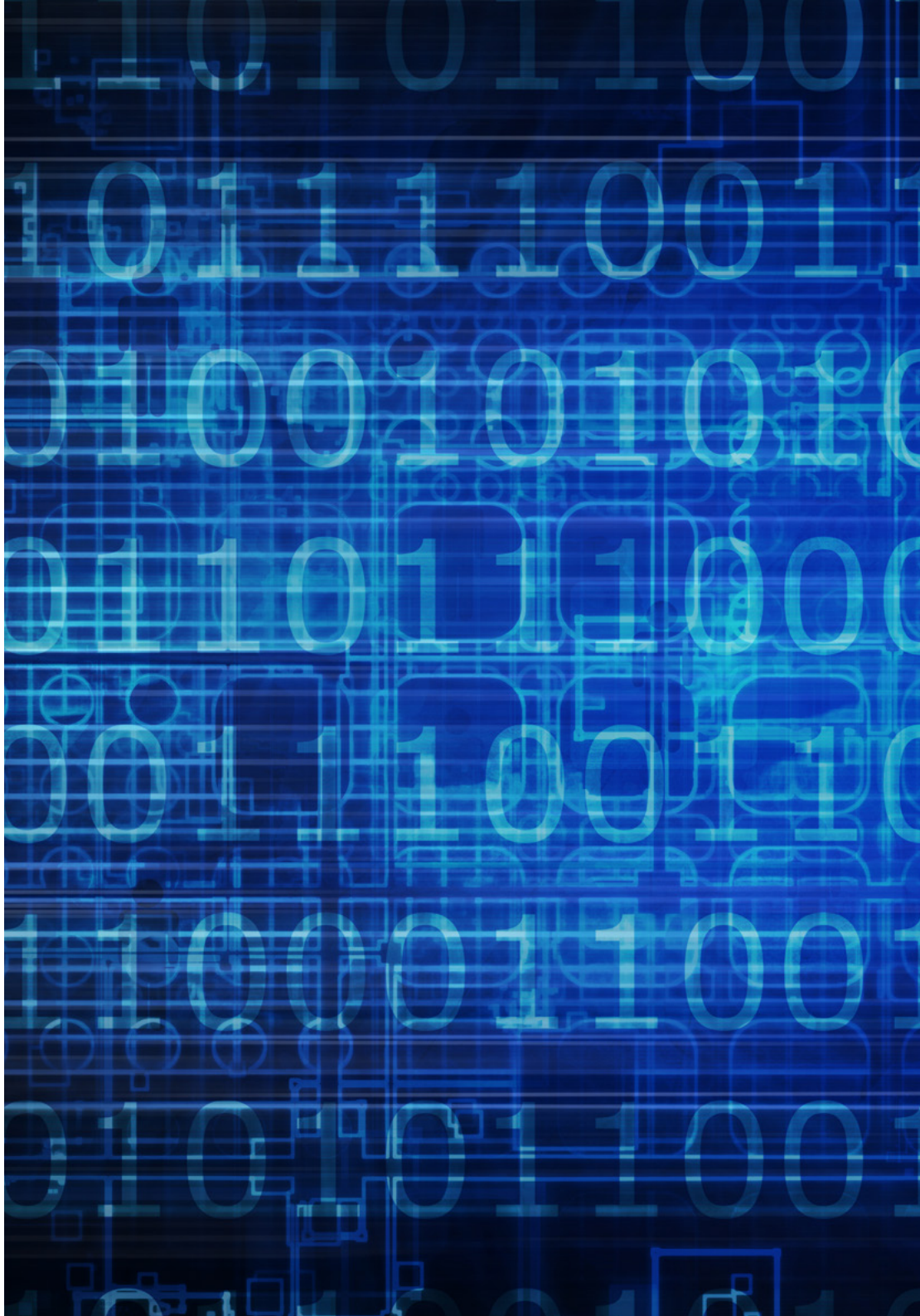
- Understand the limits of computation through theoretical models
- Study non-computable problems and their classification
- Analyze the P and NP classes in computational theory
- Apply theoretical concepts to the resolution of abstract problems

Module 7. Automata Theory and Formal Languages

- Design deterministic and nondeterministic finite automata
- Generate regular and context-free grammars
- Analyze formal languages using automata and regular expressions
- Relate language types to formal hierarchies

Module 8. Language Processors

- Implement basic lexical and syntactic analyzers
- Understand the translation process in compilers
- Design grammars for specific programming languages
- Integrate compiler phases into a functional environment



Module 9. Computer Graphics and Visualization

- Use graphics libraries to represent data visually
- Understand the fundamentals of computational graphic modeling
- Implement two- and three-dimensional rendering techniques
- Design interactive information visualization systems

Module 10. Bioinspired Computing

- Apply evolutionary algorithms to solve complex problems
- Understand the fundamentals of neural networks and their training
- Use bioinspired strategies in artificial intelligence contexts
- Compare bioinspired methods with traditional algorithms

Project yourself toward emerging sectors such as quantum computing, natural language processing, and advanced cybersecurity. Make yourself indispensable!"

05

Career Opportunities

The constant advancement of artificial intelligence, formal languages, and software engineering has significantly expanded employment opportunities in the field of computing. Consequently, possessing a solid specialization in these areas enables access to highly in-demand profiles such as language designer, compiler developer, researcher in computational theory, or expert in algorithmic analysis. In addition, this type of preparation is essential for entering strategic sectors such as cybersecurity, natural language processing, or quantum computing, where in-depth mastery of computational foundations is increasingly valued.

You will model computational processes using cutting-edge formal and mathematical systems”

Graduate Profile

This program fosters the development of a technical and analytical profile capable of understanding and applying the principles that underpin modern computational logic. Through an integrative vision that combines theory and practice, graduates are characterized by their ability to design efficient solutions, model complex systems, and develop innovative tools across diverse technological environments. Moreover, their mastery of formal languages, advanced algorithms, and computational structures positions them as highly competent professionals in academic, scientific, or industrial settings where deep knowledge of the internal behavior of digital systems is required.

You will incorporate bioinspired computing techniques into innovation projects that emulate natural and evolutionary processes.

- **Critical Thinking and Complex Problem-Solving:** Ability to analyze situations from a logical perspective and propose well-founded solutions
- **Effective Technical Communication:** Ability to express computational concepts clearly and precisely, both orally and in writing
- **Collaborative Work in Multidisciplinary Environments:** Willingness to integrate and contribute within teams composed of diverse technological profiles
- **Adaptability to Change and Continuous Learning:** Willingness to continuously update knowledge in a sector defined by constant innovation

After completing the university program, you will be able to apply your knowledge and skills in the following positions:

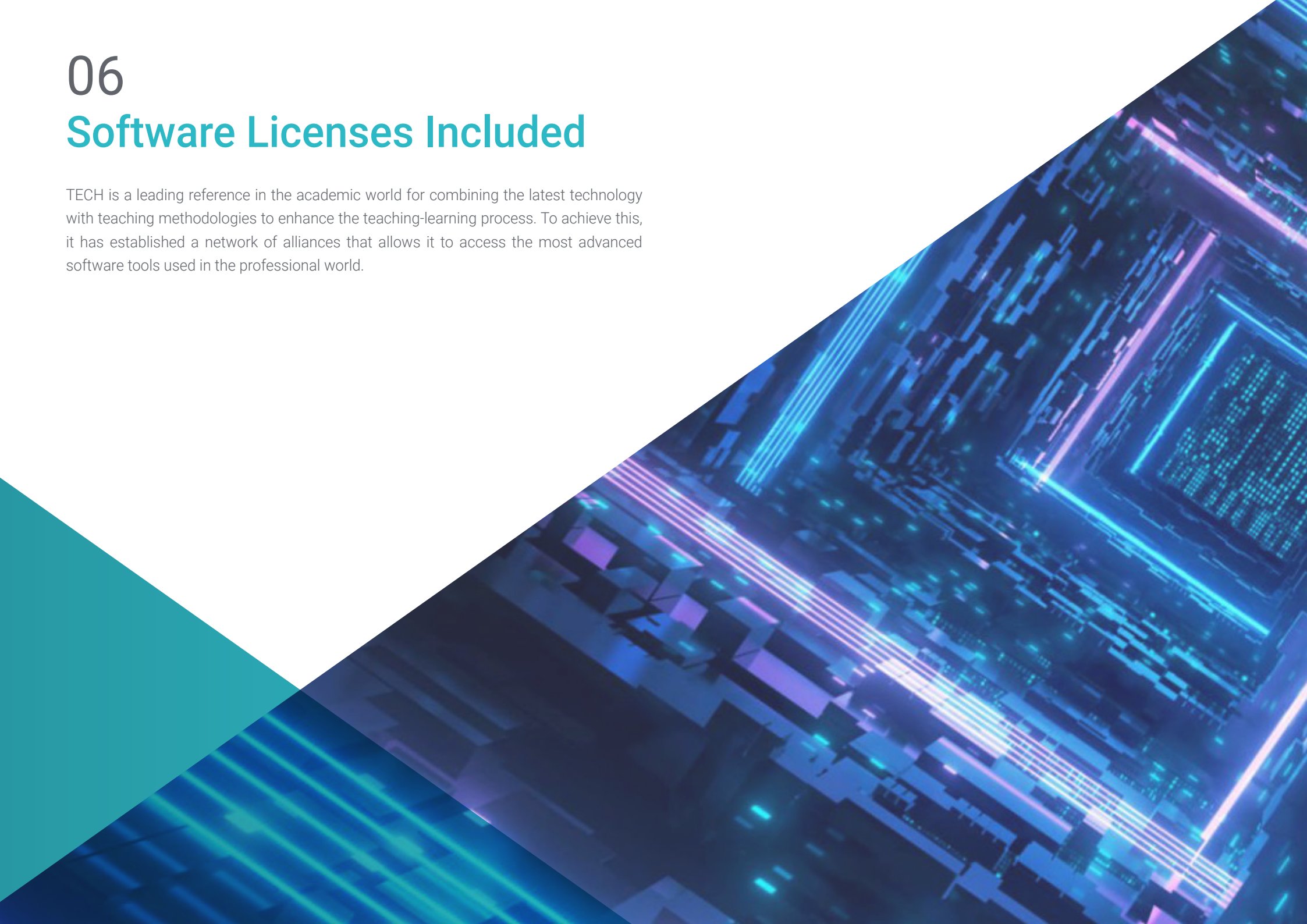
1. **Compiler Software Engineer:** Designs and implements programming language translators, optimizing their performance and functionality.
2. **Programming Language Architect:** Creates new languages or adapts existing ones according to the specific needs of technological sectors.
3. **Theoretical Computer Science Researcher:** Studies models of computation, algorithmic complexity, and formal languages to solve abstract problems applicable to real systems.
4. **Intelligent Systems Developer:** Builds solutions based on computational logic and natural language processing for advanced applications.
5. **Algorithm Analyst:** Evaluates and optimizes algorithms to improve efficiency in high-performance computing contexts.
6. **Natural Language Processing Specialist:** Designs systems capable of interpreting, generating, or translating human language in digital environments.
7. **Computer Graphics Engineer:** Develops interactive visual solutions, simulations, and high-level graphical environments for various industries.
8. **Computational Structures Consultant:** Advises on the selection and implementation of models and structures for efficient software design.

"You will implement interactive graphical solutions and useful visualizations in software development"

06

Software Licenses Included

TECH is a leading reference in the academic world for combining the latest technology with teaching methodologies to enhance the teaching-learning process. To achieve this, it has established a network of alliances that allows it to access the most advanced software tools used in the professional world.



Upon enrolling, you will receive, completely free of charge, academic credentials for the following professional software applications"

TECH has established a network of professional alliances with the leading providers of software applied to various professional fields. These alliances allow TECH to access hundreds of software applications and licenses, making them available to its students.

The software licenses for academic use will allow students to utilize the most advanced applications in their professional field, enabling them to become familiar with and master these tools without incurring any costs. TECH will handle the licensing process so that students can use them unlimitedly during the time they are enrolled in the Master's Degree in Computing and Programming Languages, and they will be able to do so completely free of charge.

TECH will provide free access to the following software applications:

Google Career Launchpad

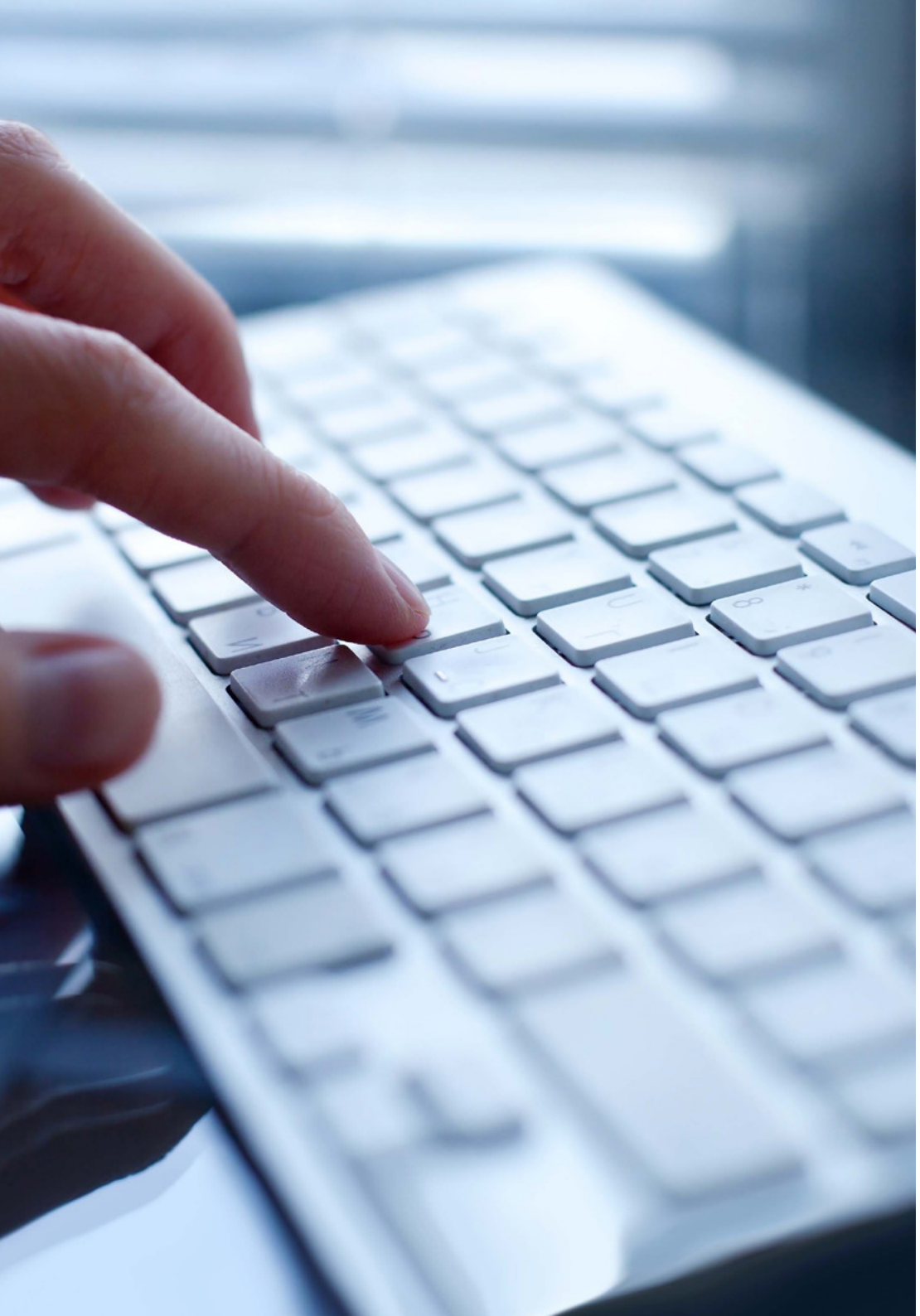
Google Career Launchpad is a solution for developing digital skills in technology and data analysis. With an estimated value of **5,000 dollars**, it is included **for free** in TECH's university program, providing access to interactive labs and certifications recognized in the industry.

This platform combines technical training with practical cases, using technologies such as BigQuery and Google AI. It offers simulated environments to work with real data, along with a network of experts for personalized guidance.

Key Features:

- **Specialized Courses:** Updated content in cloud computing, machine learning, and data analysis
- **Live Labs:** Hands-on practice with real Google Cloud tools, no additional configuration required
- **Integrated Certifications:** Preparation for official exams with international validity
- **Professional Mentoring:** Sessions with Google experts and technology partners
- **Collaborative Projects:** Challenges based on real-world problems from leading companies

In conclusion, **Google Career Launchpad** connects users with the latest market technologies, facilitating their entry into fields such as artificial intelligence and data science with industry-backed credentials.



DBeaver Enterprise Edition

DBeaver Enterprise Edition is the professional version of the renowned database management tool DBeaver, with a commercial price of approximately **250 euros** annually. It is offered **for free** during the university program at TECH, enabling graduates to professionally and securely manage, develop, and analyze data in complex environments.

This platform equips TECH graduates to optimize the management of relational and non-relational databases, generate smart SQL queries, design advanced schemas, and visualize information with interactive charts. Additionally, it integrates business analysis functions by connecting with *Business Intelligence* tools, transforming data into strategic knowledge for decision-making.

Key Features:

- **Wide Compatibility:** Supports Oracle, SQL Server, PostgreSQL, MongoDB, Cassandra, and more
- **Advanced SQL Editor:** Autocompletion, debugging, and smart assistants
- **Data Visualization:** Interactive dashboards and integrated charts
- **Integration with Tableau:** Direct connection to Business Intelligence tools
- **Schema Design:** ERD editing and reverse engineering
- **Full Administration:** Backup, restoration, comparison, and user management

In conclusion, **DBeaver Enterprise Edition** empowers TECH graduates to master data management with precision, efficiency, and innovation.

07

Study Methodology

TECH is the first university in the world to combine case study methodology with Relearning, a 100% online learning system based on guided repetition.

This innovative pedagogical strategy has been conceived to offer professionals the opportunity to update knowledge and develop skills in an intensive and rigorous way. A learning model that places the student at the center of the academic process and gives them the leading role, adapting to their needs and leaving aside the more conventional methodologies.

TECH prepares you to face new challenges in uncertain environments and achieve success in your career"

The student: the priority of all TECH programs

In TECH's study methodology, the student is the main protagonist.
The teaching tools of each program have been selected taking into account the demands of time, availability and academic rigor that, today, not only students demand but also the most competitive positions in the market.

With TECH's asynchronous educational model, it is students who choose the time they dedicate to study, how they decide to establish their routines, and all this from the comfort of the electronic device of their choice. The student will not have to participate in live classes, which in many cases they will not be able to attend. The learning activities will be done when it is convenient for them. They can always decide when and from where they want to study.

"At TECH you will NOT have live classes (which you might not be able to attend)"

The most comprehensive academic programs worldwide

TECH is distinguished by offering the most complete academic itineraries on the university scene. This comprehensiveness is achieved through the creation of syllabi that not only cover the essential knowledge, but also the most recent innovations in each area.

By being constantly up to date, these programs allow students to keep up with market changes and acquire the skills most valued by employers. In this way, those who complete their studies at TECH receive a comprehensive education that provides them with a notable competitive advantage to further their careers.

And what's more, they will be able to do so from any device, pc, tablet or smartphone.

> *TECH's model is asynchronous, so it allows you to study with your pc, tablet or your smartphone wherever you want, whenever you want and for as long as you want"*

Case Studies or Case Method

The case method has been the learning system most used by the world's best business schools. Developed in 1912 so that law students would not only learn the law based on theoretical content, its function was also to present them with real complex situations. In this way, they could make informed decisions and value judgments about how to resolve them. In 1924, Harvard adopted it as a standard teaching method.

With this teaching model, it is students themselves who build their professional competence through strategies such as Learning by Doing or Design Thinking, used by other renowned institutions such as Yale or Stanford.

This action-oriented method will be applied throughout the entire academic itinerary that the student undertakes with TECH. Students will be confronted with multiple real-life situations and will have to integrate knowledge, research, discuss and defend their ideas and decisions. All this with the premise of answering the question of how they would act when facing specific events of complexity in their daily work.

Relearning **Method**

At TECH, case studies are enhanced with the best 100% online teaching method: Relearning.

This method breaks with traditional teaching techniques to put the student at the center of the equation, providing the best content in different formats. In this way, it manages to review and reiterate the key concepts of each subject and learn to apply them in a real context.

In the same line, and according to multiple scientific researches, reiteration is the best way to learn. For this reason, TECH offers between 8 and 16 repetitions of each key concept within the same lesson, presented in a different way, with the objective of ensuring that the knowledge is completely consolidated during the study process.

Relearning will allow you to learn with less effort and more performance, involving you more in your specialization, developing a critical spirit, defending arguments and contrasting opinions: a direct equation to success.

A 100% online Virtual Campus with the best teaching resources

In order to apply its methodology effectively, TECH focuses on providing graduates with teaching materials in different formats: texts, interactive videos, illustrations and knowledge maps, among others. All of them are designed by qualified teachers who focus their work on combining real cases with the resolution of complex situations through simulation, the study of contexts applied to each professional career and learning based on repetition, through audios, presentations, animations, images, etc.

The latest scientific evidence in the field of Neuroscience points to the importance of taking into account the place and context where the content is accessed before starting a new learning process. Being able to adjust these variables in a personalized way helps people to remember and store knowledge in the hippocampus to retain it in the long term. This is a model called Neurocognitive context-dependent e-learning that is consciously applied in this university qualification.

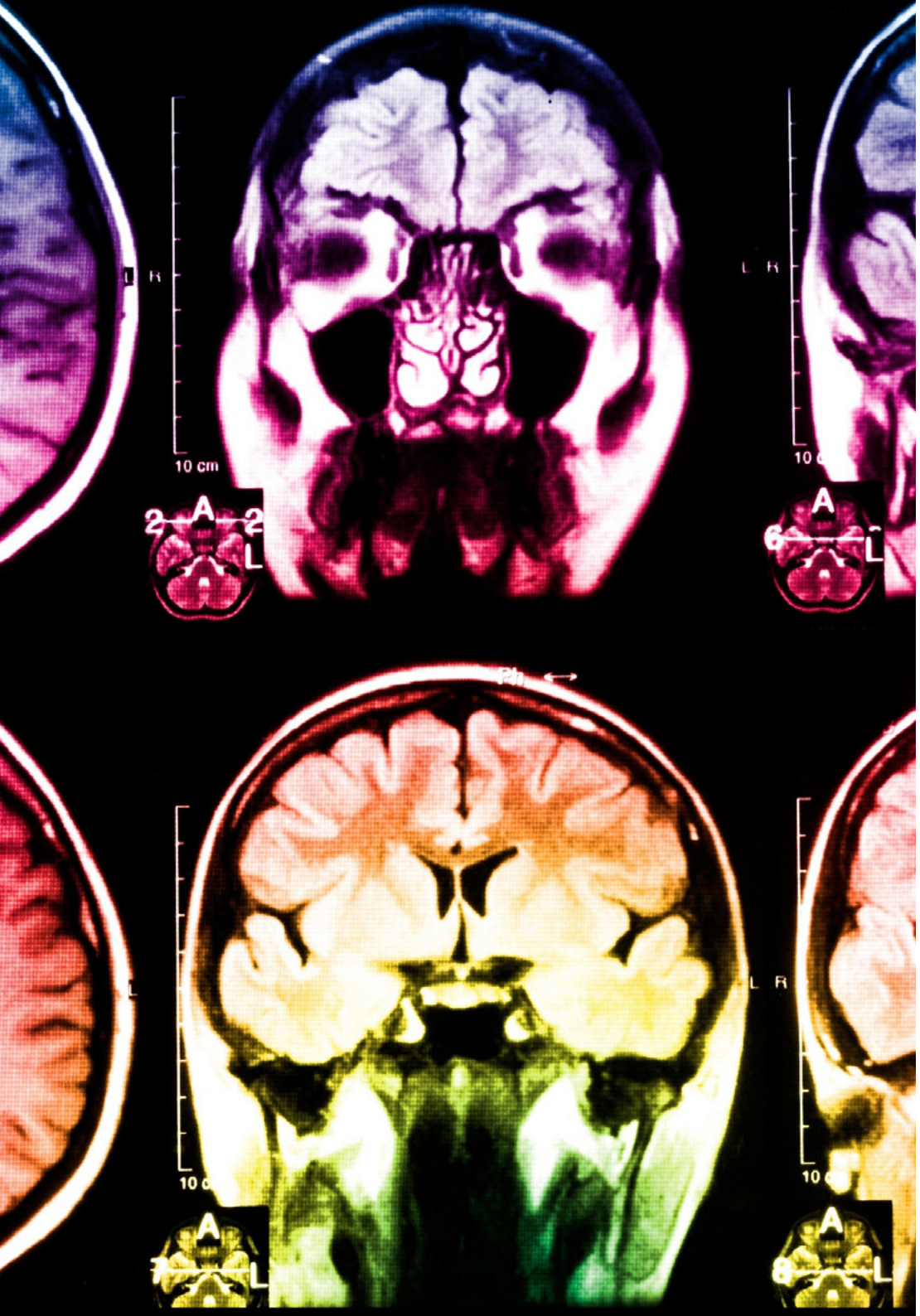
In order to facilitate tutor-student contact as much as possible, you will have a wide range of communication possibilities, both in real time and delayed (internal messaging, telephone answering service, email contact with the technical secretary, chat and videoconferences).

Likewise, this very complete Virtual Campus will allow TECH students to organize their study schedules according to their personal availability or work obligations. In this way, they will have global control of the academic content and teaching tools, based on their fast-paced professional update.

"The online study mode of this program will allow you to organize your time and learning pace, adapting it to your schedule"

The effectiveness of the method is justified by four fundamental achievements:

1. Students who follow this method not only achieve the assimilation of concepts, but also a development of their mental capacity, through exercises that assess real situations and the application of knowledge.
2. Learning is solidly translated into practical skills that allow the student to better integrate into the real world.
3. Ideas and concepts are understood more efficiently, given that the example situations are based on real-life.
4. Students like to feel that the effort they put into their studies is worthwhile. This then translates into a greater interest in learning and more time dedicated to working on the course.



The university methodology best rated by its students

The results of this innovative teaching model can be seen in the overall satisfaction levels of TECH graduates.

The students' assessment of the teaching quality, the quality of the materials, the structure of the program and its objectives is excellent. Not surprisingly, the institution became the top-rated university by its students according to the global score index, obtaining a 4.9 out of 5.

Access the study contents from any device with an Internet connection (computer, tablet, smartphone) thanks to the fact that TECH is at the forefront of technology and teaching.

You will be able to learn with the advantages that come with having access to simulated learning environments and the learning by observation approach, that is, Learning from an expert.

As such, the best educational materials, thoroughly prepared, will be available in this program:

Study Material

All teaching material is produced by the specialists who teach the course, specifically for the course, so that the teaching content is highly specific and precise.

This content is then adapted in an audiovisual format that will create our way of working online, with the latest techniques that allow us to offer you high quality in all of the material that we provide you with.

Practicing Skills and Abilities

You will carry out activities to develop specific competencies and skills in each thematic field. Exercises and activities to acquire and develop the skills and abilities that a specialist needs to develop within the framework of the globalization we live in.

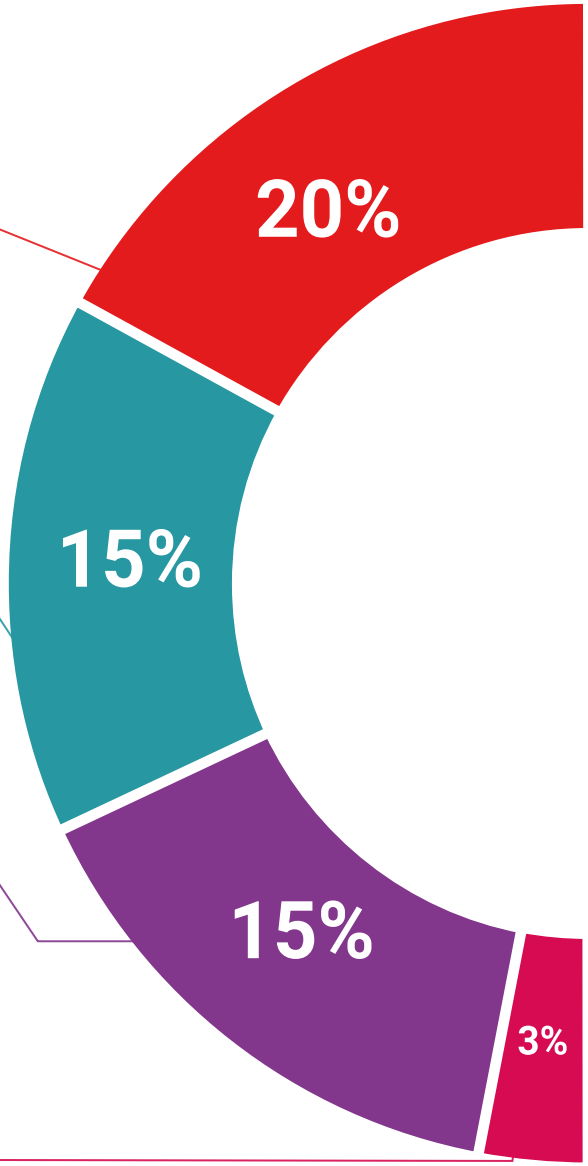
Interactive Summaries

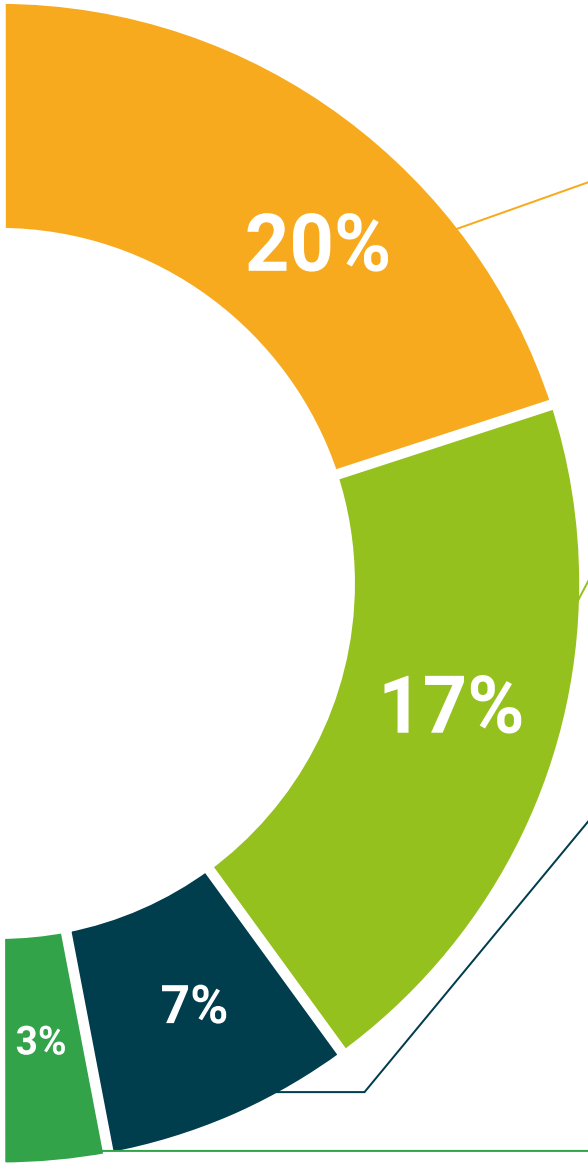
We present the contents attractively and dynamically in multimedia lessons that include audio, videos, images, diagrams, and concept maps in order to reinforce knowledge.

This exclusive educational system for presenting multimedia content was awarded by Microsoft as a "European Success Story".

Additional Reading

Recent articles, consensus documents, international guides... In our virtual library you will have access to everything you need to complete your education.





Case Studies

Students will complete a selection of the best case studies in the field. Cases that are presented, analyzed, and supervised by the best specialists in the world.

Testing & Retesting

We periodically assess and re-assess your knowledge throughout the program. We do this on 3 of the 4 levels of Miller's Pyramid.

Classes

There is scientific evidence suggesting that observing third-party experts can be useful.

Learning from an expert strengthens knowledge and memory, and generates confidence for future difficult decisions.

Quick Action Guides

TECH offers the most relevant contents of the course in the form of worksheets or quick action guides. A synthetic, practical and effective way to help students progress in their learning.

08

Teaching Staff

The faculty of this program is composed of experts with extensive academic and professional experience in key areas of computing and formal languages. Thanks to their background in applied research and the development of advanced technological solutions, they offer an up-to-date and rigorous perspective on the sector. In addition, their participation in international projects allows real-world cases and innovative methodologies to be brought into the classroom, significantly enriching the learning process. This combination of in-depth knowledge and a practical approach ensures a high-level academic experience aligned with the demands of the current scientific and technological environment.

You will be advised at all times by the faculty team, made up of renowned specialists in Computing and Programming Languages"

International Guest Director

Dr. Jeremy Gibbons is considered an **international eminence** for his contributions in the field of **Programming Methodology** and its applications in **Software Engineering**. For more than two decades, this expert, associated with the Department of Computer Science at the University of Oxford, has driven **different development projects** whose most tangible results are applied by computer scientists from different parts of the world.

His work covers areas such as **generic programming**, formal methods, computational biology, bioinformatics and algorithm design with Haskell. This last topic he developed extensively in conjunction with his mentor, Dr. Richard Bird.

In his role as **Director** of the **Algebra of Programming Research Group**, Gibbons has led advances in **Functional Programming Languages** and **Pattern Theory in Programming**. At the same time, the applications of his innovations have been linked to the healthcare framework, as evidenced by his collaboration with **CancerGrid** and **Datatype-Generic Programming**. These and other initiatives reflect his interest in solving practical problems in **cancer research** and **clinical informatics**.

Gibbons has also made a significant mark as **Editor-in-Chief** of **scholarly publications** in The Journal of Functional Programming and The Programming Journal: The Art, Science, and Engineering of Programming. Through these responsibilities he has carried out intensive **outreach** and **dissemination of knowledge**. In addition, he has led several study chairs linked to renowned institutions such as Oxford Brookes University and the University of Auckland, New Zealand.

Moreover, this specialist is a member of the Working Group 2.1 on Algorithmic Languages and Computation of the **International Federation for Information Processing (IFIP)**. With this organization, he provides maintenance for the ALGOL 60 and ALGOL 68 programming languages

Dr. Gibbons, Jeremy

- Director, Software Engineering Program, University of Oxford, United Kingdom
- Deputy Head of the Informatics Laboratory and Department of Computer Science, University of Oxford
- Professor at Kellogg College, Oxford Brookes University and the University of Auckland, New Zealand
- Director of the Algebra of Programming Research Group
- Editor-in-Chief of The Art, Science, and Engineering of Programming and the Journal of Functional Programming
- Doctorate in Computer Science from Oxford University
- Bachelor's Degree in Computer Science from the University of Edinburgh
- Member of: Working Group 2.1 on Algorithmic Languages and Computation of the International Federation for Information Processing (IFIP)

Thanks to TECH, you will be able to learn with the best professionals in the world"

09

Certificate

The Master's Degree in Computing and Programming Languages guarantees students, in addition to the most rigorous and up-to-date education, access to a diploma for the Master's Degree issued by TECH Global University.

Successfully complete this program and receive your university qualification without having to travel or fill out laborious paperwork"

This private qualification will allow you to obtain a diploma for the **Master's Degree in Computing and Programming Languages** endorsed by **TECH Global University**, the world's largest online university.

TECH Global University, is an official European University publicly recognized by the Government of Andorra (*official bulletin*). Andorra is part of the European Higher Education Area (EHEA) since 2003. The EHEA is an initiative promoted by the European Union that aims to organize the international training framework and harmonize the higher education systems of the member countries of this space. The project promotes common values, the implementation of collaborative tools and strengthening its quality assurance mechanisms to enhance collaboration and mobility among students, researchers and academics.

This private qualification from **TECH Global University** is a European continuing education and professional development program that guarantees the acquisition of competencies in its area of expertise, providing significant curricular value to the student who successfully completes the program.

TECH is a member of the **Association for Computing Machinery (ACM)**, the international network that brings together leading experts in computing and information sciences. This membership strengthens its commitment to academic excellence, technological innovation, and the training of professionals in the digital field.

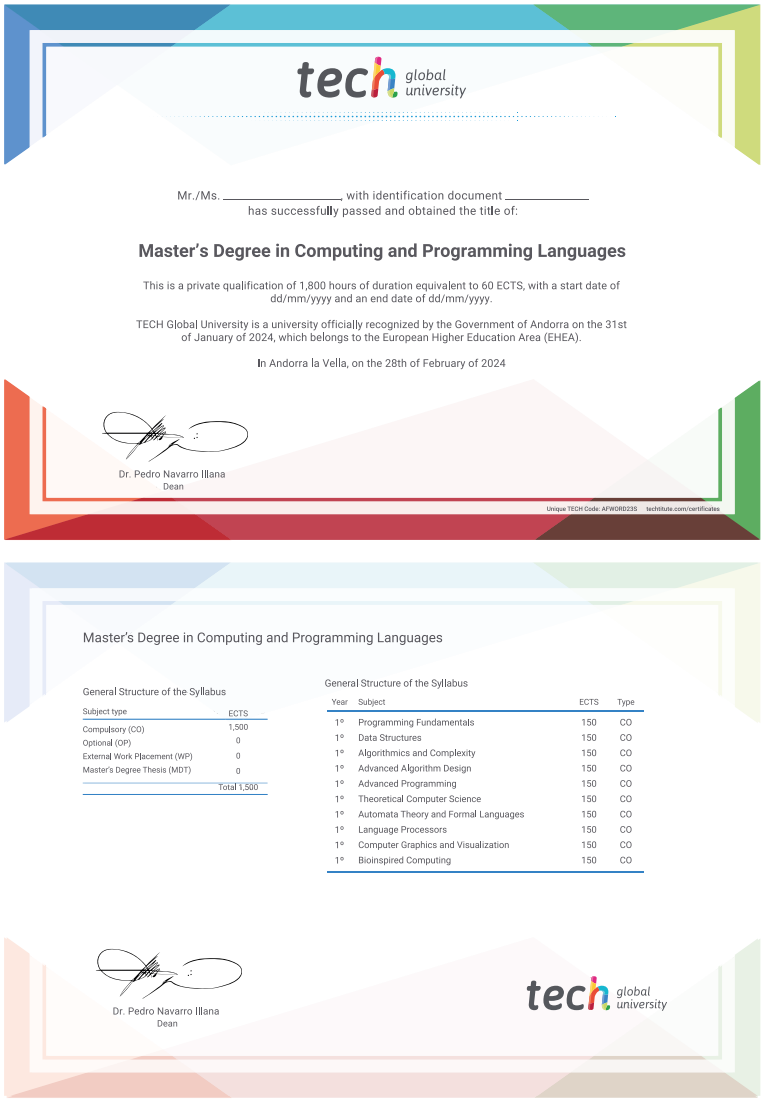
Accreditation/Membership

Title: **Master's Degree in Computing and Programming Languages**

Modality: **online**

Duration: **12 months.**

Accreditation: **60 ECTS**



tech global university

Mr./Ms. _____ with identification document _____
has successfully passed and obtained the title of:

Master's Degree in Computing and Programming Languages

This is a private qualification of 1,800 hours of duration equivalent to 60 ECTS, with a start date of dd/mm/yyyy and an end date of dd/mm/yyyy.

TECH Global University is a university officially recognized by the Government of Andorra on the 31st of January of 2024, which belongs to the European Higher Education Area (EHEA).

In Andorra la Vella, on the 28th of February of 2024

Dr. Pedro Navarro Illana
Dean

Unique TECH Code: AFWOR023S techtitute.com/certificates

Master's Degree in Computing and Programming Languages

General Structure of the Syllabus

Subject type	ECTS
Compulsory (CO)	1,500
Optional (OP)	0
External Work Placement (WP)	0
Master's Degree Thesis (MDT)	0
	Total 1,500

General Structure of the Syllabus

Year	Subject	ECTS	Type
1°	Programming Fundamentals	150	CO
1°	Data Structures	150	CO
1°	Algorithmics and Complexity	150	CO
1°	Advanced Algorithm Design	150	CO
1°	Advanced Programming	150	CO
1°	Theoretical Computer Science	150	CO
1°	Automata Theory and Formal Languages	150	CO
1°	Language Processors	150	CO
1°	Computer Graphics and Visualization	150	CO
1°	Bioinspired Computing	150	CO

Dr. Pedro Navarro Illana
Dean

tech global university

*Apostille Convention. In the event that the student wishes to have their paper diploma issued with an apostille, TECH Global University will make the necessary arrangements to obtain it, at an additional cost.

tech global university

Master's Degree
Computing and Programming Languages

- Modality: **online**
- Duration: **12 months.**
- Certificate: **TECH Global University**
- Accreditation: **60 ECTS**
- Schedule: **at your own pace**
- Exams: **online**

Master's Degree

Computing and Programming Languages

Accreditation/Membership

Association
for Computing
Machinery

tech global university