Máster Título Propio
Desarrollo de Software



tech universidad tecnológica



tech universidad tecnológica

Máster Título Propio
Desarrollo de Software

Modalidad: **Online**
Duración: **12 meses**
Titulación: **TECH Universidad Tecnológica**
Horas lectivas: **1.500 h.**
Acceso web: **www.techtitute.com/informatica/master/master-desarrollo-software**

Índice

01 Presentación — pág. 4

02 Objetivos — pág. 8

03 Competencias — pág. 14

04 Estructura y contenido — pág. 18

05 Metodología — pág. 32

06 Titulación — pág. 40

01

Presentación

Para participar de una de las áreas con mayor proyección en el sector informático, el profesional debe capacitarse científica y tecnológicamente, así como prepararse para poder enfrentar, de forma eficiente, los retos que surgen en el ejercicio profesional de la ingeniería del software, este Programa está orientado a lograr un alto dominio del Desarrollo de Software, a través de los últimos avances y desarrollos en este campo, mediante una metodología de estudio de máximo impacto y extraordinaria flexibilidad.





Adquiere los conocimientos más amplios en ingeniería del software, en la capacitación más actualizada del mercado docente online y comienza a trabajar en desarrollos en este dinámico campo profesional"

Con el avance de las nuevas tecnologías, el Software se ha convertido en un elemento sumamente importante en la actualidad. En los últimos años se ha puesto de manifiesto la necesidad de ser capaces de desarrollar productos Software con la funcionalidad y calidad adecuadas, respetando el tiempo y presupuesto establecido.

Este programa está dirigido a aquellas personas interesadas en alcanzar un nivel de conocimiento superior sobre el Desarrollo de Software. El principal objetivo es capacitar al alumno para que aplique en el mundo real los conocimientos adquiridos en este Máster Título Propio, en un entorno de trabajo que reproduzca las condiciones que se puede encontrar en el futuro, de manera rigurosa y realista.

Aproveche la oportunidad de cursar esta capacitación en un formato 100% online, sin tener que renunciar a las obligaciones, y haciendo fácil el regreso a la universidad. Actualiza sus conocimientos y consigua el título de Máster Título Propio para seguir creciendo personal y profesionalmente.

Se obtendrán amplios conocimientos en el campo de la ingeniería del software, pero también en el campo de la computación y la estructura de computadoras, incluyendo la base matemática, estadística y física imprescindible en una ingeniería.

Aprovecha la oportunidad y cursa esta capacitación en un formato 100% online, sin tener que renunciar a tus obligaciones, y haciendo fácil tu regreso a la universidad. Actualiza tus conocimientos y consigue tu título de Máster Título Propio para seguir creciendo personal y profesionalmente.

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

- Desarrollo de 100 escenarios simulados presentados por expertos en Desarrollo de Software
- Sus contenidos gráficos, esquemáticos y eminentemente prácticos con los que están concebidos, recogen una información científica y práctica 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
- Sistema interactivo de aprendizaje basado en el método del caso y su aplicación a la práctica real
- 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

Este programa te permitirá conocer la estructura básica de un ordenador y de su software, como base para incrementar tus competencias”

Aprende todo lo necesario para trabajar con lenguajes de programación con seguridad, incorporando a tus conocimientos la interpretación y diseño de algoritmos básicos para trabajar en programación”

Una capacitación que te permitirá entender el funcionamiento y cómo intervenir sobre todos los elementos esenciales de un programa informático.

Conoce los sistemas de datos más novedosos del mercado, aprende a diseñar algoritmos avanzados y todos los aspectos que el profesional de alta competencia debe dominar.

Incluye en su cuadro docente profesionales pertenecientes al ámbito del mundo del Desarrollo de Software, que vierten en esta capacitación la experiencia de su trabajo, además de reconocidos especialistas pertenecientes a sociedades de referencia y universidades de prestigio.

Gracias a su contenido multimedia elaborado con la última tecnología educativa, permitirán al profesional 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 se centra en el Aprendizaje Basado en Problemas, mediante el cual el docente deberá tratar de resolver las distintas situaciones de práctica profesional que se le planteen a lo largo del programa. Para ello, el profesional contará con la ayuda de un novedoso sistema de vídeo interactivo realizado por reconocidos expertos en Desarrollo de Software con gran experiencia docente.



02

Objetivos

El objetivo de esta capacitación es ofrecer a los profesionales que trabajan en el Desarrollo Software, los conocimientos y habilidades necesarios para realizar su actividad utilizando los protocolos y técnicas más avanzados del momento. Mediante un planteamiento de trabajo totalmente adaptable al alumno, este Máster Título Propio te llevará progresivamente a adquirir las competencias que lo impulsarán hacia un nivel profesional superior.





Profundizarás en el campo de la computación y la estructura de computadores, materias esenciales para cualquier desarrollador de software"

Objetivos generales

- Capacitar científica y tecnológicamente, así como preparar para el ejercicio profesional de la ingeniería del Software, todo ello con una capacitación transversal y versátil adaptada a las nuevas tecnologías e innovaciones en este campo
- Obtener amplios conocimientos en el campo de la ingeniería del Software, pero también en el campo de la computación y la estructura de computadoras, todo ello incluyendo la base matemática, estadística y física imprescindible en una ingeniería

Alcanza el nivel de conocimiento que deseas y domina el Desarrollo de Software con esta capacitación de alto nivel"





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. Conoceremos también el uso de algoritmos greedy sobre grafos
- 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 MongoDB

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 el porqué hay que tenerlas en cuenta a la hora de diseñar nuestro 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 *Frontend* 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

03

Competencias

Al superar las evaluaciones del Máster Título Propio en Desarrollo de Software, habrá adquirido las competencias profesionales necesarias para realizar un trabajo de calidad y además podrá adquirido nuevas habilidades y técnicas que le ayudarán a complementar los conocimientos informáticos que poseía previamente.



Aumenta tus competencias en Desarrollo de Software y pasa al siguiente nivel como profesional en este campo en constante evolución"

Competencia general

- Responder a las necesidades actuales del área de desarrollo de Software

Un programa excepcional por su densidad, su completa actualidad y la forma de enseñanza en la que se ofrece, que te permitirá avanzar de manera rápida y eficaz"





Competencias específicas

- 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
- 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
- Ser capaz de aplicar Sentar, los principales procesos y conceptos de las bases de la ingeniería del Software y el modelado

04

Estructura y contenido

La estructura de los contenidos ha sido diseñada por un equipo de profesionales de Ingeniería Informática con el objetivo de conseguir que los alumnos del Máster Título Propio puedan aprender de manera eficaz y rápida. Para ello se han organizado los contenidos de manera que el aprendizaje sea intensivo y constante, tratando de mantener la motivación en base a la sensación de progreso del alumno.





Un programa educativo destinado a conseguir una habilidad completa en desarrollo de software, que te impulsará a un nuevo nivel profesional"

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

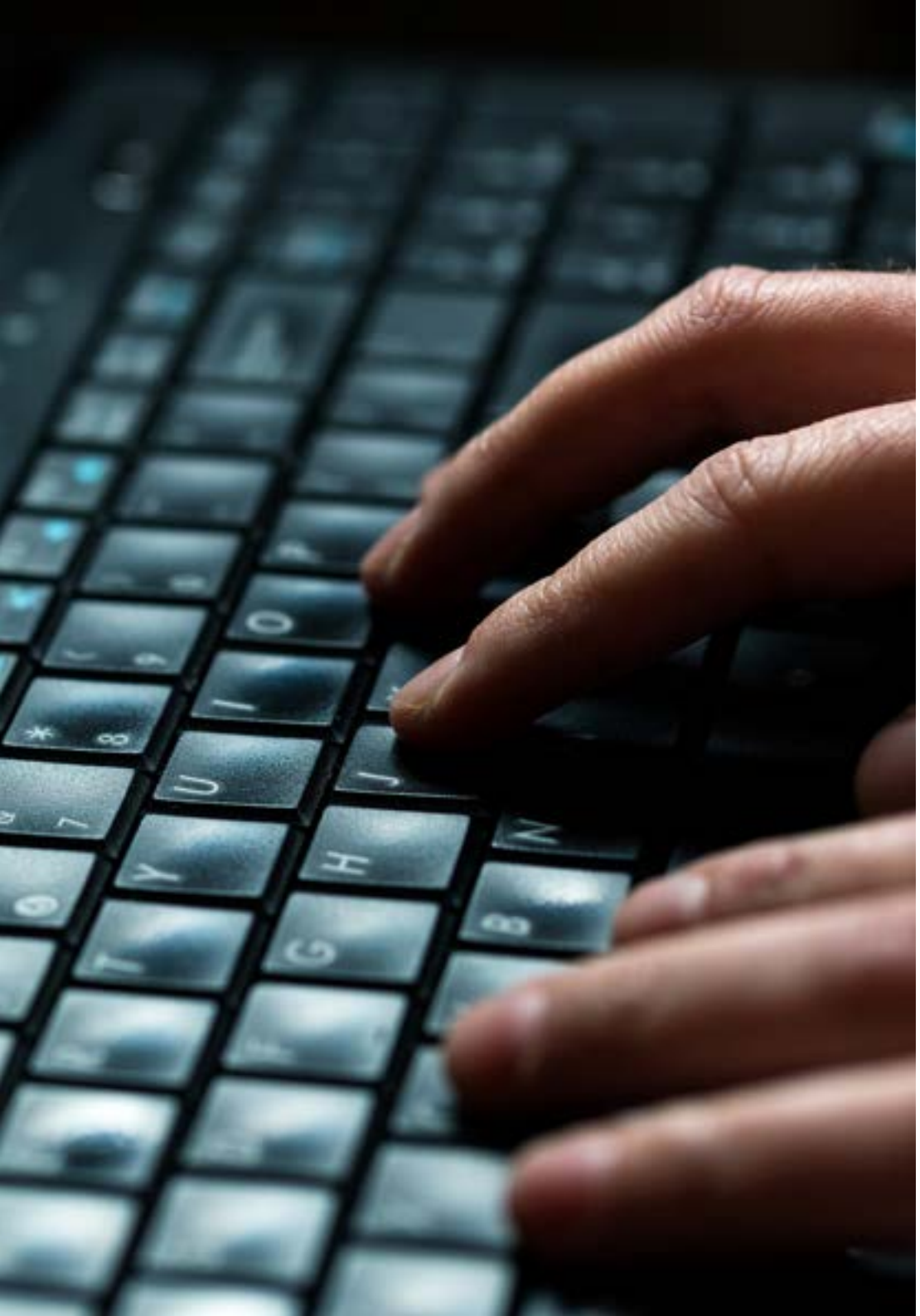
1.5. Abstracción y modularidad: funciones
- 1.5.1. Diseño modular
- 1.5.2. Concepto de función y utilidad
- 1.5.3. Definición de una función
- 1.5.4. Flujo de ejecución en la llamada de una función
- 1.5.5. Prototipo de una función
- 1.5.6. Devolución de resultados
- 1.5.7. Llamada a una función: parámetros
- 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 base 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

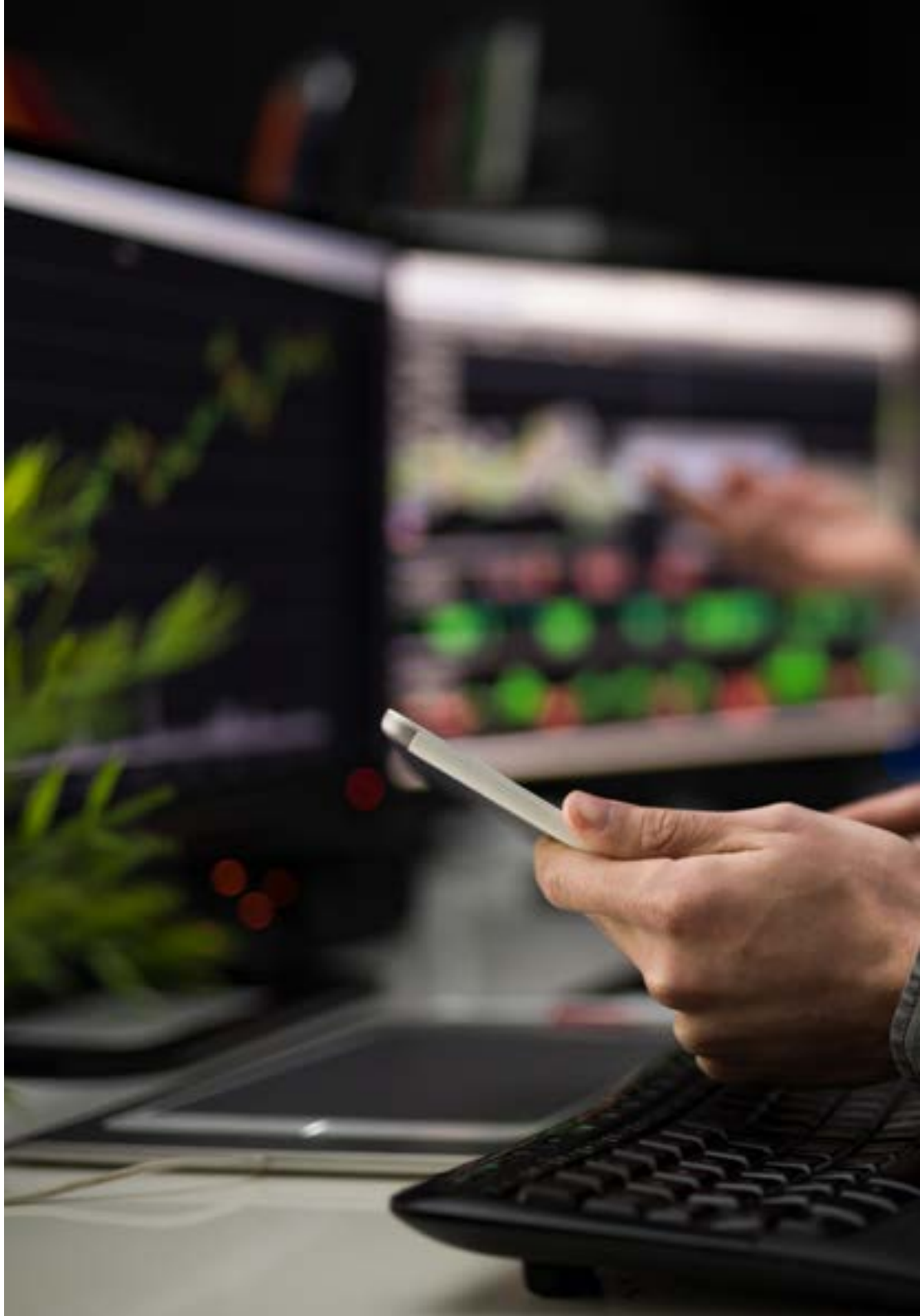
10.5. Modelado de los requisitos: escenarios, información y clases de análisis
- 10.5.1. Análisis de los requisitos
- 10.5.2. Modelado basado en escenarios
- 10.5.3. Modelos UML que proporcionan el caso de uso
- 10.5.4. Conceptos de modelado de datos
- 10.5.5. Modelado basado en clases
- 10.5.6. Diagramas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

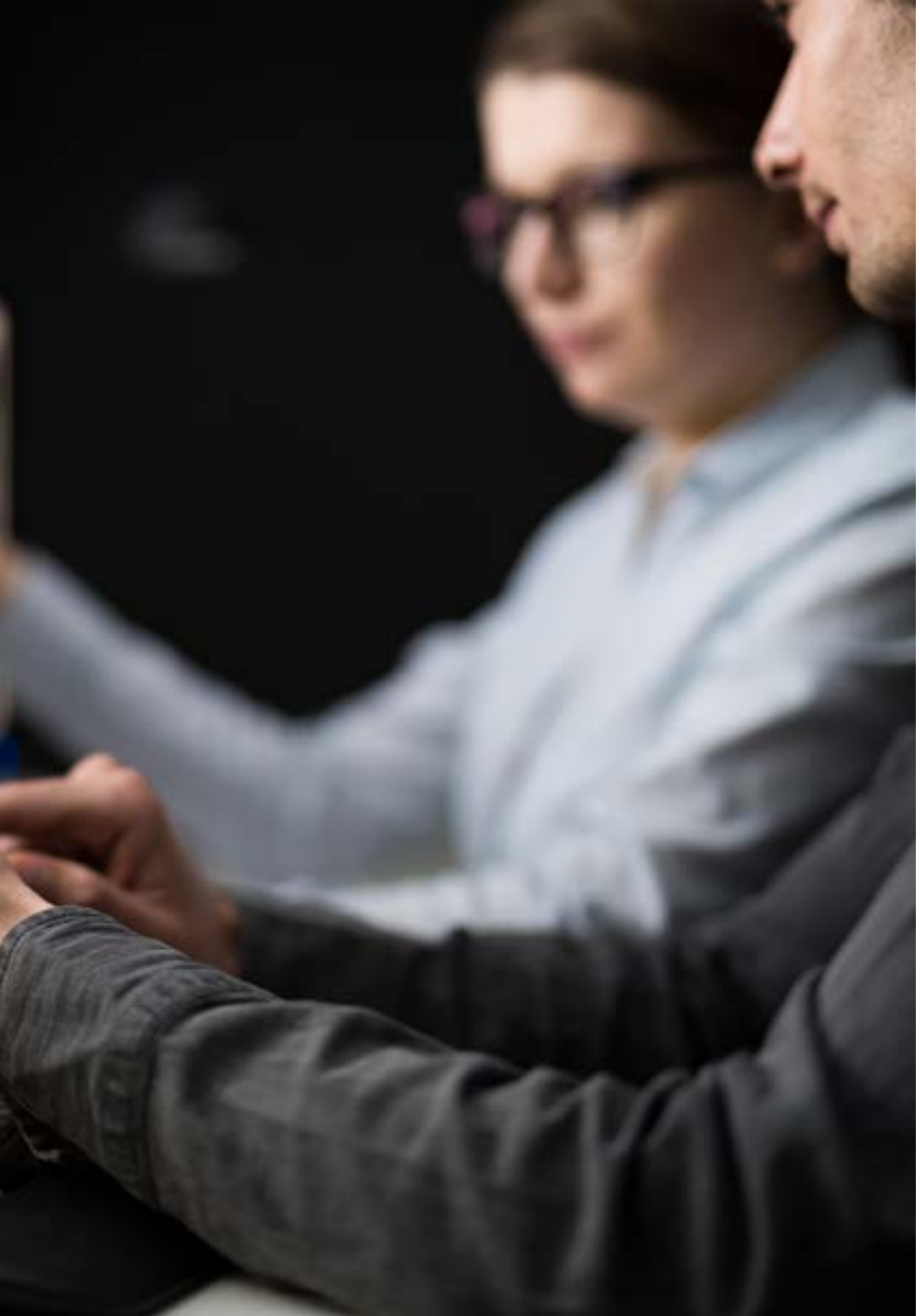
10.6. Modelado de los requisitos: flujo, comportamiento y patrones
- 10.6.1. Requisitos que modelan las estrategias
- 10.6.2. Modelado orientado al flujo
- 10.6.3. Diagramas de estado
- 10.6.4. Creación de un modelo de comportamiento
- 10.6.5. Diagramas de secuencia
- 10.6.6. Diagramas de comunicación
- 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 software y administración de proyectos
- 10.10.1. Calidad
- 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

Una experiencia de capacitación única, clave y decisiva para impulsar tu desarrollo profesional"

05

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.

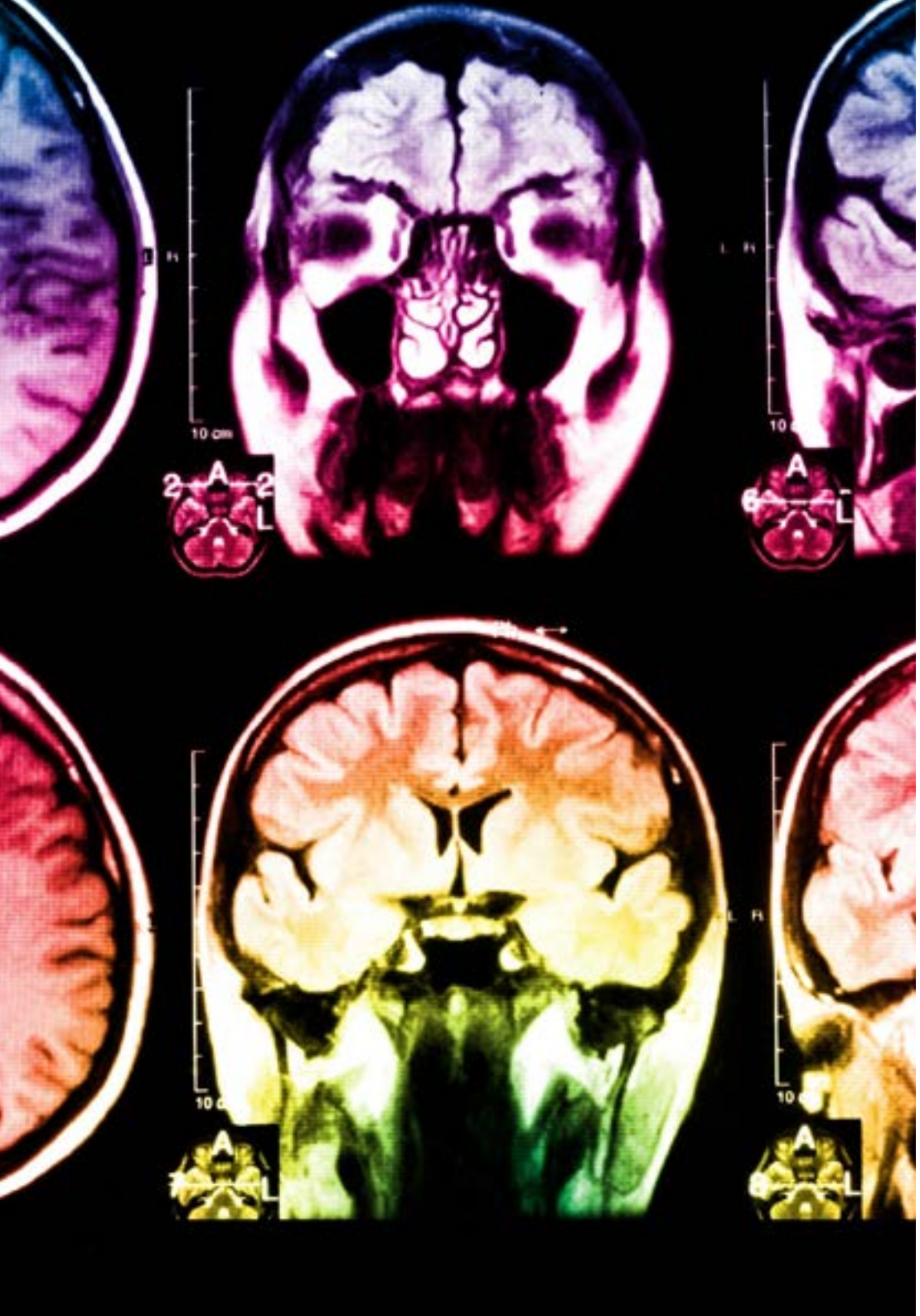
Potenciamos el Estudio de Caso con el mejor método de enseñanza 100% online: el Relearning.

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.

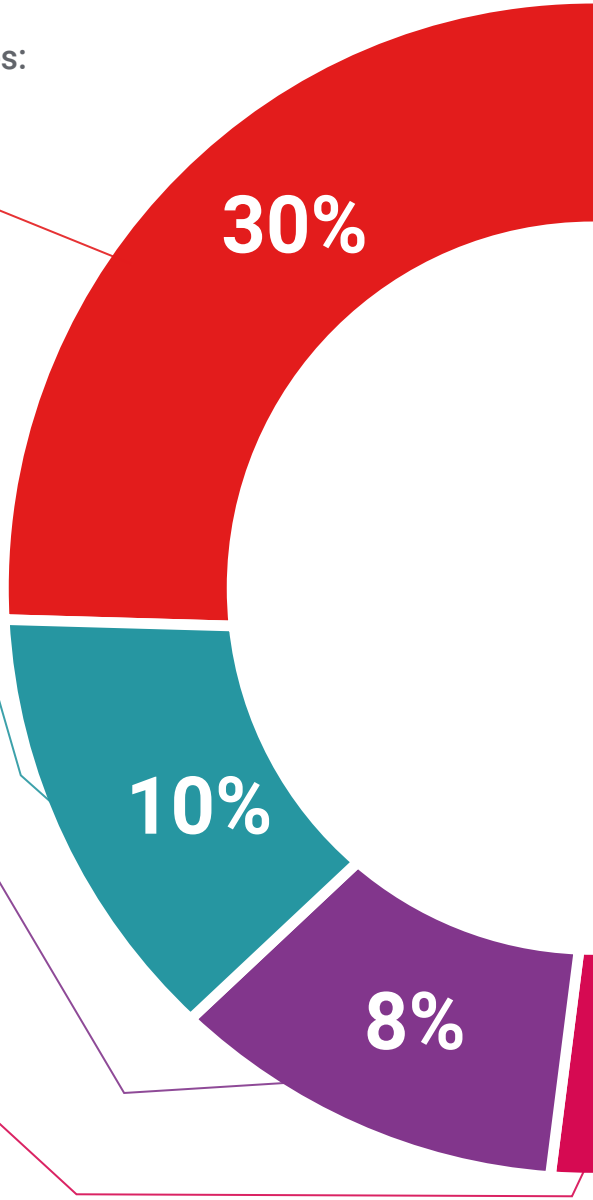
El denominado Learning from an Expert afianza el conocimiento y el recuerdo, y genera seguridad en las futuras decisiones difíciles.

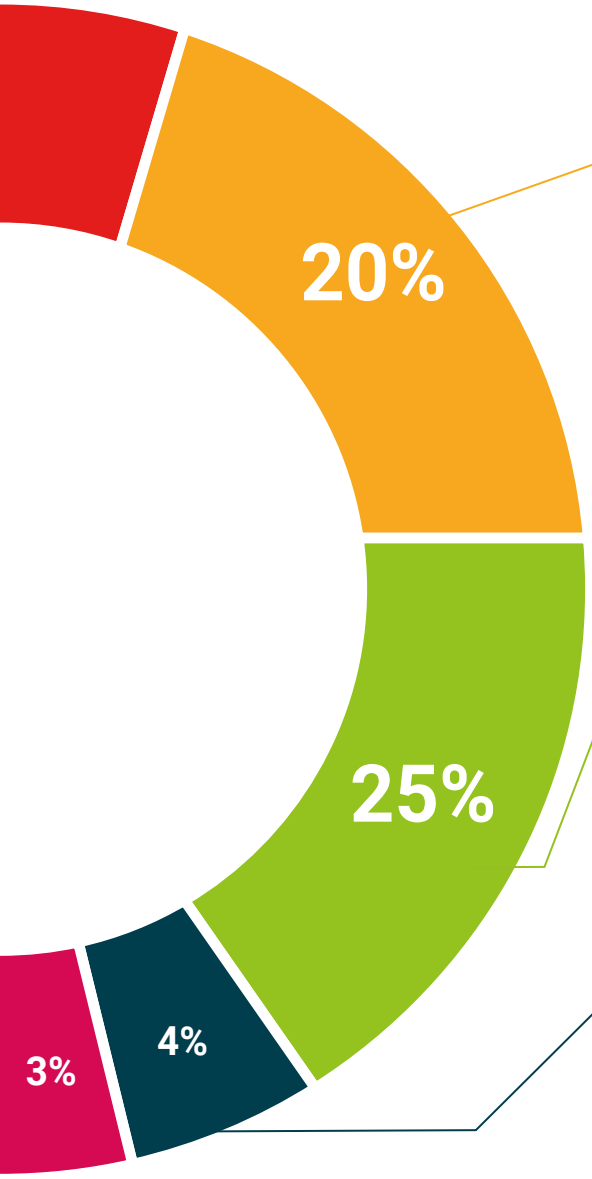
Prácticas de habilidades y competencias

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.

06

Titulación

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





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

Este **Máster Título Propio en Desarrollo de Software** contiene el programa más completo y actualizado del mercado.

Tras la superación de la evaluación, el alumno recibirá por correo postal* con acuse de recibo su correspondiente título de **Máster Propio** emitido por **TECH Universidad Tecnológica**.

El título por **TECH Universidad Tecnológica** expresará la calificación que haya obtenido en el Máster Título Propio y reunirá los requisitos comúnmente exigidos por las bolsas de trabajo, oposiciones y comités evaluadores de carreras profesionales.

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

N.º Horas Oficiales: **1.500 h.**


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 TÍTULO PROPIO

en

Desarrollo de Software

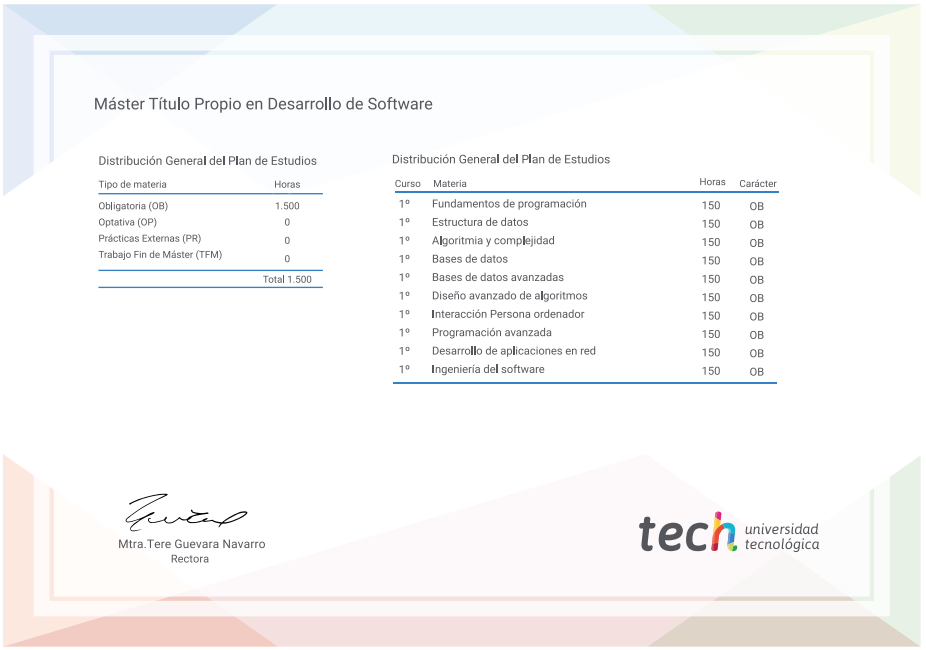
Se trata de un título propio de esta Universidad con una duración de 1.500 horas, con fecha de inicio dd/mm/aaaa y fecha de finalización dd/mm/aaaa.

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

Este título propio se deberá acompañar siempre del título universitario habilitante expedido por la autoridad competente para ejercer profesionalmente en cada país. código único TECH: AFWOR23S techtitute.com/titulos


Máster Título Propio en Desarrollo de Software

Distribución General del Plan de Estudios

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

Distribución General del Plan de Estudios

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



Máster Título Propio

Desarrollo de Software

Modalidad: **Online**

Duración: **12 meses**

Titulación: **TECH Universidad Tecnológica**

Horas lectivas: **1.500 h.**

Máster Título Propio
Desarrollo de Software



tech universidad tecnológica