Máster Semipresencial
Desarrollo de Software

tech universidad tecnológica

tech universidad tecnológica

Máster Semipresencial
Desarrollo de Software

Modalidad: **Semipresencial (Online + Prácticas)**
Duración: **12 meses**
Titulación: **TECH Universidad Tecnológica**
Acceso web: www.techtitute.com/informatica/master-semipresencial/master-semipresencial-desarrollo-software

Índice

01

Presentación

El Desarrollo de Software se ha convertido en un componente crucial de la infraestructura tecnológica en la mayoría de las industrias, desde la salud hasta las finanzas y el entretenimiento. Con la creciente complejidad de las aplicaciones, los informáticos deben adaptarse a un entorno de cambio constante. Esto requiere que los desarrolladores actualicen sus conocimientos con frecuencia para mantenerse al corriente de los últimos avances en materias como los lenguajes de programación o estrategias para la creación de algoritmos. En este contexto, TECH presenta una innovadora titulación universitaria que ahondará en las innovaciones más recientes en el ámbito del Desarrollo de Software.

Gracias a este Máster Semipresencial, aplicarás patrones de diseños para resolver una amplia gama de problemas informáticos y construir soluciones escalables”

El Desarrollo de Software se ha convertido en factor esencial en la economía digital actual, con una demanda creciente de profesionales capacitados en todo el mundo. Según la Organización para la Cooperación y el Desarrollo Económicos, se espera que este sector profesional experimente un crecimiento del 30% de cara al próximo año. Esto subraya la importancia de que los profesionales de la Informática permanezcan al corriente de las últimas tendencias en este campo. De lo contrario, los desarrolladores podrían enfrentarse a problemáticas como el uso de metodologías desactualizadas que conduzcan a procesos menos eficientes.

En este marco, TECH lanza un revolucionario Máster Semipresencial en Desarrollo de Software. Se trata de un programa universitario que aúna el mejor contenido teórico con 3 semanas de estancia práctica en una entidad de referencia en este ámbito. El itinerario académico profundizará en aspectos como la Programación en C++, la creación de bases de datos avanzadas o el diseño arquitectónico de las aplicaciones. Todo ello mediante materiales didácticos confeccionados por un experimentado claustro docente, que incluyen un amplio abanico de recursos multimedia (como resúmenes interactivos, casos de estudio o vídeos explicativos) para garantizar una puesta al día amena. De este modo, los informáticos disfrutarán de un aprendizaje totalmente progresivo y natural, sin tener que recurrir a técnicas tradicionales como la memorización.

Por otro lado, la titulación universitaria prevé que los egresados lleven a cabo una capacitación práctica en una prestigiosa entidad. Allí los informáticos participarán activamente en los proyectos que se estén desarrollando en ese momento. Cabe destacar que un tutor especializado guiará a los alumnos durante esta estancia presencial, asegurándole la realización de un plan de actividades que le permitirán optimizar sus competencias en base a las exigencias del mercado laboral actual.

Este **Máster Semipresencial en Desarrollo de Software** contiene el programa más completo y actualizado del mercado. Sus características más destacadas son:

- Desarrollo de más de 100 casos prácticos presentados por profesionales en Desarrollo del Software
- Sus contenidos gráficos, esquemáticos y eminentemente prácticos con los que están concebidos, recogen una información imprescindible sobre el Desarrollo de Software
- Novedades sobre los últimos avances en el Desarrollo de Software
- Contiene ejercicios prácticos donde realizar el proceso de autoevaluación para mejorar el aprendizaje.
- Todo esto se complementará con lecciones teóricas, preguntas al experto, foros de discusión de temas controvertidos y trabajos de reflexión individual
- Disponibilidad de los contenidos desde cualquier dispositivo fijo o portátil con conexión a internet
- Además, podrás realizar una estancia de prácticas en una de las mejores empresas

Implementarás a tu práctica los procesos de aseguramiento de la calidad para garantizar la fiabilidad y el rendimiento del Software”

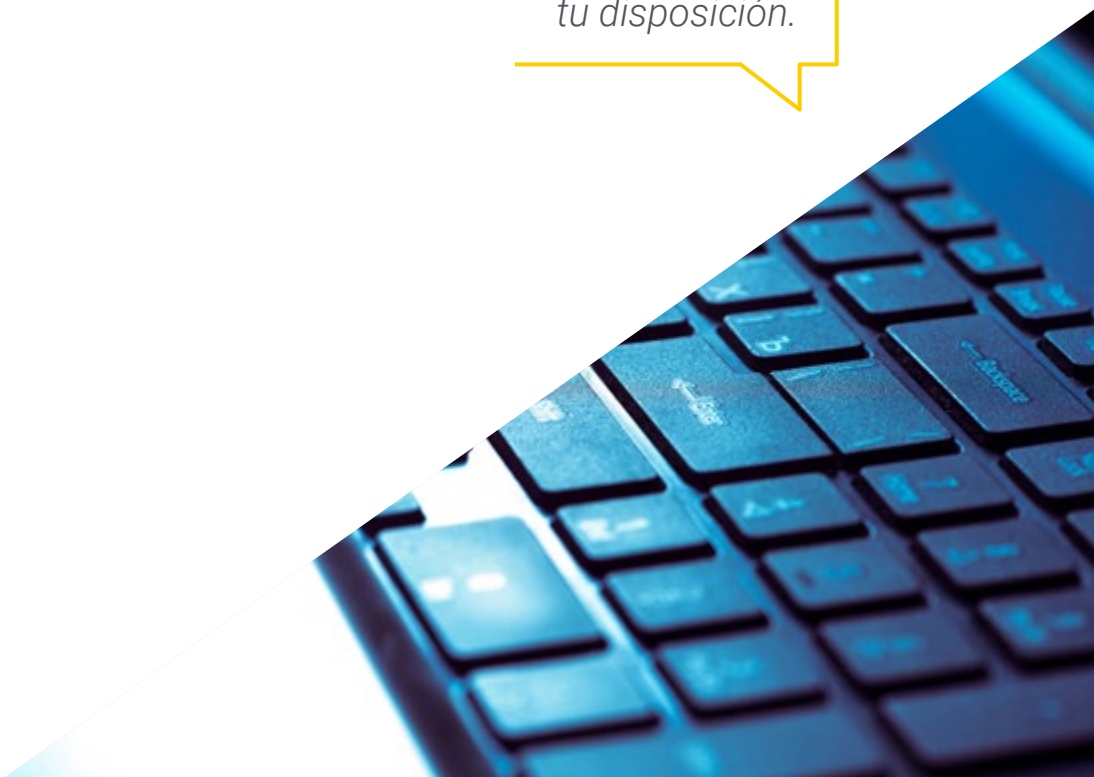
Cursa una estancia intensiva de 3 semanas en una empresa de prestigio y adquiere todo el conocimiento para experimentar un considerable salto de calidad profesional"

En esta propuesta de Máster, de carácter profesionalizante y modalidad semipresencial, el programa está dirigido a la actualización de profesionales de la Informática que quieren incorporar a su praxis las técnicas más avanzadas para el Desarrollo de Software. Los contenidos están basados en la última evidencia científica, y orientados de manera didáctica para integrar el saber teórico en la práctica informática, y los elementos teórico-prácticos facilitarán la actualización del conocimiento.

Gracias a su contenido multimedia elaborado con la última tecnología educativa, permitirán al profesional de la Informática un aprendizaje situado y contextual, es decir, un entorno simulado que proporcionará un aprendizaje inmersivo programado para entrenarse ante situaciones reales. El diseño de este programa está basado en el Aprendizaje Basado en Problemas, mediante el cual deberá tratar de resolver las distintas situaciones de práctica profesional que se le planteen a lo largo del mismo. Para ello, contará con la ayuda de un novedoso sistema de vídeo interactivo realizado por reconocidos expertos.

Esta titulación universitaria incluirá casos reales para acercar al máximo el desarrollo del programa a la realidad de la praxis informática.

Contarás con el apoyo de la mayor institución académica online del mundo, TECH con la última tecnología educativa a tu disposición.



02

¿Por qué cursar este Máster Semipresencial?

La demanda de Desarrolladores de Software está en constante crecimiento, debido al avance tecnológico. De hecho, los expertos prevén que este perfil profesional crezca un 30% durante los próximos años. Ante esto, los informáticos necesitan incorporar en su praxis las metodologías más innovadoras para el desarrollo de aplicaciones en red. Por eso, TECH ha creado esta pionera titulación, donde se combina la actualización más reciente en áreas como la Ingeniería del Software o creación de algoritmos con una estancia práctica en una entidad de referencia. De este modo, los egresados obtendrán competencias avanzadas para ofrecer servicios de elevada calidad.

Un plan de estudios de alta intensidad que sentará las bases de tu crecimiento profesional y te colocará en la cúspide de la Informática"

1. Actualizarse a partir de la última tecnología disponible

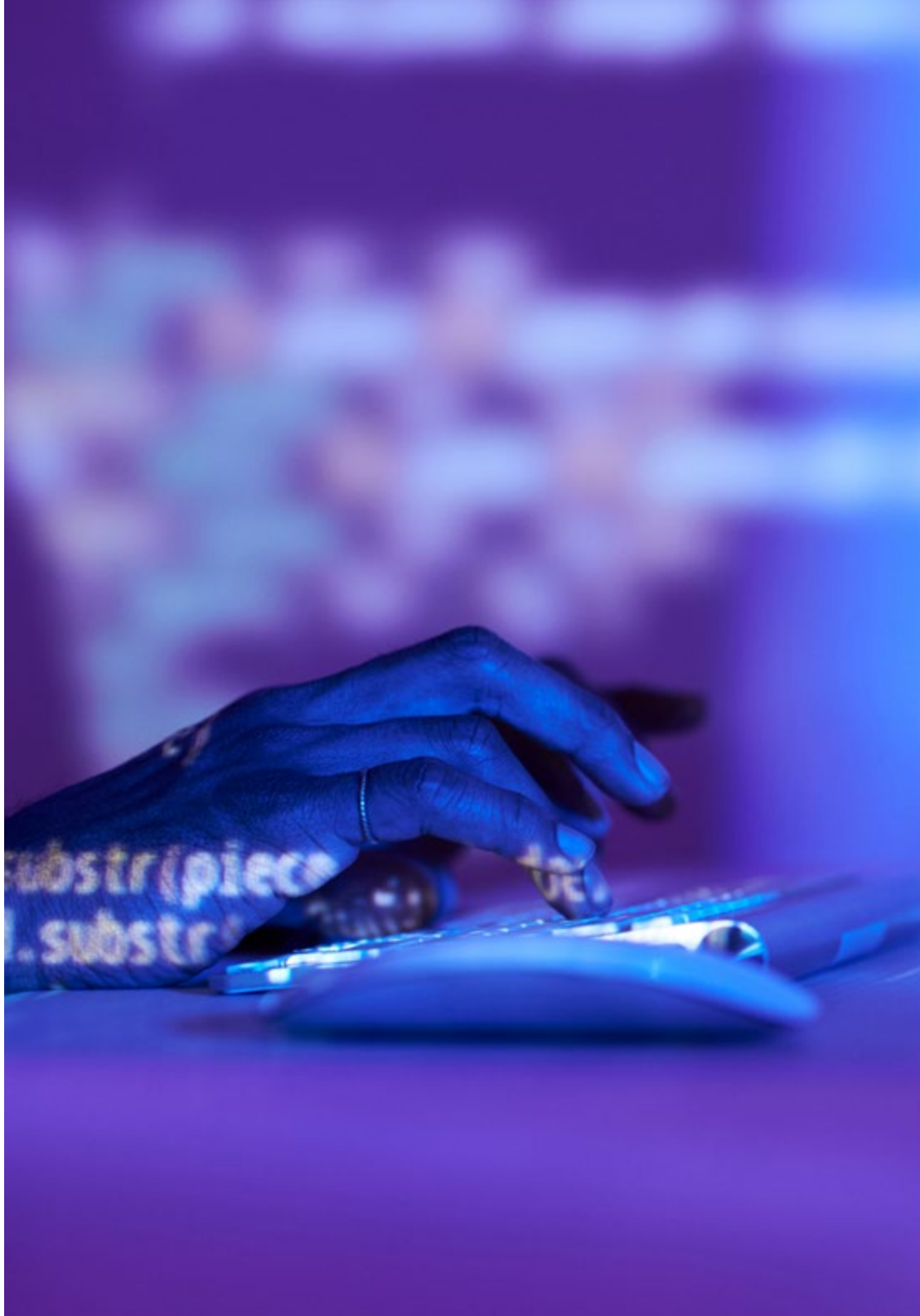
Las nuevas tecnologías están transformando significativamente el Desarrollo de Software, mejorando su productividad, eficiencia y calidad. Estas herramientas permiten a los informáticos abordar desafíos más complejos, crear aplicaciones más robustas y adaptarse rápidamente a las necesidades cambiantes el mercado. En este contexto, TECH presenta este Máster Semipresencial, que pondrá al alcance del alumnado los instrumentos más sofisticados para realizar sus labores con eficacia.

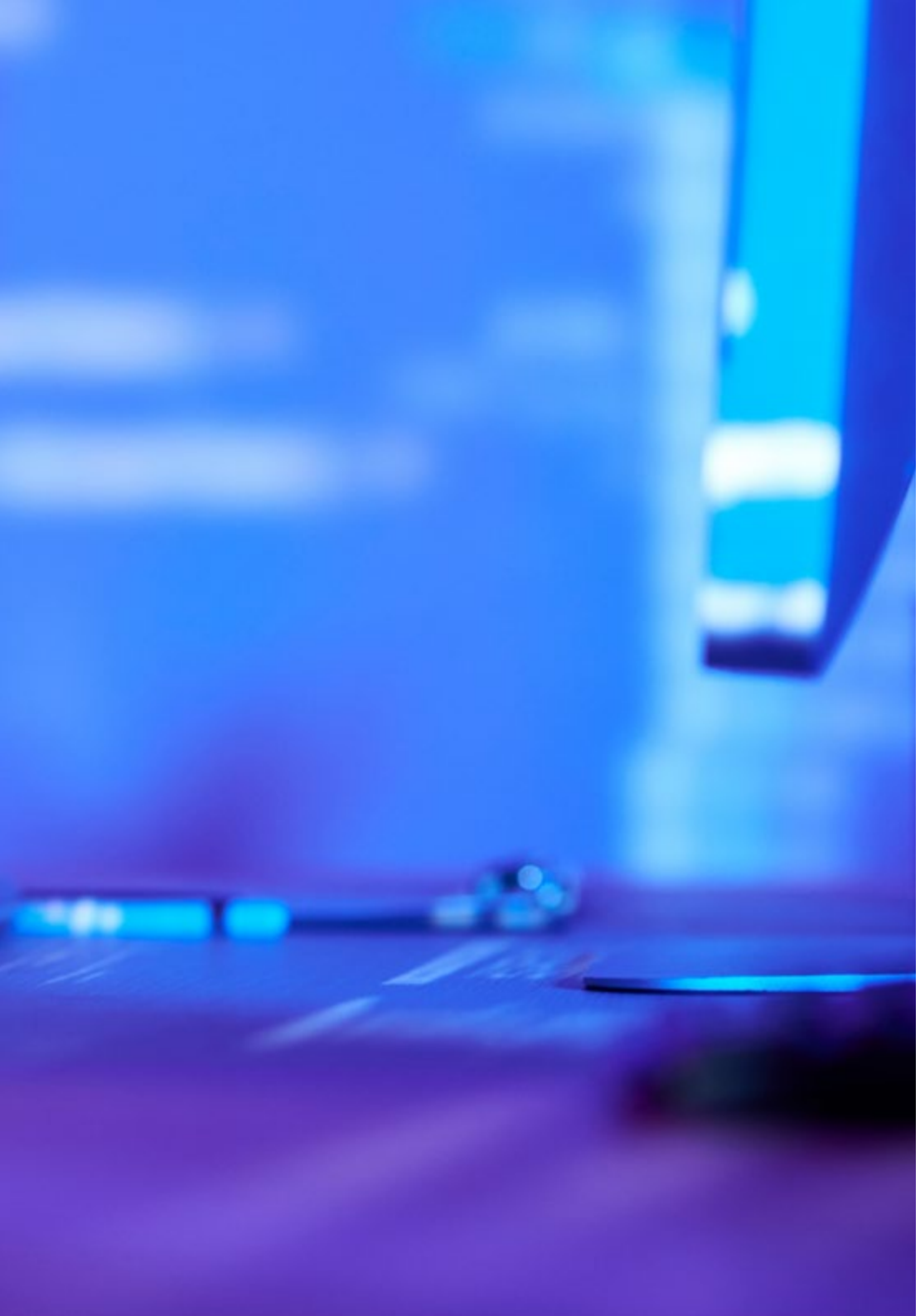
2. Profundizar a partir de la experiencia de los mejores especialistas

Durante todo el período práctico, un equipo integrado por profesionales en Desarrollo de Software acompañará a los alumnos para ayudarles a sacar el máximo partido a esta experiencia académica. Al mismo tiempo, le transmitirán las técnicas más innovadoras para crear las arquitecturas de Software más completas y accesibles.

3. Adentrarse en entornos profesionales de primera

La máxima premisa de TECH es poner a disposición de cualquier persona programas universitarios de primera categoría. Por esta razón, escoge con minuciosidad todos los centros disponibles para la realización de las estancias prácticas. Gracias a este esfuerzo, los informáticos accederán a instituciones de referencia en el campo del Desarrollo de Software. De esta forma, podrá comprobar el día a día de un área de trabajo exigente, rigurosa y exhaustiva, aplicando siempre las últimas técnicas su metodología de trabajo.





4. Combinar la mejor teoría con la práctica más avanzada

El mercado académico está repleto de titulaciones pedagógicas que se limitan a proporcionar contenidos teóricos, olvidando que la práctica es un aspecto fundamental para que los alumnos apliquen los conocimientos a situaciones de trabajo reales. Lejos de esto, TECH ofrece un modelo de aprendizaje 100% práctico, que permitirá a los egresados adquirir experiencia práctica y enfrentarse a los desafíos reales que pueden encontrar en su carrera profesional.

5. Expandir las fronteras del conocimiento

TECH brinda la oportunidad de llevar a cabo este Máster Semipresencial en entidades de envergadura internacional. Gracias a esto, los informáticos podrán expandir sus fronteras y ponerse al día con los mejores profesionales, que ejercen en su labor en compañías de primer nivel. Una oportunidad única que solo TECH, la universidad digital más grande del mundo podría ofrecer.

Tendrás una inmersión práctica total en el centro que tú mismo elijas"

03

Objetivos

Gracias a esta titulación universitaria, los profesionales de la Informática dominarán lenguajes de programación modernos como el C++. Asimismo, los egresados obtendrán habilidades para escribir códigos limpios, eficientes y fáciles de mantener. De este modo, los desarrolladores diseñarán arquitecturas de Software que soporten la escalabilidad, flexibilidad e integridad de los sistemas.

Este programa universitario te brindará las estrategias más vanguardistas para crear Bases de Datos altamente eficientes”

Objetivo general

- El presente Máster Semipresencial en Desarrollo de Software dotará a los informáticos tanto de los conocimientos como habilidades prácticas necesarias para enfrentar los desafíos en este ámbito. Los egresados aplicarán con eficacia patrones de diseño para resolver problemas comunes y construir soluciones informáticas robustas. Asimismo, estarán capacitados para mitigar vulnerabilidades en los programas mediante la implementación de prácticas de desarrollo seguro. En adición, el alumnado creará y gestionará bases de datos relacionales para satisfacer las necesidades de almacenamiento y recuperación de la información

Alcanza tu máximo potencial en el ámbito del Desarrollo de Software gracias a los materiales pedagógicos más completos del mercado académico"

Objetivos específicos

Módulo 1. Fundamentos de programación

- Comprender la estructura básica de un ordenador, el Software y de los lenguajes de programación de propósito general
- Aprender a diseñar e interpretar algoritmos, que son la base necesaria para poder desarrollar programas informáticos
- Entender los elementos esenciales de un programa informático, como son los distintos tipos de datos, operadores, expresiones, sentencias, E/S y sentencias de control
- Comprender las distintas estructuras de datos disponibles en los lenguajes de programación de propósito general tanto estáticas como dinámicas, así como adquirir los conocimientos esenciales para el manejo de ficheros
- Conocer las distintas técnicas de pruebas en los programas informáticos y la importancia de generar una buena documentación junto con un buen código fuente
- Aprender los conceptos básicos del lenguaje de programación C++, uno de los más usados a nivel mundial

Módulo 2. Estructura de datos

- Aprender los fundamentos de la programación en el lenguaje C++, incluyendo clases, variables, expresiones condicionales y objetos
- Entender los tipos abstractos de datos, los tipos de estructuras de datos lineales, estructuras de datos jerárquicas simples y complejas, así como su implementación en C++
- Comprender el funcionamiento de estructuras de datos avanzadas distintas de las habituales
- Conocer la teoría y la práctica relacionada con el uso de montículos y colas de prioridad
- Aprender el funcionamiento de las tablas hash, como tipos abstractos de datos y funciones
- Entender la teoría de grafos, así como algoritmos y concepto avanzados sobre grafos

Módulo 3. Algoritmia y complejidad

- Aprender las principales estrategias de diseño de algoritmos, así como los distintos métodos y medidas para de cálculo de los mismos
- Conocer los principales algoritmos de ordenación usados en el desarrollo de Software
- Entender el funcionamiento de los distintos algoritmos con árboles, *heaps* y grafos
- Comprender el funcionamiento de los algoritmos *greedy*, su estrategia y ejemplos de su uso en los principales problemas conocidos
- Aprenderemos las principales estrategias de búsqueda de caminos mínimos, con el planteamiento de problemas esenciales del ámbito y algoritmos para su resolución
- Entender la técnica de *backtracking* y sus principales usos, así como otras técnicas alternativas

Módulo 4. Bases de datos

- Aprender las distintas aplicaciones y propósitos de los sistemas de bases de datos, así como su funcionamiento y arquitectura
- Comprender el modelo relacional, desde su estructura y operaciones hasta el álgebra relacional extendida
- Aprender en profundidad qué son las bases de datos SQL, su funcionamiento, la definición de datos y la creación de consultas desde las más básicas hasta las más avanzadas y complejas
- Aprender a diseñar bases de datos usando el modelo entidad-relación, a crear diagramas y las características del modelo E-R extendido
- Profundizar en el diseño de bases de datos relacionales, analizando las distintas formas normales y los algoritmos de descomposición
- Sentar las bases para comprender el funcionamiento de las bases de datos NoSQL, así como introducir la base de datos Mongo DB

Módulo 5. Bases de datos avanzadas

- Introducir los distintos sistemas de bases de datos existentes actualmente en el mercado
- Aprender el uso de XML y de bases de datos para la web
- Comprender el funcionamiento de bases de datos avanzadas como son las bases de datos paralelas y las distribuidas
- Conocer la importancia de la indexación y la asociación en los sistemas de bases de datos
- Comprender el funcionamiento del procesamiento transaccional y los sistemas de recuperación
- Adquirir conocimientos relacionados con las bases de datos no relacionales y la Minería de Datos

Módulo 6. Diseño avanzado de algoritmos

- Profundizar en el diseño avanzado de algoritmos, analizando algoritmos recursivos y tipo divide y conquista, así como realizando análisis amortizado
- Comprender los conceptos de programación dinámica y los algoritmos para problemas NP
- Entender el funcionamiento de la optimización combinatoria, así como los distintos algoritmos de aleatorización y algoritmos paralelos
- Conocer y comprender el funcionamiento de los distintos métodos de búsqueda local y con candidatos
- Aprender los mecanismos de verificación de formal de programas y de programas iterativos, incluyendo la lógica de primer orden y el sistema formal de Hoare
- Aprender el funcionamiento de algunos de los principales métodos numéricos como el método de la bisección, el método de Newton Raphson y el método de la secante

Módulo 7. Interacción Persona-ordenador

- Adquirir sólidos conocimientos relacionados con la interacción persona-ordenador y la creación de interfaces usables
- Entender la importancia de la usabilidad de las aplicaciones y por qué hay que tenerlas en cuenta a la hora de diseñar el Software
- Comprender los distintos tipos de diversidad humanas, las limitaciones que suponen y cómo adaptar las interfaces de acuerdo a las necesidades específicas de cada una de ellas
- Aprender el proceso de diseño de interfaces, desde el análisis de requisitos hasta la evaluación, pasando por las distintas etapas intermedias necesarias para llevar realizar una interfaz adecuada
- Conocer las distintas pautas de accesibilidad, los estándares que las estableces y las herramientas que nos permiten evaluarla
- Entender los distintos métodos de interacción con el ordenador, mediante periféricos y dispositivos

Módulo 8. Programación avanzada

- Profundizar en los conocimientos de programación, especialmente en lo relaciona a la programación orientada a objetos, y los distintos tipos de relaciones entre clases existentes
- Conocer los distintos patrones de diseño para problemas orientados a objetos
- Aprender sobre la programación orientada a eventos y el desarrollo de interfaces de usuario con Qt
- Adquirir los conocimientos esenciales de la programación concurrente, los procesos y los hilos

- Aprender a gestionar el uso de los hilos y la sincronización, así como la resolución de los problemas comunes dentro de la programación concurrente
- Entender la importancia de la documentación y las pruebas en el desarrollo del Software

Módulo 9. Desarrollo de aplicaciones en red

- Conocer las características del lenguaje de marcado HTML y su uso en la creación web junto con las hojas de estilo CSS
- Aprender a utilizar el lenguaje de programación orientado al navegador JavaScript, y algunas de sus principales características
- Entender los conceptos de la programación orientada a componentes y de la arquitectura de componentes
- Aprender a usar el *framework* para front-end Bootstrap para el diseño de sitios web
- Entender la estructura del modelo vista controlador en el desarrollo de sitios web dinámicos
- Conocer la arquitectura orientada a servicios y las bases del protocolo HTTP

Módulo 10. Ingeniería del Software

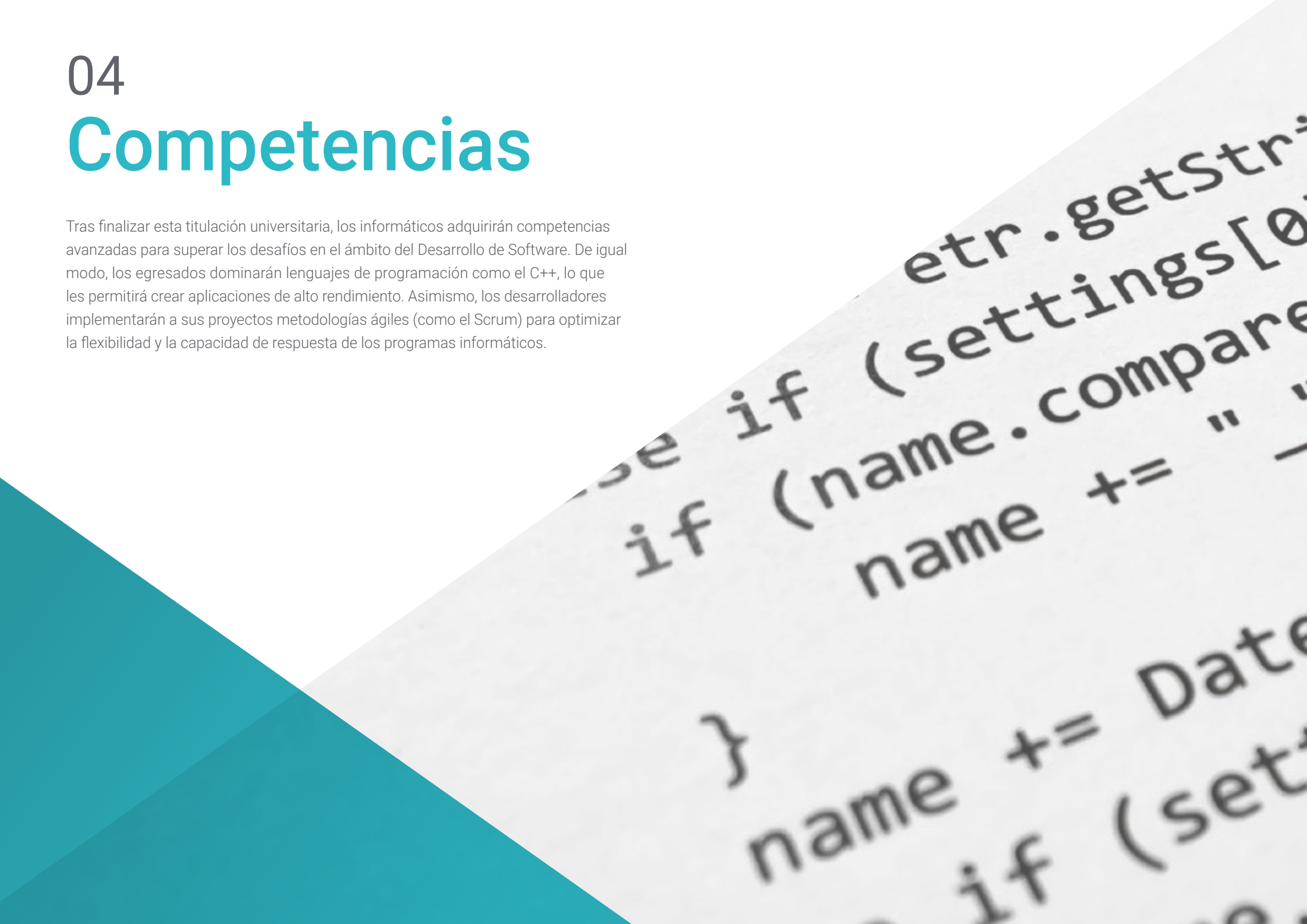
- Sentar las bases de la ingeniería del Software y el modelado, aprendiendo los principales procesos y conceptos
- Entender el proceso del Software y los distintos modelos para su desarrollo incluyendo tecnologías ágiles
- Comprender la ingeniería de requisitos, su desarrollo, elaboración, negociación y validación
- Aprender el modelado de los requisitos y de los distintos elementos como escenarios, información, clases de análisis, flujo, comportamiento y patrones
- Entender los conceptos y procesos del diseño de Software, aprendiendo también sobre el diseño de la arquitectura y sobre el diseño a nivel de componentes y basado en patrones
- Conocer las principales normas relativas a la calidad del software y a la administración de proyectos

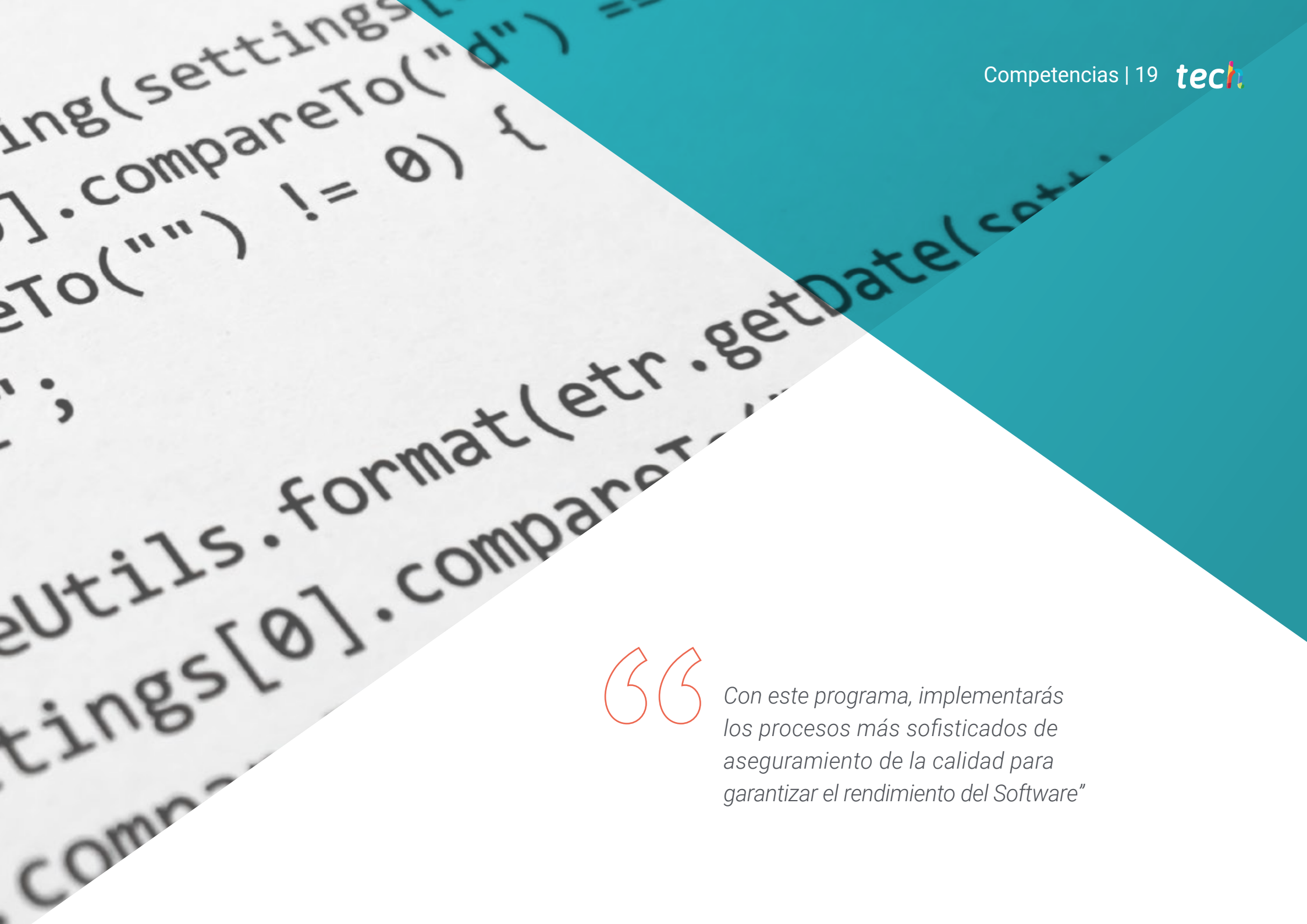
Combinarás teoría y práctica profesional a través de un enfoque educativo exigente y gratificante”

04

Competencias

Tras finalizar esta titulación universitaria, los informáticos adquirirán competencias avanzadas para superar los desafíos en el ámbito del Desarrollo de Software. De igual modo, los egresados dominarán lenguajes de programación como el C++, lo que les permitirá crear aplicaciones de alto rendimiento. Asimismo, los desarrolladores implementarán a sus proyectos metodologías ágiles (como el Scrum) para optimizar la flexibilidad y la capacidad de respuesta de los programas informáticos.





Con este programa, implementarás los procesos más sofisticados de aseguramiento de la calidad para garantizar el rendimiento del Software”

Competencias generales

- Responder a las necesidades actuales del área de desarrollo de Software
- Ser capaz de comprender la estructura básica de un ordenador, el Software y de los lenguajes de programación de propósito general
- Saber aplicar los fundamentos de la programación en el lenguaje C++, incluyendo clases, variables, expresiones condicionales y objetos
- Conocer en profundidad las principales estrategias de diseño de algoritmos, así como los distintos métodos y medidas para de cálculo de los mismos

Esta titulación universitaria te ofrece la oportunidad de ampliar tus conocimientos en escenario real, con el máximo rigor científico de una institución de vanguardia tecnológica"

Competencias específicas

- Conocer las distintas aplicaciones y propósitos de los sistemas de bases de datos, así como su funcionamiento y arquitectura, y aplicarlos en el día a día
- Ser capaz de introducir los distintos sistemas de bases de datos existentes actualmente en el mercado
- Saber analizar algoritmos recursivos y tipo divide y conquista, así como realizar análisis amortizado
- Emplear los conocimientos sobre la interacción persona-ordenador y la creación de interfaces usables en el ejercicio diario de la profesión
- Manejar en profundidad los conocimientos de programación
- Conocer las características del lenguaje de marcado HTML y su uso en la creación web junto con las hojas de estilo CSS

05

Estructura y contenido

Los materiales didácticos que conforman este Máster Semipresencial han sido elaborados por auténticos expertos en el campo del Desarrollo de Software. Compuesto por 10 módulos especializados, el programa equipará a los informáticos una amplia gama de habilidades técnicas. El temario profundizará en aspectos como el Diseño de Algoritmos, creación de Bases de Datos o Ingeniería del Software. Además, el temario brindará a los desarrolladores las técnicas más innovadoras para la optimización de recursos, entre los que destaca el *Backtracking*. De este modo, los egresados adquirirán competencias avanzadas para diseñar arquitecturas de Software que soporten el crecimiento y la evolución de los sistemas.

Dominarás metodologías ágiles como el Scrum para mejorar la eficiencia en el Desarrollo de Software"

Módulo 1. Fundamentos de programación

1.1. Introducción a la programación
- 1.1.1. Estructura básica de un ordenador
- 1.1.2. Software
- 1.1.3. Lenguajes de programación
- 1.1.4. Ciclo de vida de una aplicación informática

1.2. Diseño de algoritmos
- 1.2.1. La resolución de problemas
- 1.2.2. Técnicas descriptivas
- 1.2.3. Elementos y estructura de un algoritmo

1.3. Elementos de un programa
- 1.3.1. Origen y características del lenguaje C++
- 1.3.2. El entorno de desarrollo
- 1.3.3. Concepto de programa
- 1.3.4. Tipos de datos fundamentales
- 1.3.5. Operadores
- 1.3.6. Expresiones
- 1.3.7. Sentencias
- 1.3.8. Entrada y salida de datos

1.4. Sentencias de control
- 1.4.1. Sentencias
- 1.4.2. Bifurcaciones
- 1.4.3. Bucles

1.5. Abstracción y modularidad: funciones
- 1.5.1. Diseño modular
- 1.5.2. Concepto de función y utilidad
- 1.5.3. Definición de una función
- 1.5.4. Flujo de ejecución en la llamada de una función
- 1.5.5. Prototipo de una función
- 1.5.6. Devolución de resultados
- 1.5.7. Llamada a una función: parámetros
- 1.5.8. Paso de parámetros por referencia y por valor
- 1.5.9. Ámbito identificador

1.6. Estructuras de datos estáticas
- 1.6.1. *Arrays*
- 1.6.2. Matrices. Poliedros
- 1.6.3. Búsqueda y ordenación
- 1.6.4. Cadenas. Funciones de E/S para cadenas
- 1.6.5. Estructuras. Uniones
- 1.6.6. Nuevos tipos de datos

1.7. Estructuras de datos dinámicas: punteros
- 1.7.1. Concepto. Definición de puntero
- 1.7.2. Operadores y operaciones con punteros
- 1.7.3. *Arrays* de punteros
- 1.7.4. Punteros y *arrays*
- 1.7.5. Punteros a cadenas
- 1.7.6. Punteros a estructuras
- 1.7.7. Indirección múltiple
- 1.7.8. Punteros a funciones
- 1.7.9. Paso de funciones, estructuras y *arrays* como parámetros de funciones

1.8. Ficheros
- 1.8.1. Conceptos básicos
- 1.8.2. Operaciones con ficheros
- 1.8.3. Tipos de ficheros
- 1.8.4. Organización de los ficheros
- 1.8.5. Introducción a los ficheros C++
- 1.8.6. Manejo de ficheros

1.9. Recursividad
- 1.9.1. Definición de recursividad
- 1.9.2. Tipos de recursión
- 1.9.3. Ventajas e inconvenientes
- 1.9.4. Consideraciones
- 1.9.5. Conversión recursivo iterativa
- 1.9.6. La pila de recursión

1.10. Prueba y documentación
- 1.10.1. Pruebas de programas
- 1.10.2. Prueba de la caja blanca
- 1.10.3. Prueba de la caja negra
- 1.10.4. Herramientas para realizar las pruebas
- 1.10.5. Documentación de programas

Módulo 2. Estructura de datos

2..1. Introducción a la programación en C++
- 2.1.1. Clases, constructores, métodos y atributos
- 2.1.2. Variables
- 2.1.3. Expresiones condicionales y bucles
- 2.1.4. Objetos

2.2. Tipos abstractos de datos (TAD)
- 2.2.1. Tipos de datos
- 2.2.2. Estructuras básicas y TAD
- 2.2.3. Vectores y *Arrays*

2.3. Estructuras de datos lineales
- 2.3.1. TAD Lista definición
- 2.3.2. Listas enlazadas y doblemente enlazadas
- 2.3.3. Listas ordenadas
- 2.3.4. Listas en C++
- 2.3.5. TAD Pila
- 2.3.6. TAD Cola
- 2.3.7. Pila y Cola en C++

2.4. Estructuras de datos jerárquicas
- 2.4.1. TAD Árbol
- 2.4.2. Recorridos
- 2.4.3. Árboles n-arios
- 2.4.4. Árboles binarios
- 2.4.5. Árboles binarios de búsqueda

2.5. Estructuras de datos jerárquicas: árboles complejos
- 2.5.1. Árboles perfectamente equilibrados o de altura mínima
- 2.5.2. Árboles multicamino
- 2.5.3. Referencias bibliográficas

2.6. Montículos y Cola de prioridad
- 2.6.1. TAD Montículos
- 2.6.2. TAD Cola de prioridad

2.7. Tablas Hash
- 2.7.1. TAD Tabla Hash
- 2.7.2. Funciones Hash
- 2.7.3. Función Hash en tablas Hash
- 2.7.4. Redispersión
- 2.7.5. Tablas Hash abiertas

2.8. Grafos
- 2.8.1. TAD Grafo
- 2.8.2. Tipos de Grafo
- 2.8.3. Representación gráfica y operaciones básicas
- 2.8.4. Diseño de Grafo

2.9. Algoritmos y conceptos avanzados sobre Grafos
- 2.9.1. Problemas sobre Grafos
- 2.9.2. Algoritmos sobre caminos
- 2.9.3. Algoritmos de búsqueda o recorridos
- 2.9.4. Otros algoritmos

2.10. Otras estructuras de datos
- 2.10.1. Conjuntos
- 2.10.2. Arrays paralelos
- 2.10.3. Tablas de símbolos
- 2.10.4. Tries

Módulo 3. Algoritmia y complejidad

3.1. Introducción a las estrategias de diseño de algoritmos
- 3.1.1. Recursividad
- 3.1.2. Divide y conquista
- 3.1.3. Otras estrategias

3.2. Eficiencia y análisis de los algoritmos
- 3.2.1. Medidas de eficiencia
- 3.2.2. Medir el tamaño de la entrada
- 3.2.3. Medir el tiempo de ejecución
- 3.2.4. Caso peor, mejor y medio
- 3.2.5. Notación asintónica
- 3.2.6. Criterios de análisis matemático de algoritmos no recursivos
- 3.2.7. Análisis matemático de algoritmos recursivos
- 3.2.8. Análisis empírico de algoritmos

3.3. Algoritmos de ordenación
- 3.3.1. Concepto de ordenación
- 3.3.2. Ordenación de la burbuja
- 3.3.3. Ordenación por selección
- 3.3.4. Ordenación por inserción
- 3.3.5. Ordenación por mezcla (*Merge Sort*)
- 3.3.6. Ordenación rápida (*QuickSort*)

3.4. Algoritmos con árboles
- 3.4.1. Concepto de árbol
- 3.4.2. Árboles binarios
- 3.4.3. Recorridos de árbol
- 3.4.4. Representar expresiones
- 3.4.5. Árboles binarios ordenados
- 3.4.6. Árboles binarios balanceados

3.5. Algoritmos con *Heaps*
- 3.5.1. Los *Heaps*
- 3.5.2. El algoritmo HeapSort
- 3.5.3. Las colas de prioridad

3.6. Algoritmos con grafos
- 3.6.1. Representación
- 3.6.2. Recorrido en anchura
- 3.6.3. Recorrido en profundidad
- 3.6.4. Ordenación topológica

3.7. Algoritmos *Greedy*
- 3.7.1. La estrategia *Greedy*
- 3.7.2. Elementos de la estrategia *Greedy*
- 3.7.3. Cambio de monedas
- 3.7.4. Problema del viajante
- 3.7.5. Problema de la mochila

3.8. Búsqueda de caminos mínimos
- 3.8.1. El problema del camino mínimo
- 3.8.2. Arcos negativos y ciclos
- 3.8.3. Algoritmo de Dijkstra

3.9. Algoritmos *Greedy* sobre Grafos
- 3.9.1. El árbol de recubrimiento mínimo
- 3.9.2. El algoritmo de Prim
- 3.9.3. El algoritmo de Kruskal
- 3.9.4. Análisis de complejidad

3.10. *Backtracking*
- 3.10.1. El *Backtracking*
- 3.10.2. Técnicas alternativas

Módulo 4. Bases de datos

4.1. Aplicaciones y propósitos de los sistemas de base de datos
- 4.1.1. Aplicaciones de los diferentes sistemas de base de datos
- 4.1.2. Propósito en los diferentes sistemas de base de datos
- 4.1.3. Visión de los datos

4.2. Base de datos y arquitectura
- 4.2.1. Base de datos relacionales
- 4.2.2. El diseño de base de datos
- 4.2.3. Bases de datos basadas en objetos y semiestructuradas
- 4.2.4. Almacenamiento de datos y consultas
- 4.2.5. Gestión de transacciones
- 4.2.6. Minería y análisis de datos
- 4.2.7. Arquitectura de las bases de datos

4.3. El modelo relacional: estructura, operaciones y álgebra relacional extendida
- 4.3.1. La estructura de las BD relacionales
- 4.3.2. Operaciones fundamentales en el álgebra relacional
- 4.3.3. Otras operaciones del álgebra relacional
- 4.3.4. Operaciones del álgebra relacional extendida
- 4.3.5. Valores nulos
- 4.3.6. Modificación de la base de datos

4.4. SQL (I)
- 4.4.1. ¿Qué es SQL?
- 4.4.2. La definición de datos
- 4.4.3. Estructura básica de las consultas SQL
- 4.4.4. Operaciones sobre conjuntos
- 4.4.5. Funciones de agregación
- 4.4.6. Valores nulos

4.5. SQL (II)
- 4.5.1. Subconsultas anidadas
- 4.5.2. Consultas complejas
- 4.5.3. Vistas
- 4.5.4. Cursores
- 4.5.5. Consultas complejas
- 4.5.6. Disparadores

4.6. Diseño de base de datos y el modelo E-R
- 4.6.1. Visión general del proceso de diseño
- 4.6.2. El modelo entidad relación
- 4.6.3. Restricciones

4.7. Diagramas entidad relación
- 4.7.1. Diagramas entidad relación
- 4.7.2. Aspectos del diseño entidad relación
- 4.7.3. Conjuntos de entidades débiles

4.8. El modelo entidad relación extendido
- 4.8.1. Características del modelo E-R extendido
- 4.8.2. Diseño de una base de datos
- 4.8.3. Reducción a esquemas relacionales

4.9. Diseño de bases de datos relacionales
- 4.9.1. Características de los buenos diseños relacionales
- 4.9.2. Dominios atómicos y la primera forma normal (1FN)
- 4.9.3. Descomposición mediante dependencias funcionales
- 4.9.4. Teoría de las dependencias funcionales
- 4.9.5. Algoritmos de descomposición
- 4.9.6. Descomposición mediante dependencias multivaloradas
- 4.9.7. Más formas normales
- 4.9.8. Proceso de diseño de las bases de datos

4.10. Bases de datos NoSQL
- 4.10.1. ¿Qué son las bases de datos NoSQL?
- 4.10.2. Análisis de las diferentes opciones de NoSQL y sus características
- 4.10.3. MongoDB

Módulo 5. Bases de datos avanzadas

5.1. Introducción a los diferentes sistemas de bases de datos
- 5.1.1. Repaso histórico
- 5.1.2. Bases de datos jerárquicas
- 5.1.3. Bases de datos red
- 5.1.4. Bases de datos relacionales
- 5.1.5. Bases de datos no relacionales

5.2. XML y bases de datos para la web
- 5.2.1. Validación de documentos XML
- 5.2.2. Transformaciones de documentos XML
- 5.2.3. Almacenamiento de datos XML
- 5.2.4. Bases de datos relacionales XML
- 5.2.5. SQL/XML
- 5.2.6. Bases de datos nativas XML

5.3. Bases de datos paralelas
- 5.3.1. Sistemas paralelos
- 5.3.2. Arquitecturas paralelas de bases de datos
- 5.3.4. Paralelismo en consultas
- 5.3.5. Paralelismo entre consultas
- 5.3.6. Diseño de sistemas paralelos
- 5.3.7. Procesamiento paralelo en SQL

5.4. Bases de datos distribuidas
- 5.4.1. Sistemas distribuidos
- 5.4.2. Almacenamiento distribuido
- 5.4.3. Disponibilidad
- 5.4.4. Procesamiento distribuido de consultas
- 5.4.5. Proveedores de bases de datos distribuidas

5.5. Indexación y asociación
- 5.5.1. Índices ordenados
- 5.5.2. Índices densos y dispersos
- 5.5.3. Índices multinivel
- 5.5.4. Actualización del índice
- 5.5.5. Asociación estática
- 5.5.6. Cómo usar índices en bases de datos

5.6. Introducción al procesamiento transaccional
- 5.6.1. Estados de una transacción
- 5.6.2. Implementación de la atomicidad y durabilidad
- 5.6.3. Secuencialidad
- 5.6.4. Recuperabilidad
- 5.6.5. Implementación del aislamiento

5.7. Sistemas de recuperación
- 5.7.1. Clasificación de fallos
- 5.7.2. Estructuras de almacenamiento
- 5.7.3. Recuperación y atomicidad
- 5.7.4. Recuperación basada en registro histórico
- 5.7.5. Transacciones concurrentes y recuperación
- 5.7.6. Alta disponibilidad en bases de datos

5.8. Ejecución y procesamiento de consultas
- 5.8.1. Coste de una consulta
- 5.8.2. Operación de selección
- 5.8.3. Ordenación
- 5.8.4. Introducción a la optimización de consultas
- 5.8.5. Monitorización del rendimiento

5.9. Bases de datos no relacionales
- 5.9.1. Bases de datos orientadas a documentos
- 5.9.2. Bases de datos orientadas a Grafos
- 5.9.3. Bases de datos clave-valor

5.10. Data Warehouse, OLAP y minería de datos
- 5.10.1. Componentes de los almacenes de datos
- 5.10.2. Arquitectura de un data Warehouse
- 5.10.3. OLAP
- 5.10.4. Funcionalidades de la Minería de Datos
- 5.10.5. Otros tipos de minería

Módulo 6. Diseño avanzado de algoritmos

6.1. Análisis de algoritmos recursivos y tipo divide y conquista
- 6.1.1. Planteamiento y resolución de ecuaciones de recurrencia homogéneas y no homogéneas
- 6.1.2. Descripción general de la estrategia divide y conquista

6.2. Análisis amortizado
- 6.2.1. El análisis agregado
- 6.2.2. El método de contabilidad
- 6.2.3. El método del potencial

6.3. Programación dinámica y algoritmos para problemas NP
- 6.3.1. Características de la programación dinámica
- 6.3.2. Vuelta atrás: *Backtracking*
- 6.3.3. Ramificación y poda

6.4. Optimización combinatoria
- 6.4.1. Representación de problemas
- 6.4.2. Optimización en 1D

6.5. Algoritmos de aleatorización
- 6.5.1. Ejemplos de algoritmos de aleatorización
- 6.5.2. El teorema Buffon
- 6.5.3. Algoritmo de Monte Carlo
- 6.5.4. Algoritmo Las Vegas

6.6. Búsqueda local y con candidatos
- 6.6.1. *Gradient Ascent*
- 6.6.2. *Hill Climbing*
- 6.6.3. *Simulated Annealing*
- 6.6.4. *Tabu Search*
- 6.6.5. Búsqueda con candidatos

6.7. Verificación formal de programas
- 6.7.1. Especificación de abstracciones funcionales
- 6.7.2. El lenguaje de la lógica de primer orden
- 6.7.3. El sistema formal de Hoare

6.8. Verificación de programas iterativos
- 6.8.1. Reglas del sistema formal de Hoare
- 6.8.2. Concepto de invariante de iteraciones

6.9. Métodos numéricos
- 6.9.1. El método de la bisección
- 6.9.2. El método de Newton Raphson
- 6.9.3. El método de la secante

6.10. Algoritmos paralelos
- 6.10.1. Operaciones binarias paralelas
- 6.10.2. Operaciones paralelas con grafos
- 6.10.3. Paralelismo en divide y vencerás
- 6.10.4. Paralelismo en programación dinámica

Módulo 7. Interacción Persona ordenador

7.1. Introducción a la interacción persona ordenador
- 7.1.1. Qué es la interacción persona ordenador
- 7.1.2. Relación de la interacción persona ordenador con otras disciplinas
- 7.1.3. La interfaz de usuario
- 7.1.4. Usabilidad y accesibilidad
- 7.1.5. Experiencia de usuario y diseño centrado en el usuario

7.2. El ordenador y la interacción: interfaz de usuario y paradigmas de interacción
- 7.2.1. La interacción
- 7.2.2. Paradigmas y estilos de interacción
- 7.2.3. Evolución de las interfaces de usuario
- 7.2.4. Interfaces de usuario clásicas: WIMP/GUI, comandos, voz, realidad virtual
- 7.2.5. Interfaces de usuario innovadoras: móviles, portátiles, colaborativas, BCI

7.3. El factor humano: aspectos psicológicos y cognitivos
- 7.3.1. La importancia del factor humano en la interacción
- 7.3.2. El procesamiento humano de información
- 7.3.3. La entrada y salida de la información: visual, auditiva y táctil
- 7.3.4. Percepción y atención
- 7.3.5. Conocimiento y modelos mentales: representación, organización y adquisición

7.4. El factor humano: limitaciones sensoriales y físicas
- 7.4.1. Diversidad funcional, discapacidad y deficiencia
- 7.4.2. Diversidad visual
- 7.4.3. Diversidad auditiva
- 7.4.4. Diversidad cognitiva
- 7.4.5. Diversidad motórica
- 7.4.6. El caso de los inmigrantes digitales

7. 5. El proceso de diseño (I): análisis de requisitos para el diseño de la interfaz de usuario
- 7.5.1. Diseño centrado en el usuario
- 7.5.2. Qué es el análisis de requisitos
- 7.5.3. La recogida de información
- 7.5.4. Análisis e interpretación de la información
- 7.5.5. Análisis de la usabilidad y la accesibilidad

7.6. El proceso de diseño (II): prototipado y análisis de tareas
- 7.6.1. Diseño conceptual
- 7.6.2. Prototipado
- 7.6.3. Análisis jerárquico de tareas

7.7. El proceso de diseño (III): la evaluación
- 7.7.1. Evaluación en el proceso de diseño: objetivos y métodos
- 7.7.2. Métodos de evaluación sin usuarios
- 7.7.3. Métodos de evaluación con usuarios
- 7.7.4. Estándares y normas de evaluación

7.8. Accesibilidad: definición y pautas
- 7.8.1. Accesibilidad y diseño universal
- 7.8.2. La iniciativa WAI y las pautas WCAG
- 7.8.3. Pautas WCAG 2.0 y 2.1

7.9. Accesibilidad: evaluación y diversidad funcional
- 7.9.1. Herramientas de evaluación de la accesibilidad en la web
- 7.9.2. Accesibilidad y diversidad funcional

7.10. El ordenador y la interacción: periféricos y dispositivos
- 7.10.1. Dispositivos y periféricos tradicionales
- 7.10.2. Dispositivos y periféricos alternativos
- 7.10.3. Móviles y tabletas
- 7.10.4. Diversidad funcional, interacción y periféricos

Módulo 8. Programación avanzada

8.1. Introducción a la programación orientada a objetos
- 8.1.1. Introducción a la programación orientada a objetos
- 8.1.2. Diseño de clases
- 8.1.3. Introducción a UML para el modelado de los problemas

8.2. Relaciones entre clases
- 8.2.1. Abstracción y herencia
- 8.2.2. Conceptos avanzados de herencia
- 8.2.3. Polimorfismo
- 8.2.4. Composición y agregación

8.3. Introducción a los patrones de diseño para problemas orientados a objetos
- 8.3.1. Qué son los patrones de diseño
- 8.3.2. Patrón *Factory*
- 8.3.4. Patrón *Singleton*
- 8.3.5. Patrón *Observer*
- 8.3.6. Patrón *Composite*

8.4. Excepciones
- 8.4.1. ¿Qué son las excepciones?
- 8.4.2. Captura y gestión de excepciones
- 8.4.3. Lanzamiento de excepciones
- 8.4.4. Creación de excepciones

8.5. Interfaces de usuarios
- 8.5.1. Introducción a Qt
- 8.5.2. Posicionamiento
- 8.5.3. ¿Qué son los eventos?
- 8.5.4. Eventos: definición y captura
- 8.5.5. Desarrollo de interfaces de usuario

8.6. Introducción a la programación concurrente
- 8.6.1. Introducción a la programación concurrente
- 8.6.2. El concepto de proceso e hilo
- 8.6.3. Interacción entre procesos o hilos
- 8.6.4. Los hilos en C++
- 8.6.6. Ventajas e inconvenientes de la programación concurrente

8.7. Gestión de hilos y sincronización
- 8.7.1. Ciclo de vida de un hilo
- 8.7.2. La clase *Thread*
- 8.7.3. Planificación de hilos
- 8.7.4. Grupos hilos
- 8.7.5. Hilos de tipo demonio
- 8.7.6. Sincronización
- 8.7.7. Mecanismos de bloqueo
- 8.7.8. Mecanismos de comunicación
- 8.7.9. Monitores

8.8. Problemas comunes dentro de la programación concurrente
- 8.8.1. El problema de los productores consumidores
- 8.8.2. El problema de los lectores y escritores
- 8.8.3. El problema de la cena de los filósofos

8.9. Documentación y pruebas de software
- 8.9.1. ¿Por qué es importante documentar el software?
- 8.9.2. Documentación de diseño
- 8.9.3. Uso de herramientas para la documentación

8.10. Pruebas de software
- 8.10.1. Introducción a las pruebas del software
- 8.10.2. Tipos de pruebas
- 8.10.3. Prueba de unidad
- 8.10.4. Prueba de integración
- 8.10.5. Prueba de validación
- 8.10.6. Prueba del sistema

Módulo 9. Desarrollo de aplicaciones en red

9.1. Lenguajes de marcado HTML5
- 9.1.1. Conceptos básicos de HTML
- 9.1.2. Nuevos elementos HTML 5
- 9.1.3. Formularios: nuevos controles

9.2. Introducción a hojas de estilo CSS
- 9.2.1. Primeros pasos con CSS
- 9.2.2. Introducción a CSS3

9.3. Lenguaje script de navegador: JavaScript
- 9.3.1. Conceptos básicos de JavaScript
- 9.3.2. DOM
- 9.3.3. Eventos
- 9.3.4. JQuery
- 9.3.5. Ajax

9.4. Concepto de la programación orientada a componentes
- 9.4.1. Contexto
- 9.4.2. Componentes e interfaces
- 9.4.3. Estados de un componente

9.5. Arquitectura de componentes
- 9.5.1. Arquitecturas actuales
- 9.5.2. Integración y despliegue de componentes

9.6. *Framework Frontend*: Bootstrap
- 9.6.1. Diseño con rejilla
- 9.6.2. Formularios
- 9.6.3. Componentes

9.7. Modelo vista controlador
- 9.7.1. Métodos de desarrollo Web
- 9.7.2. Patrón de diseño: MVC

9.8. Tecnologías Grid de la información
- 9.8.1. Incremento de recursos en computación
- 9.8.2. Concepto de tecnología Grid

9.9. Arquitectura orientada a servicios
- 9.9.1. SOA y servicios web
- 9.9.2. Topología de un servicio web
- 9.9.3. Plataformas para los servicios web

9.10. Protocolo HTTP
- 9.10.1. Mensajes
- 9.10.2. Sesiones persistentes
- 9.10.3. Sistema criptográfico
- 9.10.4. Funcionamiento del protocolo HTTPS

Módulo 10. Ingeniería del Software

10.1. Introducción a la Ingeniería del Software y al modelado
- 10.1.1. La naturaleza del software
- 10.1.2. La naturaleza única de las WebApps
- 10.1.3. Ingeniería del Software
- 10.1.4. El proceso del Software
- 10.1.5. La práctica de la Ingeniería del Software
- 10.1.6. Mitos del Software
- 10.1.7. Cómo comienza todo
- 10.1.8. Conceptos orientados a objetos
- 10.1.9. Introducción a UML

10.2. El proceso del Software
- 10.2.1. Un modelo general de proceso
- 10.2.2. Modelos de proceso prescriptivos
- 10.2.3. Modelos de proceso especializado
- 10.2.4. El proceso unificado
- 10.2.5. Modelos del proceso personal y del equipo
- 10.2.6. ¿Qué es la agilidad?
- 10.2.7. ¿Qué es un proceso ágil?
- 10.2.8. Scrum
- 10.2.9. Conjunto de herramientas para el proceso ágil

10.3. Principios que guían la práctica de la Ingeniería del Software
- 10.3.1. Principios que guían el proceso
- 10.3.2. Principios que guían la práctica
- 10.3.3. Principios de comunicación
- 10.3.4. Principios de planificación
- 10.3.5. Principios de modelado
- 10.3.6. Principios de construcción
- 10.3.7. Principios de despliegue

10.4. Comprensión de los requisitos
- 10.4.1. Ingeniería de requisitos
- 10.4.2. Establecer las bases
- 10.4.3. Indagación de los requisitos
- 10.4.4. Desarrollo de casos de uso
- 10.4.5. Elaboración del modelo de los requisitos
- 10.4.6. Negociación de los requisitos
- 10.4.7. Validación de los requisitos

10.5. Modelado de los requisitos: escenarios, información y clases de análisis
- 10.5.1. Análisis de los requisitos
- 10.5.2. Modelado basado en escenarios
- 10.5.3. Modelos UML que proporcionan el caso de uso
- 10.5.4. Conceptos de modelado de datos
- 10.5.5. Modelado basado en clases
- 10.5.6. Diagramas de clases

10.6. Modelado de los requisitos: flujo, comportamiento y patrones
- 10.6.1. Requisitos que modelan las estrategias
- 10.6.2. Modelado orientado al flujo
- 10.6.3. Diagramas de estado
- 10.6.4. Creación de un modelo de comportamiento
- 10.6.5. Diagramas de secuencia
- 10.6.6. Diagramas de comunicación
- 10.6.7. Patrones para el modelado de requisitos

10.7. Conceptos de diseño
- 10.7.1. Diseño en el contexto de la ingeniería del software
- 10.7.2. El proceso de diseño
- 10.7.3. Conceptos de diseño
- 10.7.4. Conceptos de diseño orientado a objetos
- 10.7.5. El modelo del diseño

10.8. Diseño de la arquitectura
- 10.8.1. Arquitectura del software
- 10.8.2. Géneros arquitectónicos
- 10.8.3. Estilos arquitectónicos
- 10.8.4. Diseño arquitectónico
- 10.8.5. Evolución de los diseños alternativos para la arquitectura
- 10.8.6. Mapeo de la arquitectura con el uso del flujo de datos

10.9. Diseño en el nivel de componentes y basado en patrones
- 10.9.1. ¿Qué es un componente?
- 10.9.2. Diseño de componentes basados en clase
- 10.9.3. Realización del diseño en el nivel de componentes
- 10.9.4. Diseño de componentes tradicionales
- 10.9.5. Desarrollo basado en componentes
- 10.9.6. Patrones de diseño
- 10.9.7. Diseño de Software basado en patrones
- 10.9.8. Patrones arquitectónicos
- 10.9.9. Patrones de diseño en el nivel de componentes
- 10.9.10. Patrones de diseño de la interfaz de usuario

10.10. Calidad del sSftware y administración de proyectos
- 10.10.1. Calidad del Software
- 10.10.2. El dilema de la calidad del Software
- 10.10.3. Lograr la calidad del Software
- 10.10.4. Aseguramiento de la calidad del Software
- 10.10.5. El espectro administrativo
- 10.10.6. El personal
- 10.10.7. El producto
- 10.10.8. El proceso
- 10.10.9. El proyecto
- 10.10.10. Principios y prácticas

06

Prácticas

Una vez superado el período teórico online, este programa universitario prevé una fase de capacitación práctica en una entidad de referencia. Durante este período, los egresados contarán con el apoyo de un tutor que les ayudará a sacarle el máximo provecho a esta experiencia. Gracias a esto, los informáticos adquirirán competencias avanzadas para experimentar un notable salto de calidad en el ejercicio de su profesión.

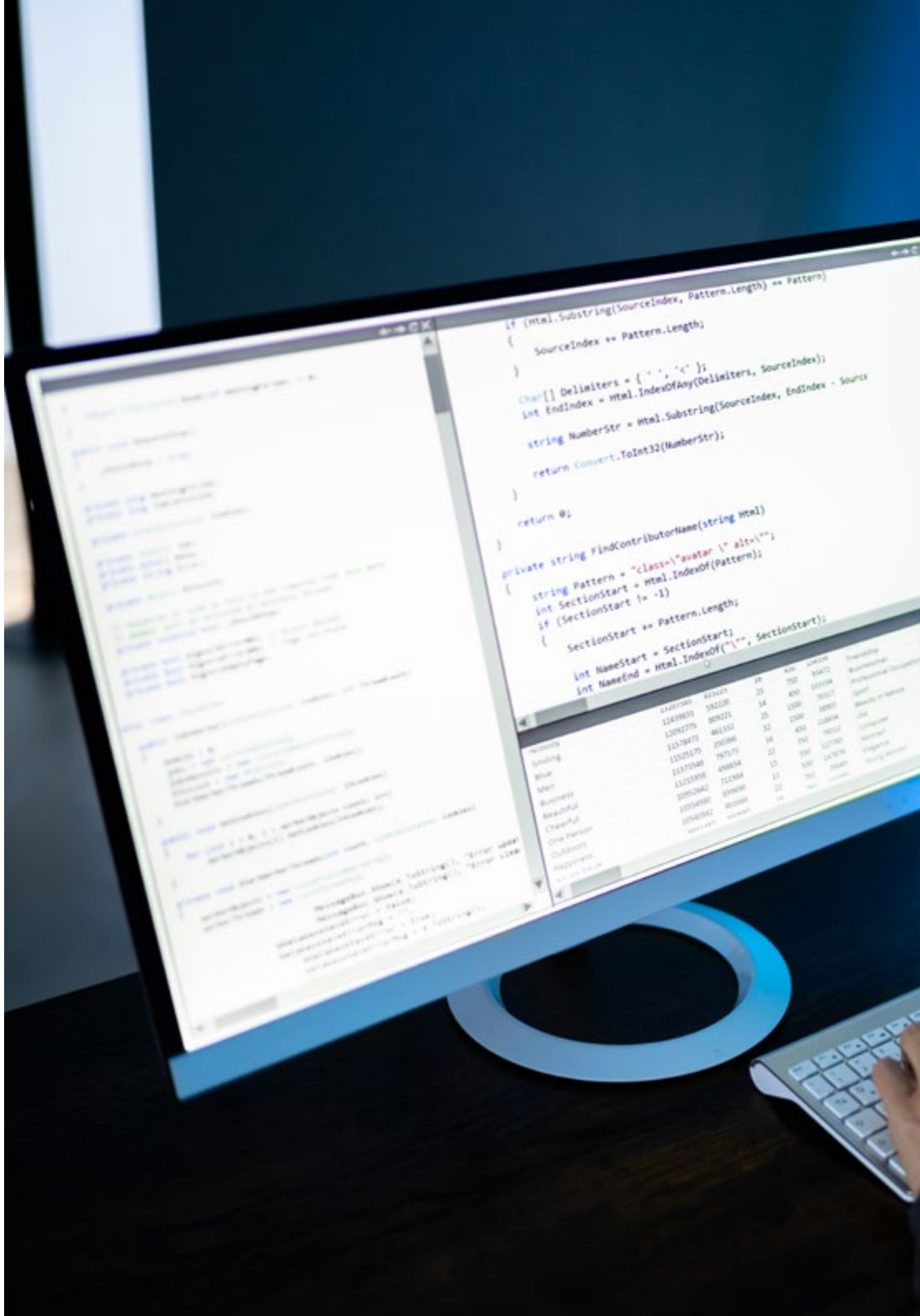
Mediante esta titulación universitaria, estarás preparado para superar los desafíos en el campo del Desarrollo de Software"

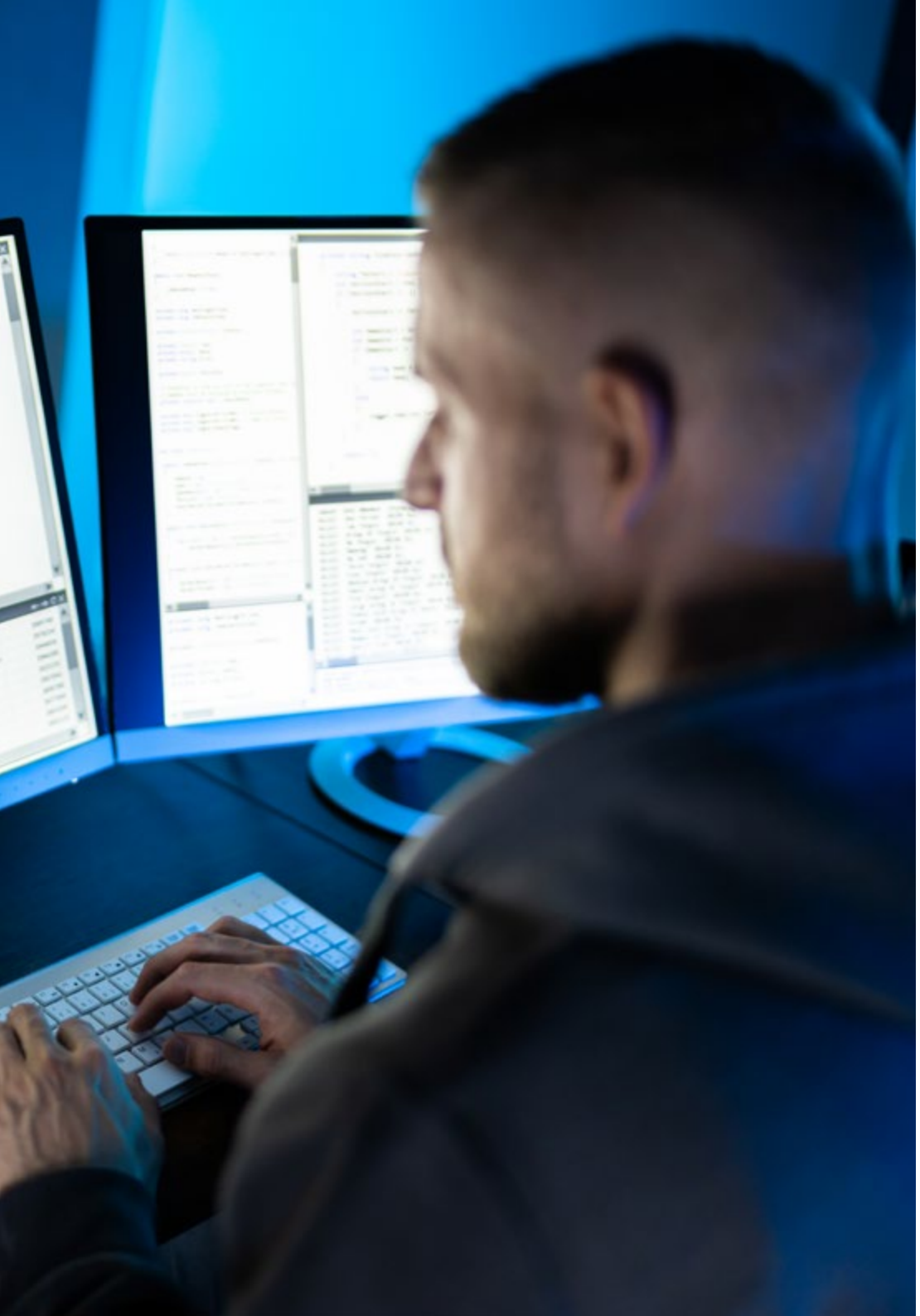
El período de Capacitación Práctica de este programa en Desarrollo de Software está conformada por una estancia práctica intensiva en una prestigiosa empresa, de 3 semanas de duración, de lunes a viernes y con jornadas de 8 horas consecutivas de capacitación práctica, al lado de un especialista adjunto. Así, esta estancia les permitirá a los profesionales aplicar los conceptos teóricos aprendidos en situaciones reales de trabajo, ya sea en empresas del sector o en proyectos colaborativos

En esta propuesta de capacitación, de carácter completamente práctica, las actividades están dirigidas al desarrollo y perfeccionamiento de las competencias necesarias para brindar servicios de Desarrollo de Software, así como condiciones que requieren un alto nivel de cualificación, orientadas a la capacitación específica para el ejercicio de la actividad.

Sin duda, se trata de una excelente oportunidad para que los egresados se sumerjan en la realidad de una profesión llena de desafíos. De esta forma, colaborarán en proyectos relacionados con el diseño de Software, creación de bases de datos o elaboración de algoritmos, entre otros. Así pues, los informáticos desarrollarán competencias avanzadas para optimizar su praxis y elevar sus horizontes profesionales.

La parte práctica se realizará con la participación activa del estudiante desempeñando las actividades y procedimientos de cada área de competencia (aprender a aprender y aprender a hacer), con el acompañamiento y guía de los profesores y demás compañeros de entrenamiento que faciliten el trabajo en equipo y la integración multidisciplinar como competencias transversales para la praxis del Desarrollo de Software (aprender a ser y aprender a relacionarse).





Los procedimientos descritos a continuación serán la base de la parte práctica de la capacitación, y su realización estará sujeta a la disponibilidad propia del centro y su volumen de trabajo, siendo las actividades propuestas las siguientes:

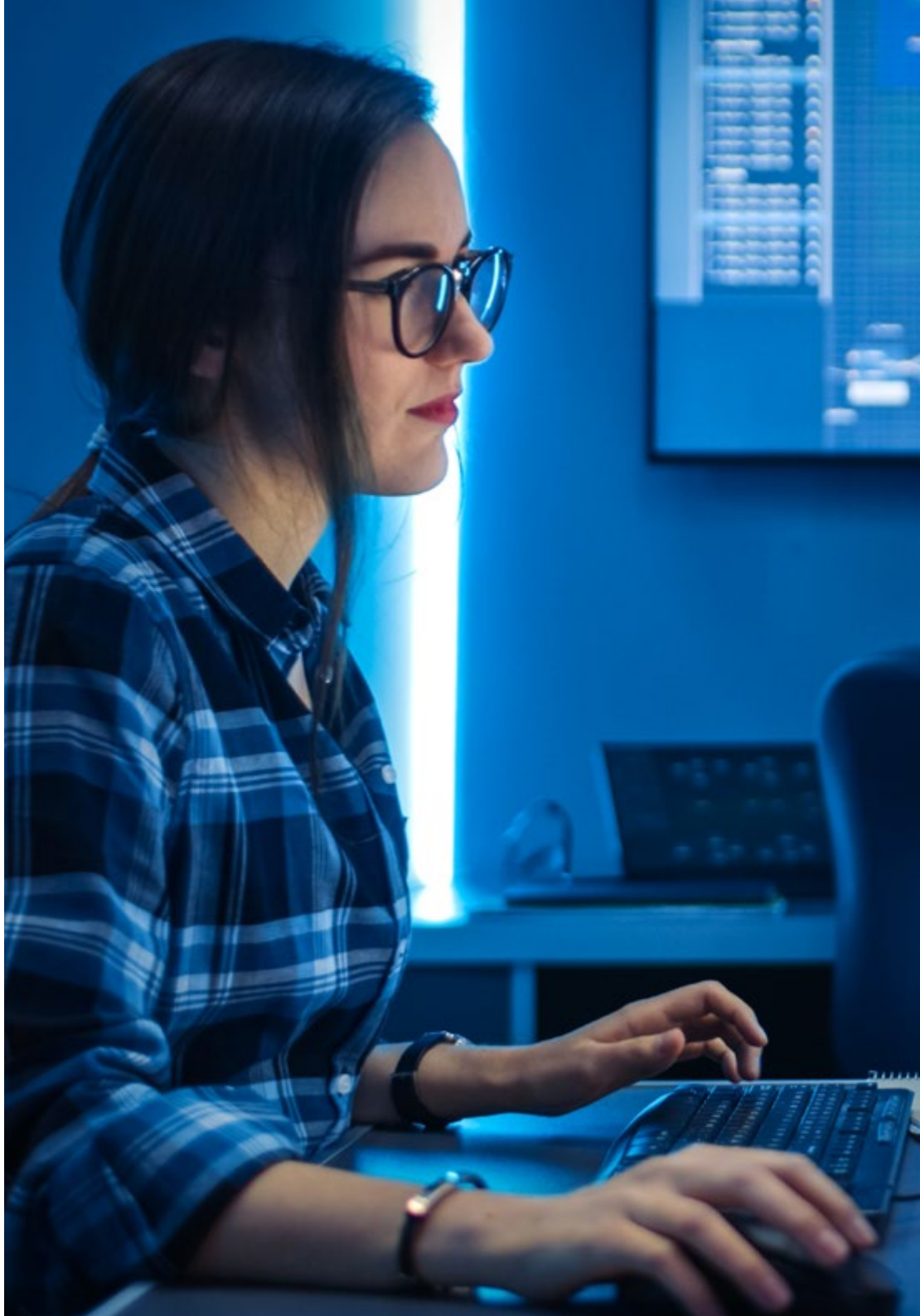
Módulo	Actividad Práctica
Arquitectura de datos	Crear nuevas estructuras de datos que sean eficientes y adecuadas para resolver problemas específicos
	Implementar estructuras de datos básicas como pulas, colas, árboles, grafos, etc.
	Evaluar la complejidad temporal de diferentes estructuras de algoritmos
	Realizar esquemas de bases de datos eficientes empleando sistemas de datos que optimicen el almacenamiento y la recuperación de datos
Técnicas de Algoritmos	Diseñar algoritmos en diferentes lenguajes de programación
	Utilizar técnicas avanzadas como la Programación Dinámica y Algoritmos de Grafos
	Desarrollar algoritmia que minimice el empleo de recursos computacionales como memoria y tiempo de CPU
	Probar algoritmos para verificar su correcto funcionamiento en diferentes escenarios y con diversos conjuntos de datos
Sistemas de Almacenamiento de Datos	Configurar bases de datos en sistemas de gestión como MySQL, PostgreSQL, etc.
	Utilizar herramientas de monitoreo para supervisar el rendimiento de la base de datos, con el fin de asegurar su fiabilidad y disponibilidad
	Aplicar controles de acceso y políticas de seguridad a fin de proteger los datos contra accesos no autorizados
	Ejecutar la migración de bases de datos de un entorno a otro (por ejemplo, de una base de datos local a una en la nube)
Desarrollo de Software	Llevar a cabo la arquitectura del sistema, incluyendo la división en módulos y la especificación de interfaces
	Elaborar diseños detallados de cada componente o módulo del sistema, incluyendo diagramas UML y especificaciones técnicas
	Efectuar pruebas unitarias para verificar el correcto funcionamiento de los módulos de código individuales
	Identificar y corregir errores en el Software después de su despliegue, para implementar nuevas funcionalidades o mejoras

Seguro de responsabilidad civil

La máxima preocupación de esta institución es garantizar la seguridad tanto de los profesionales en prácticas como de los demás agentes colaboradores necesarios en los procesos de capacitación práctica en la empresa. Dentro de las medidas dedicadas a lograrlo, se encuentra la respuesta ante cualquier incidente que pudiera ocurrir durante todo el proceso de enseñanza-aprendizaje.

Para ello, esta entidad educativa se compromete a contratar un seguro de responsabilidad civil que cubra cualquier eventualidad que pudiera surgir durante el desarrollo de la estancia en el centro de prácticas.

Esta póliza de responsabilidad civil de los profesionales en prácticas tendrá coberturas amplias y quedará suscrita de forma previa al inicio del periodo de la capacitación práctica. De esta forma el profesional no tendrá que preocuparse en caso de tener que afrontar una situación inesperada y estará cubierto hasta que termine el programa práctico en el centro.



Condiciones generales de la capacitación práctica

Las condiciones generales del acuerdo de prácticas para el programa serán las siguientes:

1. TUTORÍA: durante el Máster Semipresencial el alumno tendrá asignados dos tutores que le acompañarán durante todo el proceso, resolviendo las dudas y cuestiones que pudieran surgir. Por un lado, habrá un tutor profesional perteneciente al centro de prácticas que tendrá como fin orientar y apoyar al alumno en todo momento. Por otro lado, también tendrá asignado un tutor académico cuya misión será la de coordinar y ayudar al alumno durante todo el proceso resolviendo dudas y facilitando todo aquello que pudiera necesitar. De este modo, el profesional estará acompañado en todo momento y podrá consultar las dudas que le surjan, tanto de índole práctica como académica.

2. DURACIÓN: el programa de prácticas tendrá una duración de tres semanas continuadas de formación práctica, distribuidas en jornadas de 8 horas y cinco días a la semana. Los días de asistencia y el horario serán responsabilidad del centro, informando al profesional debidamente y de forma previa, con suficiente tiempo de antelación para favorecer su organización.

3. INASISTENCIA: en caso de no presentarse el día del inicio del Máster Semipresencial, el alumno perderá el derecho a la misma sin posibilidad de reembolso o cambio de fechas. La ausencia durante más de dos días a las prácticas sin causa justificada/médica, supondrá la renuncia las prácticas y, por tanto, su finalización automática. Cualquier problema que aparezca durante el transcurso de la estancia se tendrá que informar debidamente y de forma urgente al tutor académico.

4. CERTIFICACIÓN: el alumno que supere el Máster Semipresencial recibirá un certificado que le acreditará la estancia en el centro en cuestión.

5. RELACIÓN LABORAL: el Máster Semipresencial no constituirá una relación laboral de ningún tipo.

6. ESTUDIOS PREVIOS: algunos centros podrán requerir certificado de estudios previos para la realización del Máster Semipresencial. En estos casos, será necesario presentarlo al departamento de prácticas de TECH para que se pueda confirmar la asignación del centro elegido.

7. NO INCLUYE: el Máster Semipresencial no incluirá ningún elemento no descrito en las presentes condiciones. Por tanto, no incluye alojamiento, transporte hasta la ciudad donde se realicen las prácticas, visados o cualquier otra prestación no descrita.

No obstante, el alumno podrá consultar con su tutor académico cualquier duda o recomendación al respecto. Este le brindará toda la información que fuera necesaria para facilitarle los trámites.

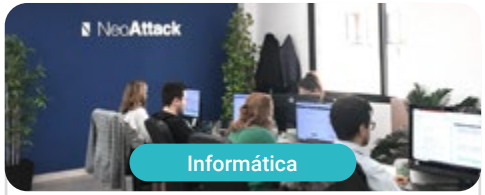
07

¿Dónde puedo hacer las Prácticas?

En línea con su compromiso de brindar una educación de alta calidad accesible para todos, TECH amplía las oportunidades académicas de los alumnos y posibilita que la estancia práctica se pueda llevar a cabo en diversas entidades de prestigio internacional. Así, los egresados tienen una oportunidad ideal para mejorar su calidad profesional trabajando con los mejores especialistas en el campo del Desarrollo de Software.

Completarás tu aprendizaje en Desarrollo de Software con una experiencia práctica junto a profesionales del sector”

El alumno podrá cursar la parte práctica de este Máster Semipresencial en los siguientes centros:



NeoAttack

País
España

Ciudad
Madrid

Dirección: Calle Santa Engracia 151, Planta 1, 1, Madrid

NeoAttack lidera el mercado llevando a cabo estrategias SEO y de publicidad

Capacitaciones prácticas relacionadas:
-Diseño Gráfico
-Desarrollo de Software

Informática

Captia Ingeniería

País	Ciudad
España	Madrid

Dirección: Av. de las Nieves, 37, Bloque A Planta 1 Oficina E, 28935, Móstoles, Madrid

Empresa informática dedicada a proporcionar soluciones tecnológicas avanzadas a las industrias

Capacitaciones prácticas relacionadas:
-Visual Analytics y Big Data
-Desarrollo de Software

Impulsa tu trayectoria profesional con una enseñanza holística, que te permite avanzar tanto a nivel teórico como práctico"

08

Metodología

Este programa de capacitación ofrece una forma diferente de aprender. Nuestra metodología se desarrolla a través de un modo de aprendizaje de forma cíclica: ***el Relearning***.

Este sistema de enseñanza es utilizado, por ejemplo, en las facultades de medicina más prestigiosas del mundo y se ha considerado uno de los más eficaces por publicaciones de gran relevancia como el ***New England Journal of Medicine.***

Descubre el Relearning, un sistema que abandona el aprendizaje lineal convencional para llevarte a través de sistemas cíclicos de enseñanza: una forma de aprender que ha demostrado su enorme eficacia, especialmente en las materias que requieren memorización"

Estudio de Caso para contextualizar todo el contenido

Nuestro programa ofrece un método revolucionario de desarrollo de habilidades y conocimientos. Nuestro objetivo es afianzar competencias en un contexto cambiante, competitivo y de alta exigencia.

Con TECH podrás experimentar una forma de aprender que está moviendo los cimientos de las universidades tradicionales de todo el mundo"

Accederás a un sistema de aprendizaje basado en la reiteración, con una enseñanza natural y progresiva a lo largo de todo el temario.

El alumno aprenderá, mediante actividades colaborativas y casos reales, la resolución de situaciones complejas en entornos empresariales reales.

Un método de aprendizaje innovador y diferente

El presente programa de TECH es una enseñanza intensiva, creada desde 0, que propone los retos y decisiones más exigentes en este campo, ya sea en el ámbito nacional o internacional. Gracias a esta metodología se impulsa el crecimiento personal y profesional, dando un paso decisivo para conseguir el éxito. El método del caso, técnica que sienta las bases de este contenido, garantiza que se sigue la realidad económica, social y profesional más vigente.

> *Nuestro programa te prepara para afrontar nuevos retos en entornos inciertos y lograr el éxito en tu carrera"*

El método del caso ha sido el sistema de aprendizaje más utilizado por las mejores escuelas de Informática del mundo desde que éstas existen. Desarrollado en 1912 para que los estudiantes de Derecho no solo aprendiesen las leyes a base de contenidos teóricos, el método del caso consistió en presentarles situaciones complejas reales para que tomasen decisiones y emitiesen juicios de valor fundamentados sobre cómo resolverlas. En 1924 se estableció como método estándar de enseñanza en Harvard.

Ante una determinada situación, ¿qué debería hacer un profesional? Esta es la pregunta a la que te enfrentamos en el método del caso, un método de aprendizaje orientado a la acción. A lo largo del curso, los estudiantes se enfrentarán a múltiples casos reales. Deberán integrar todos sus conocimientos, investigar, argumentar y defender sus ideas y decisiones.

Relearning Methodology

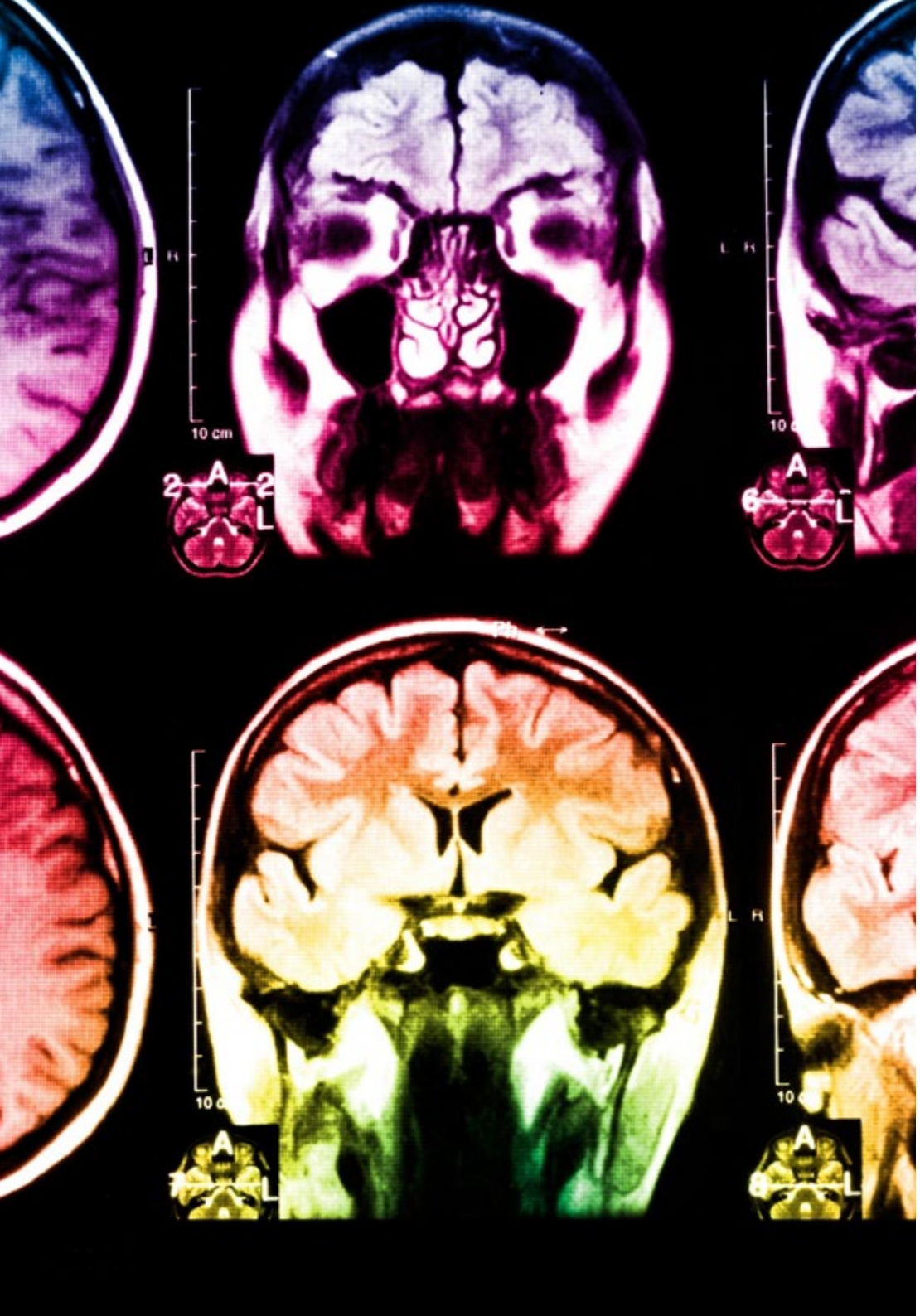
TECH aúna de forma eficaz la metodología del Estudio de Caso con un sistema de aprendizaje 100% online basado en la reiteración, que combina elementos didácticos diferentes en cada lección.

Potenciamos el Estudio de Caso con el mejor método de enseñanza 100% online: el Relearning.

En 2019 obtuvimos los mejores resultados de aprendizaje de todas las universidades online en español en el mundo.

En TECH aprenderás con una metodología vanguardista concebida para capacitar a los directivos del futuro. Este método, a la vanguardia pedagógica mundial, se denomina Relearning.

Nuestra universidad es la única en habla hispana licenciada para emplear este exitoso método. En 2019, conseguimos mejorar los niveles de satisfacción global de nuestros alumnos (calidad docente, calidad de los materiales, estructura del curso, objetivos…) con respecto a los indicadores de la mejor universidad online en español.



En nuestro programa, el aprendizaje no es un proceso lineal, sino que sucede en espiral (aprender, desaprender, olvidar y reaprender). Por eso, se combinan cada uno de estos elementos de forma concéntrica. Con esta metodología se han capacitado más de 650.000 graduados universitarios con un éxito sin precedentes en ámbitos tan distintos como la bioquímica, la genética, la cirugía, el derecho internacional, las habilidades directivas, las ciencias del deporte, la filosofía, el derecho, la ingeniería, el periodismo, la historia o los mercados e instrumentos financieros. Todo ello en un entorno de alta exigencia, con un alumnado universitario de un perfil socioeconómico alto y una media de edad de 43,5 años.

El Relearning te permitirá aprender con menos esfuerzo y más rendimiento, implicándote más en tu capacitación, desarrollando el espíritu crítico, la defensa de argumentos y el contraste de opiniones: una ecuación directa al éxito.

A partir de la última evidencia científica en el ámbito de la neurociencia, no solo sabemos organizar la información, las ideas, las imágenes y los recuerdos, sino que sabemos que el lugar y el contexto donde hemos aprendido algo es fundamental para que seamos capaces de recordarlo y almacenarlo en el hipocampo, para retenerlo en nuestra memoria a largo plazo.

De esta manera, y en lo que se denomina Neurocognitive context-dependent e-learning, los diferentes elementos de nuestro programa están conectados con el contexto donde el participante desarrolla su práctica profesional.

Este programa ofrece los mejores materiales educativos, preparados a conciencia para los profesionales:

Material de estudio

Todos los contenidos didácticos son creados por los especialistas que van a impartir el curso, específicamente para él, de manera que el desarrollo didáctico sea realmente específico y concreto.

Estos contenidos son aplicados después al formato audiovisual, para crear el método de trabajo online de TECH. Todo ello, con las técnicas más novedosas que ofrecen piezas de gran calidad en todos y cada uno los materiales que se ponen a disposición del alumno.

Clases magistrales

Existe evidencia científica sobre la utilidad de la observación de terceros expertos.

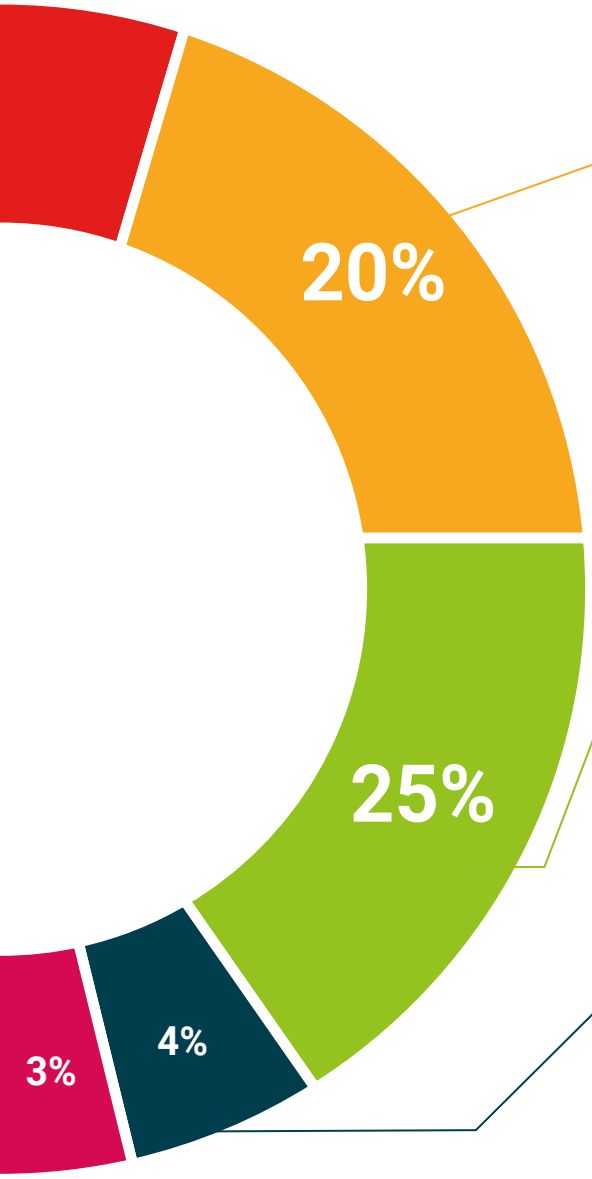
El denominado Learning from an Expert afianza el conocimiento y el recuerdo, y genera seguridad en las futuras decisiones difíciles.

Prácticas de habilidades y competencias

Realizarán actividades de desarrollo de competencias y habilidades específicas en cada área temática. Prácticas y dinámicas para adquirir y desarrollar las destrezas y habilidades que un especialista precisa desarrollar en el marco de la globalización que vivimos.

Lecturas complementarias

Artículos recientes, documentos de consenso y guías internacionales, entre otros. En la biblioteca virtual de TECH el estudiante tendrá acceso a todo lo que necesita para completar su capacitación.



Case studies

Completarán una selección de los mejores casos de estudio elegidos expresamente para esta titulación. Casos presentados, analizados y tutorizados por los mejores especialistas del panorama internacional.

Resúmenes interactivos

El equipo de TECH presenta los contenidos de manera atractiva y dinámica en píldoras multimedia que incluyen audios, vídeos, imágenes, esquemas y mapas conceptuales con el fin de afianzar el conocimiento.

Este exclusivo sistema educativo para la presentación de contenidos multimedia fue premiado por Microsoft como "Caso de éxito en Europa".

Testing & Retesting

Se evalúan y reevalúan periódicamente los conocimientos del alumno a lo largo del programa, mediante actividades y ejercicios evaluativos y autoevaluativos para que, de esta manera, el estudiante compruebe cómo va consiguiendo sus metas.

09

Titulación

El Título de Máster Semipresencial en Desarollo de Software garantiza, además de la capacitación más rigurosa y actualizada, el acceso a un título de Máster Semipresencial expedido por TECH Universidad Tecnológica.

Supera con éxito este programa y recibe tu titulación universitaria sin desplazamientos ni farragosos trámites”

Este **Título de Máster Semipresencial en Desarrollo de Software** contiene el programa más completo y actualizado del panorama profesional y académico.

Tras la superación de las pruebas por parte del alumno, este recibirá por correo postal, con acuse de recibo, el correspondiente Certificado de Máster Semipresencial expedido por TECH.

Además del Diploma, podrá obtener un certificado, así como el certificado del contenido del programa. Para ello, deberá ponerse en contacto con su asesor académico, que le brindara toda la información necesaria.

Título: **Máster Semipresencial en Desarrollo de Software**

Modalidad: **Semipresencial (Online + Prácticas)**

Duración: **12 meses**


tech universidad tecnológica

Otorga la presente

CONSTANCIA

a

C. _____, con documento de identificación nº_____

Por haber superado con éxito y acreditado el programa de

MÁSTER SEMIPRESENCIAL

en

Desarrollo de Software

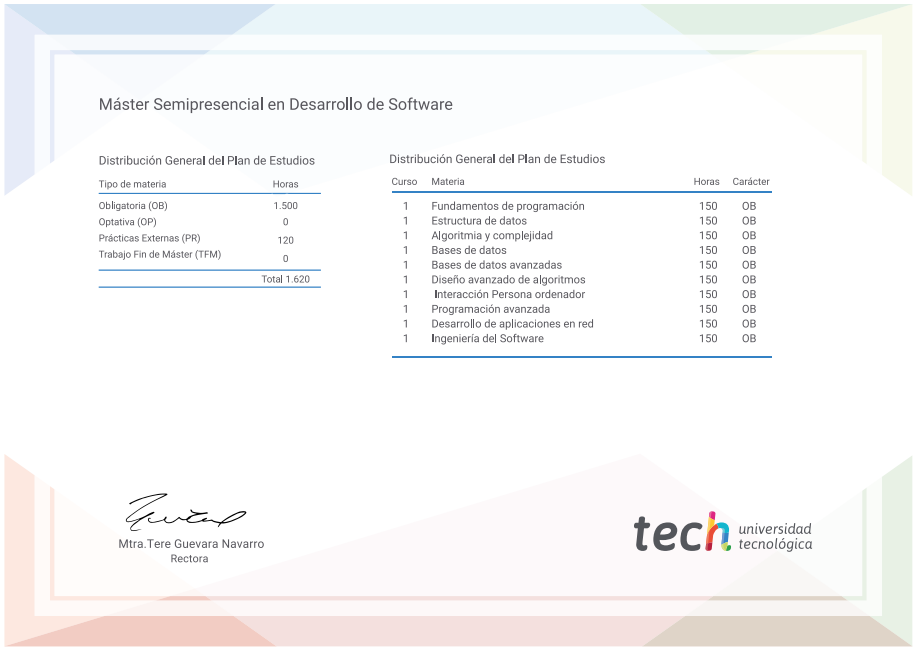
Se trata de un título propio de esta Universidad con una duración de 1.620 horas, con fecha de inicio dd/mm/aaaa y fecha de finalización dd/mm/aaaa.

TECH es una Institución Particular de Educación Superior reconocida por la Secretaría de Educación Pública a partir del 28 de junio de 2018.

A 17 de junio de 2020

Mtra.Tere Guevara Navarro
Rectora

código único TECH: AFWOR23S techtitute.com/titulos


Máster Semipresencial en Desarrollo de Software

Distribución General del Plan de Estudios

Tipo de materia	Horas
Obligatoria (OB)	1.500
Optativa (OP)	0
Prácticas Externas (PR)	120
Trabajo Fin de Máster (TFM)	0
	Total 1.620

Distribución General del Plan de Estudios

Curso	Materia	Horas	Carácter
1	Fundamentos de programación	150	OB
1	Estructura de datos	150	OB
1	Algoritmia y complejidad	150	OB
1	Bases de datos	150	OB
1	Bases de datos avanzadas	150	OB
1	Diseño avanzado de algoritmos	150	OB
1	Interacción Persona ordenador	150	OB
1	Programación avanzada	150	OB
1	Desarrollo de aplicaciones en red	150	OB
1	Ingeniería del Software	150	OB

Mtra.Tere Guevara Navarro
Rectora

tech universidad tecnológica

*Apostilla de La Haya. En caso de que el alumno solicite que su título en papel recabe la Apostilla de La Haya, TECH EDUCATION realizará las gestiones oportunas para su obtención, con un coste adicional.

tech universidad tecnológica

Máster Semipresencial
Desarrollo de Software

Modalidad: **Semipresencial (Online + Prácticas)**
Duración: **12 meses**
Titulación: **TECH Universidad Tecnológica**

Máster Semipresencial
Desarrollo de Software

tech universidad tecnológica