

Master Privato

Programmazione di Videogiochi



tech università
tecnologica

Master Privato Programmazione di Videogiochi

- » Modalità: online
- » Durata: 12 mesi
- » Titolo: TECH Università Tecnologica
- » Dedizione: 16 ore/settimana
- » Orario: a scelta
- » Esami: online

Accesso al sito web: www.techitute.com/it/informatica/master/master-programmazione-videogiochi

Indice

01

Presentazione

pag. 4

02

Obiettivi

pag. 8

03

Competenze

pag. 12

04

Struttura e contenuti

pag. 16

05

Metodologia

pag. 32

06

Titolo

pag. 40

01

Presentazione

La maggiore attrattiva di un videogioco risiede nei suoi aspetti più visivi, come la grafica o il design. Tuttavia, questi non possono distinguersi senza la programmazione. La programmazione è la chiave di ogni videogioco, poiché ne determina la giocabilità o il modo in cui la grafica interagisce con il giocatore. Senza una buona programmazione, qualsiasi gioco fallirebbe, in quanto presenterebbe numerosi bug e un'esperienza spiacevole. Le aziende ne sono consapevoli e per questo richiedono sviluppatori di alto livello. La presente qualifica risponde a questa esigenza, in quanto prepara gli studenti ad affrontare tutte le sfide del settore, grazie alle quali potranno ottenere numerose opportunità professionali.



“

Le aziende sanno che la chiave di un videoggioco sta nella programmazione. Specializzati e diventa lo sviluppatore più richiesto nel tuo ambiente"

Dietro ad ogni grande videogioco c'è un'enorme squadra di professionisti specializzati in ogni area di lavoro che cercheranno di portare al successo la propria azienda. Di solito, gli aspetti più importanti per i fan sono quelli che possono percepire direttamente, come le immagini o quelli relativi al controllo dei personaggi, alle meccaniche o all'interazione con gli oggetti.

Tuttavia, affinché tutti questi elementi funzionino e siano correttamente integrati, è bene tenere conto di un lavoro essenziale che in genere non viene preso in considerazione: la programmazione. Lo sviluppo di un videogioco prevede diverse fasi e coinvolge vari reparti, ma la programmazione è quella che dà un senso a tutto e costituisce lo scheletro di base su cui verranno incorporate le altre aree.

Per questo motivo le aziende del settore prestano molta attenzione a questo aspetto, poiché sanno che uno sviluppo corretto ed efficiente dei loro videogiocchi faciliterà l'avanzamento del progetto ed eviterà la comparsa di errori e *Bug*. Di conseguenza, cercano i migliori programmatori specializzati in questo campo.

Ma non è facile trovare veri specialisti del settore. Questo Master Privato in Programmazione di Videogiocchi risponde perciò a questa esigenza, facendo sì che gli studenti diventino grandi esperti nello sviluppo di videogiocchi e che possano svilupparsi nel settore con facilità, ottenendo grandi opportunità di carriera grazie alle competenze e alle abilità acquisite nel corso del programma.

Questo **Master Privato in Programmazione di Videogiocchi** possiede il programma educativo più completo e aggiornato del mercato. Le caratteristiche principali del corso sono:

- ◆ Elaborazione di casi di studio presentati da esperti in sviluppo e programmazione di videogiochi
- ◆ Contenuti grafici, schematici ed eminentemente che forniscono informazioni scientifiche e sanitarie sulle discipline essenziali per l'esercizio professionale
- ◆ Esercizi pratici in cui il processo di autovalutazione può essere utilizzato per migliorare l'apprendimento
- ◆ Speciale enfasi sulle metodologie innovative
- ◆ Lezioni teoriche, domande all'esperto, forum di discussione su questioni controverse e lavoro di riflessione individuale
- ◆ Disponibilità di accesso ai contenuti da qualsiasi dispositivo fisso o portatile con una connessione internet



Sviluppa qualsiasi tipo di videogioco all'interno delle migliori aziende del mondo grazie a questo Master Privato

“

La programmazione è sempre più essenziale nello sviluppo di un videogioco. Diventa parte essenziale dell'industria grazie a questa qualifica”

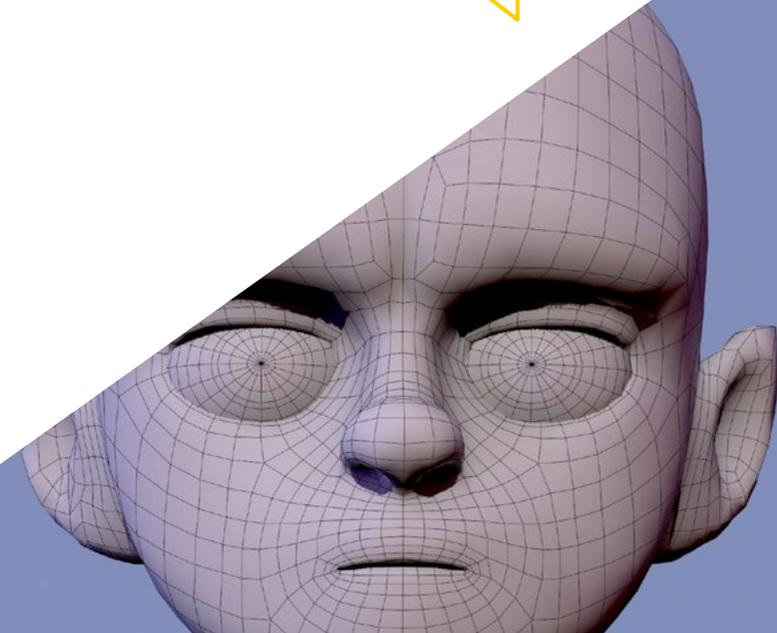
Il programma comprende, nel suo personale docente, prestigiosi professionisti che apportano la propria esperienza, così come specialisti riconosciuti e appartenenti a società scientifiche e università di riferimento.

I contenuti multimediali, sviluppati in base alle ultime tecnologie educative, forniranno al professionista un apprendimento coinvolgente e localizzato, ovvero inserito in un contesto reale.

La creazione di questo programma è incentrata sull'Apprendimento Basato su Problemi, mediante il quale lo specialista deve cercare di risolvere le diverse situazioni che gli si presentano durante il corso. A tal fine, lo studente potrà usufruire di un innovativo sistema di video interattivi creati da esperti di rinomata fama.

I videogiochi sono la tua passione e vuoi diventare un grande sviluppatore? Non esitare e iscriviti a questo Master Privato.

Le migliori aziende del settore ti aspettano. Specializzati ora.



02 Obiettivi

L'obiettivo principale di questo Master Privato è trasformare gli studenti in grandi sviluppatori di videogiochi. Questo settore è in espansione e ha sempre più necessità di programmatori e specialisti con una preparazione di alto livello. Questa qualifica è quindi l'occasione perfetta per ottenere grandi opportunità di carriera in alcune delle aziende più prestigiose del mondo. Pertanto, questo programma offre ai suoi studenti tutte le competenze necessarie per diventare esperti molto ricercati in questo settore, ottenendo un significativo e immediato avanzamento di carriera.





“

Tutti i tuoi sogni sono ora a portata di mano grazie a questo Master Privato in Programmazione di Videogiochi”



Obiettivi generali

- ◆ Apprendere i diversi linguaggi e metodi di programmazione applicati ai videogiochi
- ◆ Approfondire il processo di produzione dei videogiochi e l'integrazione della programmazione in queste fasi
- ◆ Imparare i fondamenti del Video Game Design e le conoscenze teoriche che un Video Game Designer deve conoscere
- ◆ Padroneggiare i linguaggi di programmazione di base utilizzati nei videogiochi
- ◆ Applicare la conoscenza dell'ingegneria del software e della programmazione specializzata ai videogiochi
- ◆ Comprendere il ruolo della programmazione nello sviluppo di un videogioco
- ◆ Conoscere le diverse console e piattaforme esistenti
- ◆ Sviluppare videogiochi web e multiplayer



Al termine di questa qualifica sarai il miglior sviluppatore di videogiochi del tuo territorio"



Obiettivi specifici

Modulo 1. Fondamenti di programmazione

- ◆ Comprendere la struttura di base di un computer, il software e i linguaggi di programmazione di uso generale
- ◆ Analizzare gli elementi essenziali di un programma per computer, come i diversi tipi di dati, gli operatori, le espressioni, le sentenze, le strutture di controllo e gli I/O
- ◆ Interpretare gli algoritmi, che sono la base necessaria per poter sviluppare programmi informatici

Modulo 2. Struttura dei dati e algoritmi

- ◆ Imparare le principali strategie di progettazione degli algoritmi e i diversi metodi e misure per il calcolo degli algoritmi
- ◆ Distinguere il funzionamento degli algoritmi, la loro strategia ed esempi del loro utilizzo nei principali problemi noti
- ◆ Comprendere la tecnica del *Backtracking* e i suoi principali utilizzi

Modulo 3. Programmazione orientata agli oggetti

- ◆ Conoscere i diversi Design Pattern per i problemi legati all'orientamento agli oggetti
- ◆ Comprendere l'importanza della documentazione e dei test nello sviluppo del software
- ◆ Gestire l'uso dei thread e della sincronizzazione, nonché la risoluzione di problemi comuni nell'ambito della programmazione concorrente

Modulo 4. Console e dispositivi per videogiochi

- ◆ Comprendere il funzionamento di base delle principali periferiche di ingresso e di uscita
- ◆ Comprendere le principali implicazioni progettuali delle diverse piattaforme
- ◆ Studiare la struttura, l'organizzazione, il funzionamento e l'interconnessione di dispositivi e sistemi
- ◆ Comprendere il ruolo del sistema operativo e dei kit di sviluppo per dispositivi mobili e piattaforme di videogiochi

Modulo 5. Ingegneria del software

- ◆ Distinguere le basi dell'ingegneria del software, il processo del software e i diversi modelli di sviluppo del software, comprese le tecnologie agili
- ◆ Riconoscere l'ingegneria dei requisiti, il suo sviluppo, la sua elaborazione, la sua negoziazione e la sua convalida per comprendere i principali standard relativi alla qualità del software e alla gestione dei progetti

Modulo 6. Motori dei videogiochi

- ◆ Scoprire il funzionamento e l'architettura di un motore per videogiochi
- ◆ Comprendere le caratteristiche di base dei motori di gioco esistenti
- ◆ Programmare applicazioni in modo corretto ed efficiente applicate ai motori per videogiochi
- ◆ Scegliere il paradigma e i linguaggi di programmazione più appropriati per programmare applicazioni applicate ai motori per videogiochi

Modulo 7. Sistemi intelligenti

- ◆ Stabilire i concetti relativi alla teoria e all'architettura degli agenti e ai loro processi di ragionamento
- ◆ Assimilare la teoria e la pratica dei concetti di informazione e conoscenza, nonché i diversi modi di rappresentare la conoscenza
- ◆ Comprendere il funzionamento dei ragionatori semantici, dei sistemi basati sulla conoscenza e dei sistemi esperti

Modulo 8. Programmazione in tempo reale

- ◆ Analizzare le caratteristiche principali di un linguaggio di programmazione in tempo reale che lo differenziano da un linguaggio di programmazione tradizionale
- ◆ Comprendere i concetti di base dei sistemi informatici
- ◆ Acquisire la capacità di applicare le principali basi e tecniche della programmazione in tempo reale

Modulo 9. Progettazione e sviluppo dei browser game

- ◆ Progettare giochi e applicazioni web interattive con la relativa documentazione
- ◆ Valutare le caratteristiche principali dei giochi e delle applicazioni web interattive per comunicare in modo professionale e corretto

Modulo 10. Reti e sistemi multigiocatore

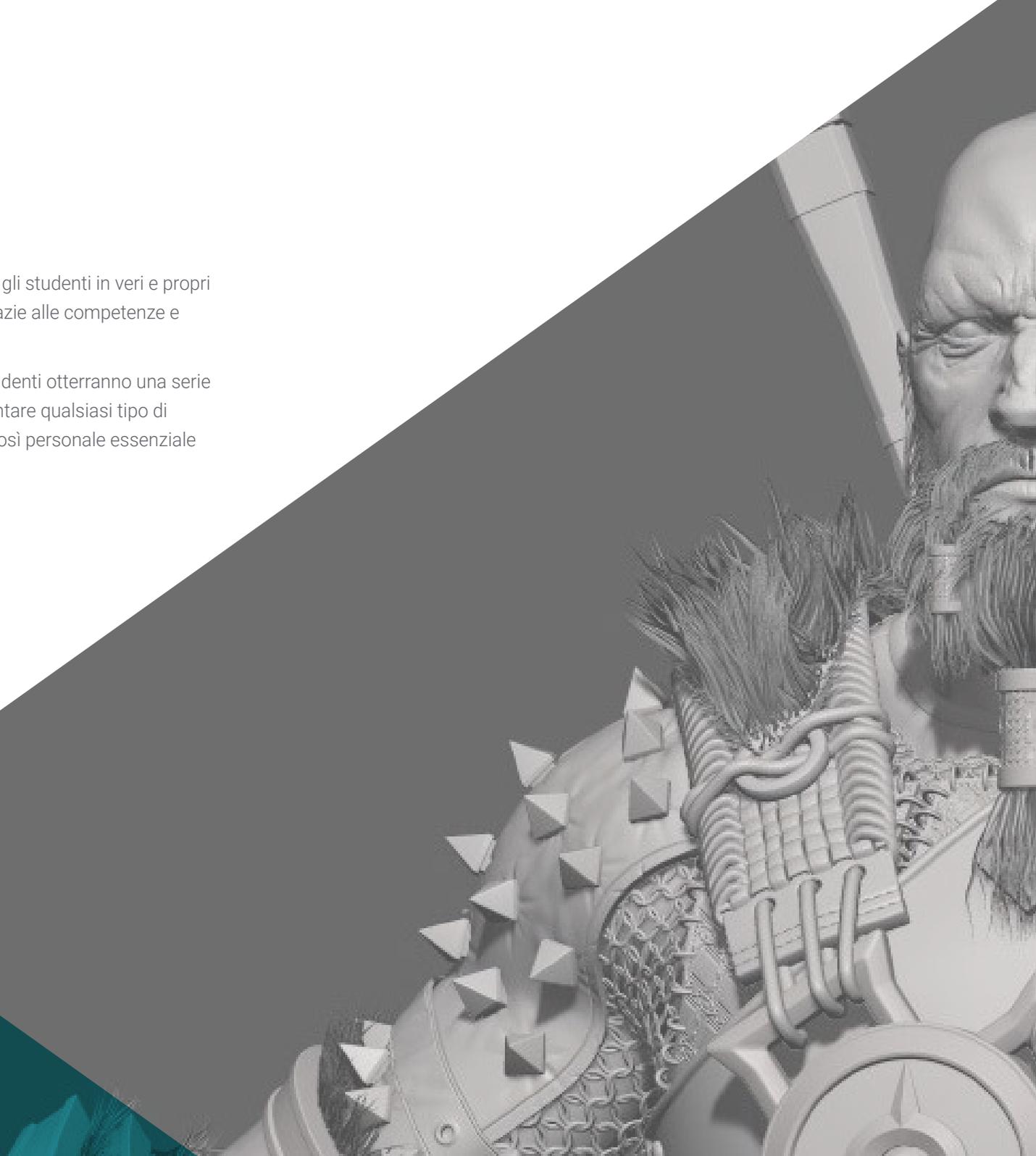
- ◆ Descrivere l'architettura del Transmission Control Protocol/Internet Protocol (TCP/IP) e il funzionamento di base delle reti wireless
- ◆ Analizzare la sicurezza applicata ai videogiochi
- ◆ Acquisire la capacità di sviluppare giochi online multiplayer

03

Competenze

Il Master Privato in Programmazione di Videogiochi trasforma gli studenti in veri e propri specialisti nello sviluppo di questo tipo di opere audiovisive grazie alle competenze e alle abilità fornite.

In questo modo, grazie a questo eccellente programma, gli studenti otterranno una serie di strumenti professionali con i quali saranno in grado di affrontare qualsiasi tipo di sfida legata alla programmazione di videogiochi, diventando così personale essenziale all'interno delle loro aziende.





“

*Imparerai a conoscere tutti gli
aspetti dello sviluppo di videogiochi”*

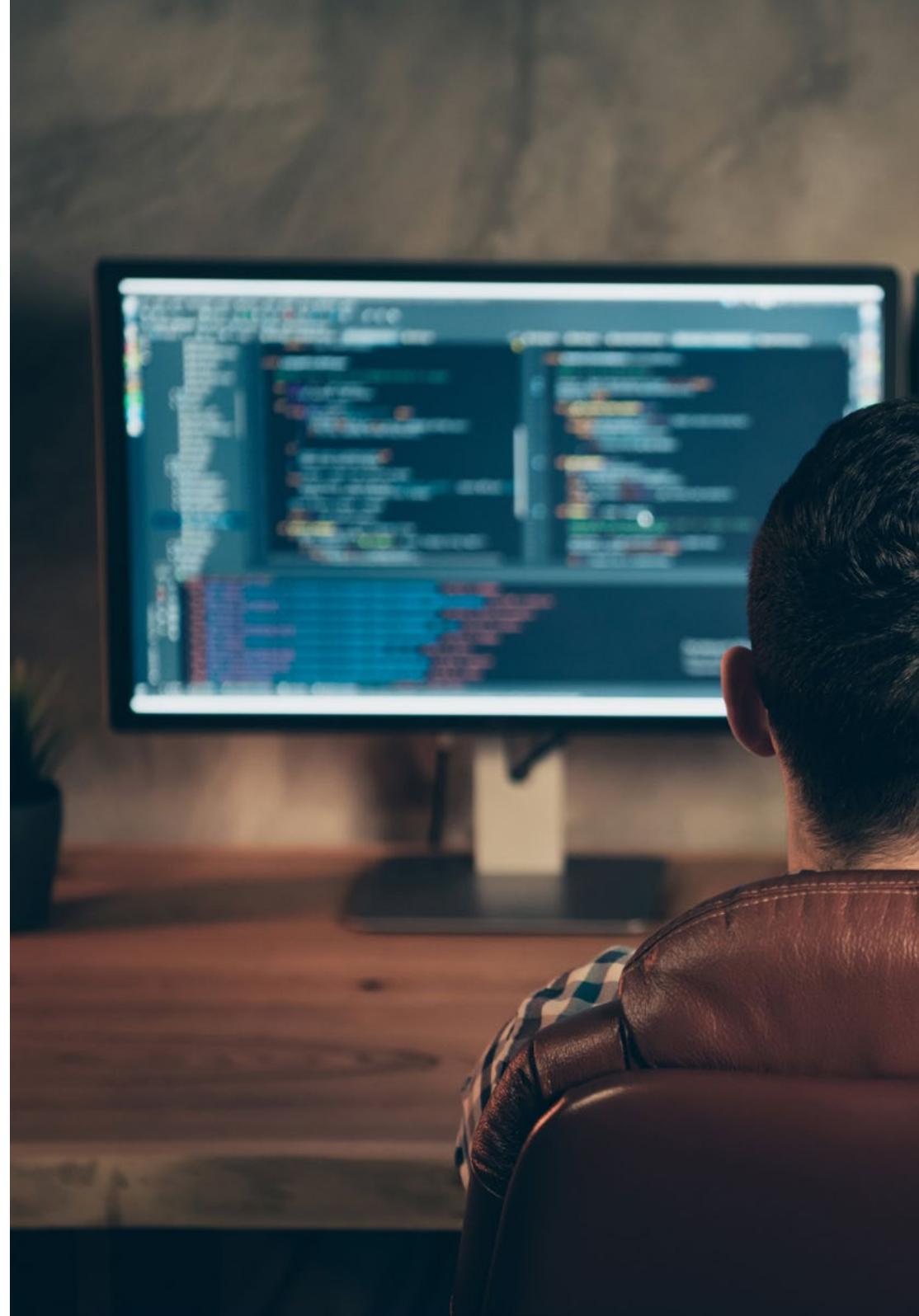


Competenze generali

- ◆ Progettare tutte le fasi di un videogioco, dall'idea iniziale al lancio finale
- ◆ Specializzarsi come programmatore di videogiochi
- ◆ Studiare a fondo tutte le parti dello sviluppo, dall'architettura iniziale, alla programmazione del personaggio giocatore e a tutti gli elementi coinvolti nel processo di gioco
- ◆ Ottenere una visione complessiva del progetto, essendo in grado di fornire soluzioni ai diversi problemi e sfide che si presentano nella progettazione di un videogioco

“

Raggiungi l'eccellenza come programmatore di videogiochi grazie a questo Master Privato”





Competenze specifiche

- ◆ Conoscere i software necessari per essere uno sviluppatore di videogiochi professionista
- ◆ Comprendere l'esperienza del giocatore e saper analizzare il gameplay del videogioco
- ◆ Comprendere tutte le procedure teoriche e pratiche del processo di programmazione dei videogiochi
- ◆ Padroneggiare i linguaggi di programmazione più utili per il mondo dei videogiochi
- ◆ Integrare la programmazione appresa con diversi tipi di console e piattaforme
- ◆ Programmare browser game e giochi multiplayer
- ◆ Assimilare il concetto di motori di gioco per poter programmare correttamente
- ◆ Applicare la conoscenza dell'ingegneria del software alla programmazione di videogiochi

04

Struttura e contenuti

I contenuti di questo Master Privato in Programmazione di Videogiochi sono stati accuratamente progettati da un personale docente composto da grandi specialisti del settore che conoscono alla perfezione lo stato attuale dell'industria. Grazie a questo programma, gli studenti potranno apprendere tutte le conoscenze necessarie per essere in grado di rispondere alle richieste delle aziende del settore, essendo stati appositamente preparati per le loro particolarità e specificità, che sono complesse e in continua evoluzione.





“

Questi contenuti ti faranno diventare un grande esperto in Programmazione di Videogiochi”

Modulo 1. Fondamenti di programmazione

- 1.1. Introduzione alla programmazione
 - 1.1.1. Struttura di base di un computer
 - 1.1.2. Software
 - 1.1.3. Il linguaggio di programmazione
 - 1.1.4. Ciclo di vita di un'applicazione informatica
- 1.2. Progettazione di algoritmi
 - 1.2.1. La risoluzione di problemi
 - 1.2.2. Tecniche descrittive
 - 1.2.3. Elementi e struttura di un algoritmo
- 1.3. Elementi di un programma
 - 1.3.1. Origine e caratteristiche del linguaggio C++
 - 1.3.2. L'ambiente di sviluppo
 - 1.3.3. Concetto di programma
 - 1.3.4. Tipi di dati fondamentali
 - 1.3.5. Operatori
 - 1.3.6. Espressioni
 - 1.3.7. Strutture
 - 1.3.8. Ingresso e uscita dati
- 1.4. Strutture di controllo
 - 1.4.1. Strutture
 - 1.4.2. Fork
 - 1.4.3. Loop
- 1.5. Astrazione e modularità: Funzioni
 - 1.5.1. Design modulare
 - 1.5.2. Concetto di funzione e utilità
 - 1.5.3. Definizione di una funzione
 - 1.5.4. Flusso di esecuzione nella chiamata di una funzione
 - 1.5.5. Prototipo di funzione
 - 1.5.6. Ritorno dei risultati
 - 1.5.7. Chiamare una funzione: Parametri
 - 1.5.8. Passaggio di parametri per riferimento e per valore
 - 1.5.9. Identificatore
- 1.6. Strutture dati statiche
 - 1.6.1. Arrays
 - 1.6.2. Matrici: Poliedri
 - 1.6.3. Ricerca e ordine
 - 1.6.4. Stringhe. Funzioni di I/O per le stringhe
 - 1.6.5. Strutture. Unioni
 - 1.6.6. Nuovi tipi di dati
- 1.7. Strutture dati dinamiche: Puntatori
 - 1.7.1. Concetto. Definizione di puntatore
 - 1.7.2. Operatori e operazioni con i puntatori
 - 1.7.3. Array di puntatori
 - 1.7.4. Puntatori e array
 - 1.7.5. Puntatori a stringhe
 - 1.7.6. Puntatori a strutture
 - 1.7.7. Indirizione multipla
 - 1.7.8. Puntatori a funzione
 - 1.7.9. Passaggio di funzioni, strutture e array come parametri di funzione
- 1.8. File
 - 1.8.1. Concetti di base
 - 1.8.2. Operazioni sui file
 - 1.8.3. Tipi di file
 - 1.8.4. Organizzazione dei file
 - 1.8.5. Introduzione ai file C++
 - 1.8.6. Gestione dei file
- 1.9. Ricorsione
 - 1.9.1. Definizione di ricorsione
 - 1.9.2. Tipi di ricorsione
 - 1.9.3. Vantaggi e svantaggi
 - 1.9.4. Considerazioni
 - 1.9.5. Conversione ricorsiva-iterativa
 - 1.9.6. Lo stack di ricorsione

- 1.10. Test e documentazione
 - 1.10.1. Test del programma
 - 1.10.2. Test della scatola bianca
 - 1.10.3. Test a scatola nera
 - 1.10.4. Strumenti di test
 - 1.10.5. Documentazione del programma

Modulo 2. Struttura dei dati e algoritmi

- 2.1. Introduzione alle strategie di progettazione degli algoritmi
 - 2.1.1. Ricorsione
 - 2.1.2. Dividere e conquistare
 - 2.1.3. Altre strategie
- 2.2. Efficienza e analisi degli algoritmi
 - 2.2.1. Misure di efficienza
 - 2.2.2. Misurare le dimensioni di ingresso
 - 2.2.3. Misurare il tempo di esecuzione
 - 2.2.4. Caso peggiore, migliore e medio
 - 2.2.5. Notazione asintotica
 - 2.2.6. Criteri di analisi matematica per algoritmi non ricorsivi
 - 2.2.7. Analisi matematica degli algoritmi ricorsivi
 - 2.2.8. Analisi empirica degli algoritmi
- 2.3. Algoritmi di ordinamento
 - 2.3.1. Concetto di ordinamento
 - 2.3.2. Ordinamento a bolle
 - 2.3.3. Ordinamento per selezione
 - 2.3.4. Ordinamento per inserimento
 - 2.3.5. Ordinamento per fusione (*merge_sort*)
 - 2.3.6. Ordinamento rapido (*quick_sort*)
- 2.4. Algoritmi ad albero
 - 2.4.1. Concetto di albero
 - 2.4.2. Alberi binari
 - 2.4.3. Percorsi degli alberi
 - 2.4.4. Rappresentare le espressioni
 - 2.4.5. Alberi binari ordinati
 - 2.4.6. Alberi binari bilanciati
- 2.5. Algoritmi con *Heaps*
 - 2.5.1. *Heaps*
 - 2.5.2. L'algoritmo *Heapsort*
 - 2.5.3. Le code di priorità
- 2.6. Algoritmi grafici
 - 2.6.1. Rappresentazione
 - 2.6.2. Percorso in ampiezza
 - 2.6.3. Percorso in profondità
 - 2.6.4. Ordinamento topologico
- 2.7. Algoritmi *Greedy*
 - 2.7.1. La strategia *Greedy*
 - 2.7.2. Elementi della strategia *Greedy*
 - 2.7.3. Cambio valuta
 - 2.7.4. Problema del commesso viaggiatore
 - 2.7.5. Problema dello zaino
- 2.8. Pathfinding minimo
 - 2.8.1. Il problema del percorso minimo
 - 2.8.2. Archi e cicli negativi
 - 2.8.3. Algoritmo di Dijkstra
- 2.9. Algoritmi *Greedy* sui grafi
 - 2.9.1. L'albero di copertura minimo
 - 2.9.2. Algoritmo di Prim
 - 2.9.3. Algoritmo di Kruksal
 - 2.9.4. Analisi della complessità
- 2.10. *Backtracking*
 - 2.10.1. Il *Backtracking*
 - 2.10.2. Tecniche alternative

Modulo 3. Programmazione orientata agli oggetti

- 3.1. Introduzione alla programmazione orientata agli oggetti
 - 3.1.1. Introduzione alla programmazione orientata agli oggetti
 - 3.1.2. Progettazione di classe
 - 3.1.3. Introduzione a UML per la modellazione dei problemi
- 3.2. Relazioni tra classi
 - 3.2.1. Astrazione e ereditarietà
 - 3.2.2. Concetti avanzati di ereditarietà
 - 3.2.3. Polimorfismo
 - 3.2.4. Composizione e aggregazione
- 3.3. Introduzione ai design pattern per i problemi orientati agli oggetti
 - 3.3.1. Cosa sono i Design Pattern?
 - 3.3.2. Pattern Factory
 - 3.3.4. Pattern Singleton
 - 3.3.5. Pattern Observer
 - 3.3.6. Pattern Composite
- 3.4. Eccezioni
 - 3.4.1. Cosa sono le eccezioni?
 - 3.4.2. Cattura e gestione delle eccezioni
 - 3.4.3. Lancio di eccezioni
 - 3.4.4. Creazione di eccezioni
- 3.5. Interfacce utente
 - 3.5.1. Introduzione a Qt
 - 3.5.2. Posizionamento
 - 3.5.3. Cosa sono gli eventi?
 - 3.5.4. Eventi: definizione e cattura
 - 3.5.5. Sviluppo di interfacce utente
- 3.6. Introduzione alla programmazione concorrente
 - 3.6.1. Introduzione alla programmazione concorrente
 - 3.6.2. Il processo e il concetto di thread
 - 3.6.3. Interazione tra processi o thread
 - 3.6.4. Thread in C++
 - 3.6.5. Vantaggi e svantaggi della programmazione concorrente
- 3.7. Gestione e sincronizzazione dei thread
 - 3.7.1. Ciclo di vita di un thread
 - 3.7.2. La Classe *Thread*
 - 3.7.3. Pianificazione dei thread
 - 3.7.4. Gruppi di thread
 - 3.7.5. Thread tipo demone
 - 3.7.6. Sincronizzazione
 - 3.7.7. Meccanismi di bloccaggio
 - 3.7.8. Meccanismi di comunicazione
 - 3.7.9. Monitor
- 3.8. Problemi comuni nella programmazione concorrente
 - 3.8.1. Il problema del consumatore-produttore
 - 3.8.2. Il problema dei lettori e degli scrittori
 - 3.8.3. Il problema della cena dei filosofi
- 3.9. Documentazione e test del software
 - 3.9.1. Perché la documentazione del software è importante?
 - 3.9.2. Documentazione di progetto
 - 3.9.3. Uso degli strumenti di documentazione
- 3.10. Test del software
 - 3.10.1. Introduzione al test del software
 - 3.10.2. Tipi di test
 - 3.10.3. Test unitari
 - 3.10.4. Test di integrazione
 - 3.10.5. Test di convalida
 - 3.10.6. Test del sistema

Modulo 4. Console e dispositivi per videogiochi

- 4.1. Storia della programmazione di videogiochi
 - 4.1.1. Periodo Atari (1977-1985)
 - 4.1.2. Periodo NES e SNES (1985-1995)
 - 4.1.3. Periodo PlayStation / PlayStation 2 (1995-2005)
 - 4.1.4. Periodo Xbox 360, PS3 e Wii (2005-2013)
 - 4.1.5. Xbox One, PS4 e Wii U - Periodo Switch (2013-presente)
 - 4.1.6. Il futuro
- 4.2. Storia del gameplay nei videogiochi
 - 4.2.1. Introduzione
 - 4.2.2. Il contesto sociale
 - 4.2.3. Diagramma strutturale
 - 4.2.4. Futuro
- 4.3. Adattamento ai tempi moderni
 - 4.3.1. Giochi basati sul movimento
 - 4.3.2. Realtà virtuale
 - 4.3.3. Realtà aumentata
 - 4.3.4. Realtà mista
- 4.4. *Unity: Scripting I* ed esempi
 - 4.4.1. Che cos'è uno *Script*?
 - 4.4.2. Il nostro primo *Script*
 - 4.4.3. Aggiunta di uno *Script*
 - 4.4.4. Apertura di uno *Script*
 - 4.4.5. MonoBehaviour
 - 4.4.6. *Debugging*
- 4.5. *Unity: Scripting II* ed esempi
 - 4.5.1. Input da tastiera e mouse
 - 4.5.2. Raycast
 - 4.5.3. Installazione
 - 4.5.4. Variabili
 - 4.5.5. Variabili pubbliche e serializzate
- 4.6. *Unity: Scripting III* ed esempi
 - 4.6.1. Ottenere i componenti
 - 4.6.2. Modifica dei componenti
 - 4.6.3. Test
 - 4.6.4. Oggetti multipli
 - 4.6.5. Colliders e Triggers
 - 4.6.6. Quaternioni
- 4.7. Periferiche
 - 4.7.1. Evoluzione e classificazioni
 - 4.7.2. Periferiche e interfacce
 - 4.7.3. Periferiche attuali
 - 4.7.4. Futuro prossimo
- 4.8. Videogiochi: prospettive future
 - 4.8.1. Giochi basati sul cloud
 - 4.8.2. Mancanza di controller
 - 4.8.3. Realtà immersiva
 - 4.8.4. Altre alternative
- 4.9. Architettura
 - 4.9.1. Esigenze speciali dei videogiochi
 - 4.9.2. Evoluzione dell'architettura
 - 4.9.3. Architettura contemporanea
 - 4.9.4. Differenze tra le architetture
- 4.10. Kit di sviluppo e loro evoluzione
 - 4.10.1. Introduzione
 - 4.10.2. Kit di sviluppo di terza generazione
 - 4.10.3. Kit di sviluppo di quarta generazione
 - 4.10.4. Kit di sviluppo di quinta generazione
 - 4.10.5. Kit di sviluppo di sesta generazione

Modulo 5. Ingegneria del software

- 5.1. Introduzione all'ingegneria e alla modellazione del software
 - 5.1.1. La natura del software
 - 5.1.2. La natura unica delle webapp
 - 5.1.3. Ingegneria del software
 - 5.1.4. Il processo del software
 - 5.1.5. La pratica dell'ingegneria del software
 - 5.1.6. Miti del software
 - 5.1.7. Come tutto ha inizio
 - 5.1.8. Concetti orientati agli oggetti
 - 5.1.9. Introduzione a UML
- 5.2. Il processo del software
 - 5.2.1. Un modello generale di processo
 - 5.2.2. Modelli di processo prescrittivi
 - 5.2.3. Modelli di processo specializzati
 - 5.2.4. Il processo unificato
 - 5.2.5. Modelli di processo personali e di squadra
 - 5.2.6. Che cos'è l'agilità?
 - 5.2.7. Che cos'è un processo agile?
 - 5.2.8. Scrum
 - 5.2.9. Toolkit del processo agile
- 5.3. Principi che guidano la pratica dell'ingegneria del software
 - 5.3.1. Principi che guidano il processo
 - 5.3.2. Principi che guidano la pratica
 - 5.3.3. Principi di comunicazione
 - 5.3.4. Principi di pianificazione
 - 5.3.5. Principi di modellazione
 - 5.3.6. Principi di costruzione
 - 5.3.7. Principi di implementazione



- 5.4. Comprendere i requisiti
 - 5.4.1. Ingegneria dei requisiti
 - 5.4.2. Stabilire le basi
 - 5.4.3. Richiesta di requisiti
 - 5.4.4. Sviluppo di casi d'uso
 - 5.4.5. Elaborazione del modello dei requisiti
 - 5.4.6. Negoziazione dei requisiti
 - 5.4.7. Convalida dei requisiti
- 5.5. Modellazione dei requisiti: Scenari, informazioni e classi di analisi
 - 5.5.1. Analisi dei requisiti
 - 5.5.2. Modellazione basata su scenari
 - 5.5.3. Modelli UML che forniscono casi d'uso
 - 5.5.4. Concetti di modellazione dei dati
 - 5.5.5. Modellazione basata sulle classi
 - 5.5.6. Diagrammi di classe
- 5.6. Modellazione dei requisiti: Flusso, comportamento e modelli
 - 5.6.1. Requisiti di modellazione delle strategie
 - 5.6.2. Modellazione orientata al flusso
 - 5.6.3. Diagrammi di stato
 - 5.6.4. Creare un modello comportamentale
 - 5.6.5. Diagrammi di sequenza
 - 5.6.6. Diagrammi di comunicazione
 - 5.6.7. Pattern per la modellazione dei requisiti
- 5.7. Concetti di design
 - 5.7.1. La progettazione nel contesto dell'ingegneria del software
 - 5.7.2. Il processo di design
 - 5.7.3. Concetti di design
 - 5.7.4. Concetti di progettazione orientata agli oggetti
 - 5.7.5. Il modello di progettazione
- 5.8. Design dell'architettura
 - 5.8.1. Architettura del software
 - 5.8.2. Generi architettonici
 - 5.8.3. Stili architettonici
 - 5.8.4. Progettazione architettonica
 - 5.8.5. Evoluzione di progetti alternativi per l'architettura
 - 5.8.6. Mappare l'architettura utilizzando il flusso di dati
- 5.9. Progettazione a livello di componente e basata su pattern
 - 5.9.1. Che cos'è un componente?
 - 5.9.2. Progettazione di componenti basata su classi
 - 5.9.3. Realizzazione della progettazione a livello di componenti
 - 5.9.4. Design tradizionale dei componenti
 - 5.9.5. Sviluppo basato su componenti
 - 5.9.6. Pattern di progettazione
 - 5.9.7. Progettazione software basata su pattern
 - 5.9.8. Pattern architettonici
 - 5.9.9. Pattern di progettazione a livello di componente
 - 5.9.10. Pattern di progettazione dell'interfaccia utente
- 5.10. Qualità del software e gestione dei progetti
 - 5.10.1. Qualità
 - 5.10.2. Qualità del software
 - 5.10.3. Il dilemma della qualità del software
 - 5.10.4. Ottenere la qualità del software
 - 5.10.5. Garanzia di qualità del software
 - 5.10.6. Lo spettro gestionale
 - 5.10.7. Il personale
 - 5.10.8. Il prodotto
 - 5.10.9. Il processo
 - 5.10.10. Il progetto
 - 5.10.11. Principi e pratiche

Modulo 6. Motori dei videogiochi

- 6.1. Videogiochi e TIC
 - 6.1.1. Introduzione
 - 6.1.2. Opportunità
 - 6.1.3. Sfide
 - 6.1.4. Conclusioni
- 6.2. Storia dei motori per videogiochi
 - 6.2.1. Introduzione
 - 6.2.2. Era Atari
 - 6.2.3. Anni '80
 - 6.2.4. I primi motori. Anni '90
 - 6.2.5. Motori attuali
- 6.3. Motori dei videogiochi
 - 6.3.1. Tipi di motori
 - 6.3.2. Parti di un motore per videogiochi
 - 6.3.3. Motori attuali
 - 6.3.4. Selezione di un motore per il nostro progetto
- 6.4. *Motori Game Maker*
 - 6.4.1. Introduzione
 - 6.4.2. Progettazione di scenari
 - 6.4.3. Sprite e animazioni
 - 6.4.4. Collisioni
 - 6.4.5. *Scripting* in GML
- 6.5. Motore Unreal Engine 4: Introduzione
 - 6.5.1. Che cos'è Unreal Engine 4? Qual è la sua filosofia?
 - 6.5.2. Materiali
 - 6.5.3. UI
 - 6.5.4. Animazioni
 - 6.5.5. Sistema di particelle
 - 6.5.6. Intelligenza artificiale
 - 6.5.7. FPS
- 6.6. Motore Unreal Engine 4: *Visual Scripting*
 - 6.6.1. Filosofia dei *Blueprint* e il *Visual Scripting*
 - 6.6.2. *Debugging*
 - 6.6.3. Tipi di variabili
 - 6.6.4. Controllo del flusso di base
- 6.7. Motore Unity 5
 - 6.7.1. Programmazione in C# e Visual Studio
 - 6.7.2. Creazione di Prefabs
 - 6.7.3. Uso di gizmos per controllare il videogioco
 - 6.7.4. Motore adattivo: 2D e 3D
- 6.8. Motore Godot
 - 6.8.1. Filosofia del design Godot
 - 6.8.2. Progettazione e composizione orientata agli oggetti
 - 6.8.3. Pacchetto All-in-One
 - 6.8.4. Software libero e comunitario
- 6.9. Motore RPG Maker
 - 6.9.1. Filosofia dell'RPG Maker
 - 6.9.2. Come riferimento
 - 6.9.3. Creare un gioco con personalità
 - 6.9.4. Giochi commerciali di successo
- 6.10. Motore Source 2
 - 6.10.1. Filosofia di Source 2
 - 6.10.2. Source e Source 2: evoluzione
 - 6.10.3. Uso della Comunità: Contenuti audiovisivi e videogiochi
 - 6.10.4. Il futuro del motore Source 2
 - 6.10.5. Mod e giochi di successo

Modulo 7. Sistemi intelligenti

- 7.1. Teoria dell'agente
 - 7.1.1. Storia del concetto
 - 7.1.2. Definizione di agente
 - 7.1.3. Agenti nell'intelligenza artificiale
 - 7.1.4. Agenti nell'ingegneria del software
- 7.2. Architetture di agenti
 - 7.2.1. Il processo di ragionamento dell'agente
 - 7.2.2. Agenti reattivi
 - 7.2.3. Agenti deduttivi
 - 7.2.4. Agenti ibridi
 - 7.2.5. Confronto
- 7.3. Informazione e conoscenza
 - 7.3.1. Distinzione tra dati, informazioni e conoscenza
 - 7.3.2. Valutazione della qualità dei dati
 - 7.3.3. Metodi di acquisizione dei dati
 - 7.3.4. Metodi di acquisizione delle informazioni
 - 7.3.5. Metodi di acquisizione della conoscenza
- 7.4. Rappresentazione della conoscenza
 - 7.4.1. L'importanza della rappresentazione della conoscenza
 - 7.4.2. Definizione della rappresentazione della conoscenza attraverso i suoi ruoli
 - 7.4.3. Caratteristiche di una rappresentazione della conoscenza
- 7.5. Ontologie
 - 7.5.1. Introduzione ai metadati
 - 7.5.2. Concetto filosofico di ontologia
 - 7.5.3. Concetto informatico di ontologia
 - 7.5.4. Ontologie di dominio e di livello superiore
 - 7.5.5. Come costruire un'ontologia
- 7.6. Linguaggi ontologici e software per la creazione di ontologie
 - 7.6.1. Tripletta RDF, Turtle e N3
 - 7.6.2. Schema RDF
 - 7.6.3. OWL
 - 7.6.4. SPARQL
 - 7.6.5. Introduzione ai diversi strumenti per la creazione di ontologie
 - 7.6.6. Installazione e utilizzo di Protégé
- 7.7. Web semantico
 - 7.7.1. Stato attuale e futuro del Web semantico
 - 7.7.2. Applicazioni del Web semantico
- 7.8. Altri modelli di rappresentazione della conoscenza
 - 7.8.1. Vocabolari
 - 7.8.2. Visione globale
 - 7.8.3. Tassonomie
 - 7.8.4. Thesauri
 - 7.8.5. Folksonomie
 - 7.8.6. Confronto
 - 7.8.7. Mappe mentali
- 7.9. Valutazione e integrazione delle rappresentazioni della conoscenza
 - 7.9.1. Logica dell'ordine zero
 - 7.9.2. Logica del primo ordine
 - 7.9.3. Logica descrittiva
 - 7.9.4. Relazione tra i diversi tipi di logica
 - 7.9.5. Prolog: Programmazione basata sulla logica del primo ordine
- 7.10. Ragionatori semantici, sistemi basati sulla conoscenza e sistemi esperti
 - 7.10.1. Concetto di ragionatore
 - 7.10.2. Applicazioni di un ragionatore
 - 7.10.3. Sistemi basati sulla conoscenza
 - 7.10.4. MYCIN, storia dei sistemi esperti
 - 7.10.5. Elementi e architettura dei sistemi esperti
 - 7.10.6. Creazione di sistemi esperti

Modulo 8. Programmazione in tempo reale

- 8.1. Fondamenti di programmazione concorrente
 - 8.1.1. Concetti fondamentali
 - 8.1.2. Concorrenza
 - 8.1.3. Vantaggi della concorrenza
 - 8.1.4. Concorrenza e hardware
- 8.2. Strutture di supporto alla concorrenza di base in Java
 - 8.2.1. Concorrenza in Java
 - 8.2.2. Creazione di *Thread*
 - 8.2.3. Metodi
 - 8.2.4. Sincronizzazione
- 8.3. *Thread*, ciclo di vita, priorità, interruzioni, stato, esecutori
 - 8.3.1. *Thread*
 - 8.3.2. Ciclo di vita
 - 8.3.3. Priorità
 - 8.3.4. Interruzioni
 - 8.3.5. Stati
 - 8.3.6. Esecutori
- 8.4. Esclusione mutua
 - 8.4.1. Che cos'è l'esclusione mutua?
 - 8.4.2. Algoritmo di Dekker
 - 8.4.3. Algoritmo di Peterson
 - 8.4.4. Mutua esclusione in Java
- 8.5. Dipendenze di stato
 - 8.5.1. Iniezione di dipendenza
 - 8.5.2. Implementazione del pattern Java
 - 8.5.3. Modi per iniettare le dipendenze
 - 8.5.4. Esempio





- 8.6. Pattern di progettazione
 - 8.6.1. Introduzione
 - 8.6.2. Pattern di creazione
 - 8.6.3. Pattern di struttura
 - 8.6.4. Pattern comportamentali
- 8.7. Utilizzo delle librerie Java
 - 8.7.1. Cosa sono le librerie Java?
 - 8.7.2. Mockito-All, Mockito-Core
 - 8.7.3. Guava
 - 8.7.4. Commons-io
 - 8.7.5. Commons-Lang, Commons-Lang3
- 8.8. Programmazione degli shader
 - 8.8.1. Pipeline 3D y Raster
 - 8.8.2. Vertex Shading
 - 8.8.3. Pixel Shading: Illuminazione I
 - 8.8.4. Pixel Shading: Illuminazione II
 - 8.8.5. Post-Effects
- 8.9. Programmazione in tempo reale
 - 8.9.1. Introduzione
 - 8.9.2. Elaborazione degli interrupt
 - 8.9.3. Sincronizzazione e comunicazione tra processi
 - 8.9.4. Sistemi di programmazione in tempo reale
- 8.10. Pianificazione in tempo reale
 - 8.10.1. Concetti
 - 8.10.2. Modello di riferimento per i sistemi in tempo reale
 - 8.10.3. Politiche di pianificazione
 - 8.10.4. Pianificatori ciclici
 - 8.10.5. Pianificatori con proprietà statiche
 - 8.10.6. Pianificatori con proprietà dinamiche

Modulo 9. Progettazione e sviluppo dei browser game

- 9.1. Origini e standard del Web
 - 9.1.1. Le origini di Internet
 - 9.1.2. Creazione del World Wide Web
 - 9.1.3. Nascita degli standard web
 - 9.1.4. L'ascesa degli standard web
- 9.2. HTTP e struttura client-server
 - 9.2.1. Ruolo client-server
 - 9.2.2. Comunicazione client-server
 - 9.2.3. Storia recente
 - 9.2.4. Informatica centralizzata
- 9.3. Programmazione web: Introduzione
 - 9.3.1. Concetti di base
 - 9.3.2. Preparazione di un server web
 - 9.3.3. Nozioni di base di HTML5
 - 9.3.4. Moduli HTML
- 9.4. Introduzione all'HTML ed esempi
 - 9.4.1. Storia di HTML5
 - 9.4.2. Elementi di HTML5
 - 9.4.3. APIS
 - 9.4.4. CCS3
- 9.5. Modello a oggetti del documento
 - 9.5.1. Che cos'è il modello a oggetti del documento?
 - 9.5.2. Uso di DOCTYPE
 - 9.5.3. L'importanza della validazione dell'HTML
 - 9.5.4. Accesso agli elementi
 - 9.5.5. Creazione di elementi e testo
 - 9.5.6. Utilizzo di InnerHTML
 - 9.5.7. Eliminazione di un elemento o di un nodo di testo
 - 9.5.8. Lettura e scrittura degli attributi degli elementi
 - 9.5.9. Manipolazione degli stili degli elementi
 - 9.5.10. Allegare più file contemporaneamente
- 9.6. Introduzione all'CSS ed esempi
 - 9.6.1. Sintassi CSS3
 - 9.6.2. Fogli di stile
 - 9.6.3. Tag
 - 9.6.4. Selezionatori
 - 9.6.5. Progettazione web con CSS
- 9.7. Introduzione a Javascript ed esempi
 - 9.7.1. Che cos'è Javascript?
 - 9.7.2. Breve storia del linguaggio
 - 9.7.3. Versioni di Javascript
 - 9.7.4. Visualizzazione di una finestra di dialogo
 - 9.7.5. Sintassi Javascript
 - 9.7.6. Capire gli *Scripts*
 - 9.7.7. Spazi
 - 9.7.8. Commenti
 - 9.7.9. Funzioni
 - 9.7.10. JavaScript interno ed esterno alla pagina
- 9.8. Funzioni Javascript
 - 9.8.1. Dichiarazioni di funzione
 - 9.8.2. Espressioni di funzione
 - 9.8.3. Chiamare le funzioni
 - 9.8.4. Ricorsione
 - 9.8.5. Funzioni annidate e chiusure
 - 9.8.6. Conservazione delle variabili
 - 9.8.7. Funzioni multi-nidificate
 - 9.8.8. Conflitti di nome
 - 9.8.9. Chiusure
 - 9.8.10. Parametri di una funzione

- 9.9. PlayCanvas per lo sviluppo di browser game
 - 9.9.1. Che cos'è PlayCanvas?
 - 9.9.2. Configurazione del progetto
 - 9.9.3. Creare un oggetto
 - 9.9.4. Aggiunta di fisiche
 - 9.9.5. Aggiunta di un modello
 - 9.9.6. Modifica delle impostazioni di gravità e di scena
 - 9.9.7. Esecuzione di *Script*
 - 9.9.8. Controlli della telecamera
- 9.10. Phaser per lo sviluppo di browser game
 - 9.10.1. Che cos'è Phaser?
 - 9.10.2. Caricamento delle risorse
 - 9.10.3. Costruire il mondo
 - 9.10.4. Piattaforme
 - 9.10.5. Il giocatore
 - 9.10.6. Aggiungere fisiche
 - 9.10.7. Utilizzo della tastiera
 - 9.10.8. Raccogliere *Pickup*
 - 9.10.9. Punti e punteggi
 - 9.10.10. Bouncing bombs

Modulo 10. Reti e sistemi multigiocatore

- 10.1. Storia ed evoluzione dei videogiochi multiplayer
 - 10.1.1. Decennio 1970: I primi giochi multiplayer
 - 10.1.2. Anni 90: Duke Nukem, Doom, Quake
 - 10.1.3. L'ascesa dei videogiochi multiplayer
 - 10.1.4. Multiplayer locale e online
 - 10.1.5. Giochi per feste
- 10.2. Modelli di business multiplayer
 - 10.2.1. Origine e funzionamento dei modelli di business emergenti
 - 10.2.2. Servizi di vendita online
 - 10.2.3. Free to Play
 - 10.2.4. Micropagamenti
 - 10.2.5. Pubblicità
 - 10.2.6. Abbonamento con pagamenti mensili
 - 10.2.7. Pay to play
 - 10.2.8. Provare prima di acquistare
- 10.3. Videogiochi locali e videogiochi in rete
 - 10.3.1. Videogiochi locali: gli inizi
 - 10.3.2. Giochi per feste: Nintendo e la famiglia
 - 10.3.3. Giochi in rete: inizi
 - 10.3.4. Evoluzione dei giochi in rete
- 10.4. Modello OSI: Livelli I
 - 10.4.1. Modello OSI: Introduzione
 - 10.4.2. Livello fisico
 - 10.4.3. Livello di collegamento dati
 - 10.4.4. Livello di rete
- 10.5. Modello OSI: Livelli II
 - 10.5.1. Livello di trasporto
 - 10.5.2. Livello di sessione
 - 10.5.3. Livello di presentazione
 - 10.5.4. Livello applicazione

- 10.6. Reti di computer e Internet
 - 10.6.1. Che cos'è una rete di computer?
 - 10.6.2. Software
 - 10.6.3. Hardware
 - 10.6.4. Server
 - 10.6.5. Archiviazione di rete
 - 10.6.6. Protocolli di rete
- 10.7. Reti mobili e wireless
 - 10.7.1. Rete mobile
 - 10.7.2. Rete wireless
 - 10.7.3. Funzionamento delle reti mobili
 - 10.7.4. Tecnologia digitale
- 10.8. Sicurezza
 - 10.8.1. Sicurezza personale
 - 10.8.2. Hack e trucchi nei videogiochi
 - 10.8.3. Sicurezza anti-cheat
 - 10.8.4. Analisi dei sistemi di sicurezza anti-cheat
- 10.9. Sistemi multigiocatore: Server
 - 10.9.1. Hosting server
 - 10.9.2. Videogiochi MMO
 - 10.9.3. Server dedicati ai videogiochi
 - 10.9.4. LAN Parties
- 10.10. Progettazione e programmazione di giochi multiplayer
 - 10.10.1. Fondamenti di progettazione di giochi multiplayer Unreal
 - 10.10.2. Fondamenti di progettazione di giochi multiplayer in Unity
 - 10.10.3. Come rendere divertente un gioco multiplayer?
 - 10.10.4. Oltre il controller Innovazione nei controller multigiocatore





“

Se vuoi intraprendere una grande carriera nella programmazione di videogiochi di fama mondiale, questa è la qualifica che fa per te”

05 Metodologia

Questo programma ti offre un modo differente di imparare. La nostra metodologia si sviluppa in una modalità di apprendimento ciclico: ***il Relearning***.

Questo sistema di insegnamento viene applicato nelle più prestigiose facoltà di medicina del mondo ed è considerato uno dei più efficaci da importanti pubblicazioni come il ***New England Journal of Medicine***.



“

Scopri il Relearning, un sistema che abbandona l'apprendimento lineare convenzionale, per guidarti attraverso dei sistemi di insegnamento ciclici: una modalità di apprendimento che ha dimostrato la sua enorme efficacia, soprattutto nelle materie che richiedono la memorizzazione”

Caso di Studio per contestualizzare tutti i contenuti

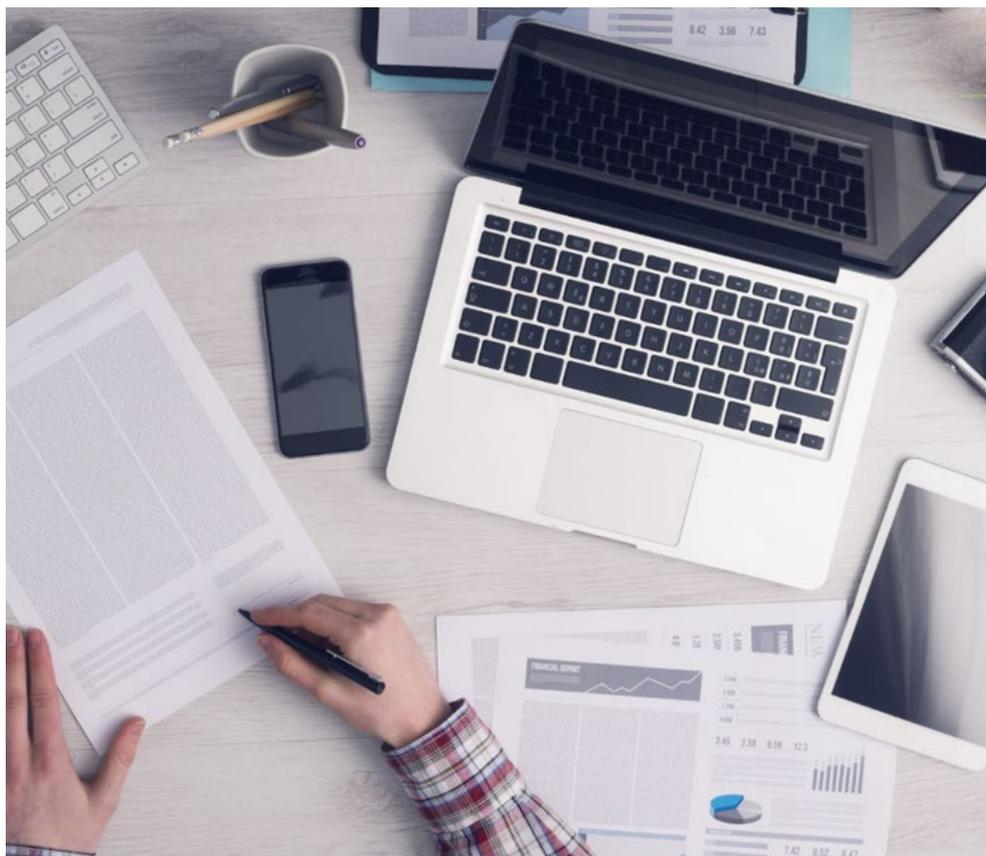
Il nostro programma offre un metodo rivoluzionario per sviluppare le abilità e le conoscenze. Il nostro obiettivo è quello di rafforzare le competenze in un contesto mutevole, competitivo e altamente esigente.

“

Con TECH potrai sperimentare un modo di imparare che sta scuotendo le fondamenta delle università tradizionali in tutto il mondo"



Avrai accesso a un sistema di apprendimento basato sulla ripetizione, con un insegnamento naturale e progressivo durante tutto il programma.



Imparerai, attraverso attività collaborative e casi reali, la risoluzione di situazioni complesse in ambienti aziendali reali.

Un metodo di apprendimento innovativo e differente

Questo programma di TECH consiste in un insegnamento intensivo, creato ex novo, che propone le sfide e le decisioni più impegnative in questo campo, sia a livello nazionale che internazionale. Grazie a questa metodologia, la crescita personale e professionale viene potenziata, effettuando un passo decisivo verso il successo. Il metodo casistico, la tecnica che sta alla base di questi contenuti, garantisce il rispetto della realtà economica, sociale e professionale più attuali.

“

Il nostro programma ti prepara ad affrontare nuove sfide in ambienti incerti e a raggiungere il successo nella tua carriera”

Il Metodo Casistico è stato il sistema di apprendimento più usato nelle migliori Scuole di Informatica del mondo da quando esistono. Sviluppato nel 1912 affinché gli studenti di Diritto non imparassero la legge solo sulla base del contenuto teorico, il metodo casistico consisteva nel presentare loro situazioni reali e complesse per prendere decisioni informate e giudizi di valore su come risolverle. Nel 1924 fu stabilito come metodo di insegnamento standard ad Harvard.

Cosa dovrebbe fare un professionista per affrontare una determinata situazione?

Questa è la domanda con cui ti confrontiamo nel metodo dei casi, un metodo di apprendimento orientato all'azione. Durante il corso, gli studenti si confronteranno con diversi casi di vita reale. Dovranno integrare tutte le loro conoscenze, effettuare ricerche, argomentare e difendere le proprie idee e decisioni.

Metodologia Relearning

TECH coniuga efficacemente la metodologia del Caso di Studio con un sistema di apprendimento 100% online basato sulla ripetizione, che combina diversi elementi didattici in ogni lezione.

Potenziamo il Caso di Studio con il miglior metodo di insegnamento 100% online: il Relearning.

Nel 2019 abbiamo ottenuto i migliori risultati di apprendimento di tutte le università online del mondo.

In TECH imparerai con una metodologia all'avanguardia progettata per formare i manager del futuro. Questo metodo, all'avanguardia della pedagogia mondiale, si chiama Relearning.

La nostra università è l'unica autorizzata a utilizzare questo metodo di successo. Nel 2019, siamo riusciti a migliorare il livello di soddisfazione generale dei nostri studenti (qualità dell'insegnamento, qualità dei materiali, struttura del corso, obiettivi...) rispetto agli indicatori della migliore università online.



Nel nostro programma, l'apprendimento non è un processo lineare, ma avviene in una spirale (impariamo, disimpariamo, dimentichiamo e re-impariamo). Pertanto, combiniamo ciascuno di questi elementi in modo concentrico. Questa metodologia ha formato più di 650.000 laureati con un successo senza precedenti in campi diversi come la biochimica, la genetica, la chirurgia, il diritto internazionale, le competenze manageriali, le scienze sportive, la filosofia, il diritto, l'ingegneria, il giornalismo, la storia, i mercati e gli strumenti finanziari. Tutto questo in un ambiente molto esigente, con un corpo di studenti universitari con un alto profilo socio-economico e un'età media di 43,5 anni.

Il Relearning ti permetterà di apprendere con meno sforzo e più performance, impegnandoti maggiormente nella tua specializzazione, sviluppando uno spirito critico, difendendo gli argomenti e contrastando le opinioni: un'equazione diretta al successo.

Dalle ultime evidenze scientifiche nel campo delle neuroscienze, non solo sappiamo come organizzare le informazioni, le idee, le immagini e i ricordi, ma sappiamo che il luogo e il contesto in cui abbiamo imparato qualcosa è fondamentale per la nostra capacità di ricordarlo e immagazzinarlo nell'ippocampo, per conservarlo nella nostra memoria a lungo termine.

In questo modo, e in quello che si chiama Neurocognitive Context-dependent E-learning, i diversi elementi del nostro programma sono collegati al contesto in cui il partecipante sviluppa la sua pratica professionale.



Questo programma offre i migliori materiali didattici, preparati appositamente per i professionisti:



Materiali di studio

Tutti i contenuti didattici sono creati appositamente per il corso dagli specialisti che lo impartiranno, per fare in modo che lo sviluppo didattico sia davvero specifico e concreto.

Questi contenuti sono poi applicati al formato audiovisivo che supporterà la modalità di lavoro online di TECH. Tutto questo, con le ultime tecniche che offrono componenti di alta qualità in ognuno dei materiali che vengono messi a disposizione dello studente.



Master class

Esistono evidenze scientifiche sull'utilità dell'osservazione di esperti terzi.

Imparare da un esperto rafforza la conoscenza e la memoria, costruisce la fiducia nelle nostre future decisioni difficili.



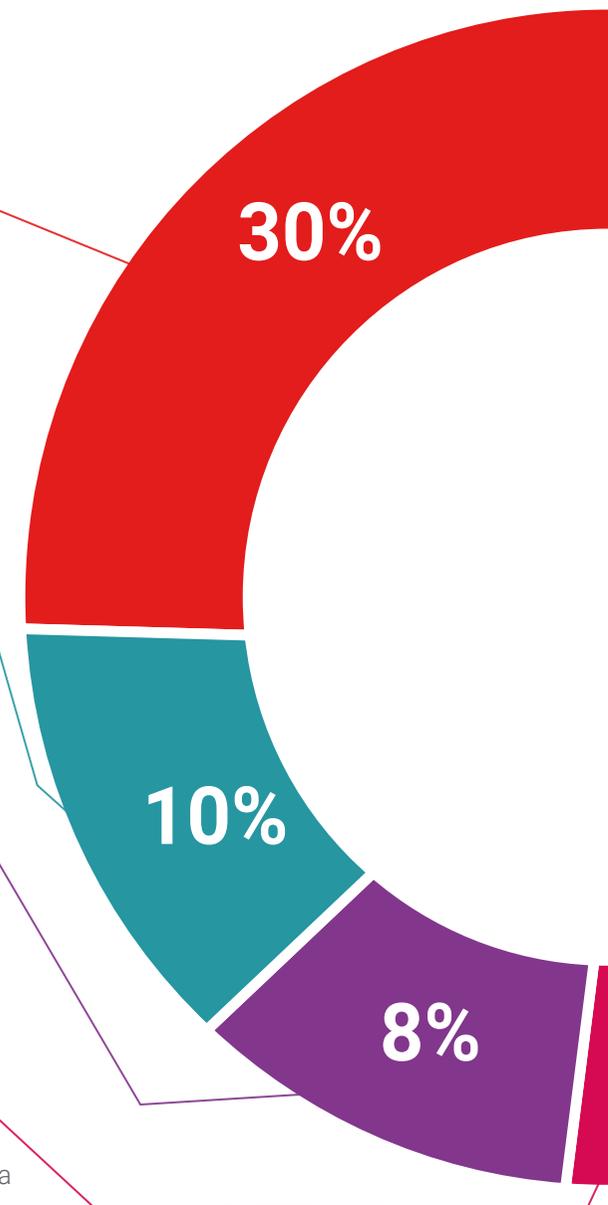
Pratiche di competenze e competenze

Svolgerai attività per sviluppare competenze e capacità specifiche in ogni area tematica. Pratiche e dinamiche per acquisire e sviluppare le competenze e le abilità che uno specialista deve sviluppare nel quadro della globalizzazione in cui viviamo.



Letture complementari

Articoli recenti, documenti di consenso e linee guida internazionali, tra gli altri. Nella biblioteca virtuale di TECH potrai accedere a tutto il materiale necessario per completare la tua specializzazione.





Casi di Studio

Completerai una selezione dei migliori casi di studio scelti appositamente per questo corso. Casi presentati, analizzati e monitorati dai migliori specialisti del panorama internazionale.



Riepiloghi interattivi

Il team di TECH presenta i contenuti in modo accattivante e dinamico in pillole multimediali che includono audio, video, immagini, diagrammi e mappe concettuali per consolidare la conoscenza.

Questo esclusivo sistema di specializzazione per la presentazione di contenuti multimediali è stato premiato da Microsoft come "Caso di successo in Europa".



Testing & Retesting

Valutiamo e rivalutiamo periodicamente le tue conoscenze durante tutto il programma con attività ed esercizi di valutazione e autovalutazione, affinché tu possa verificare come raggiungi progressivamente i tuoi obiettivi.



06 Titolo

Il Master Privato in Programmazione di Videogiochi ti garantisce, oltre alla formazione più rigorosa e aggiornata, l'accesso al Master Privato rilasciato dalla TECH Università Tecnologica.



“

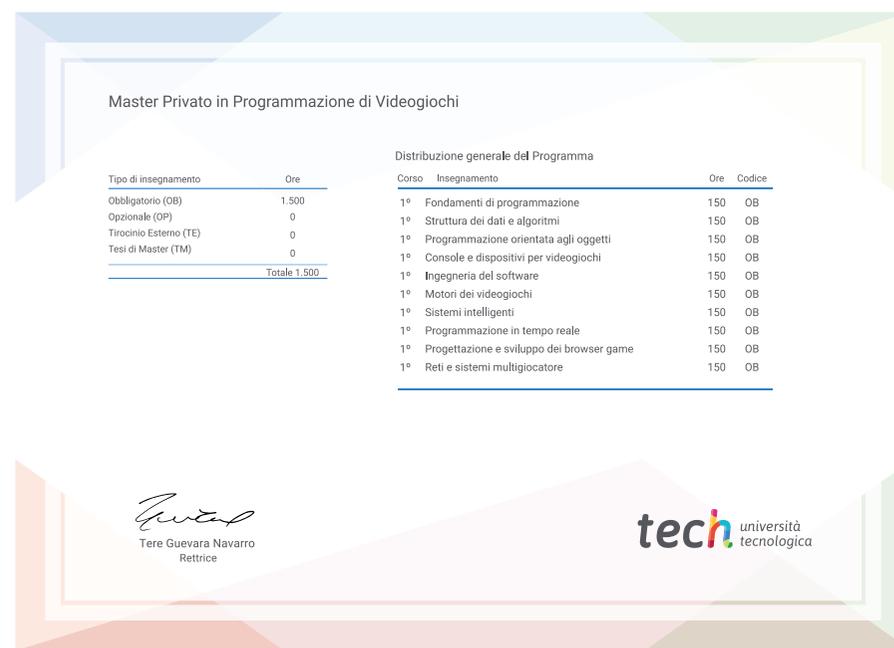
“Porta a termine questo programma e ricevi la tua qualifica universitaria senza spostamenti o fastidiose formalità”

Questo **Master Privato in Programmazione di Videogiochi** possiede il programma più completo e aggiornato presente sul mercato.

Dopo aver superato le valutazioni, lo studente riceverà, mediante lettera certificata* con ricevuta di ritorno, il suo corrispondente titolo **Master Privato** rilasciato da **TECH Università Tecnologica**.

Il titolo rilasciato da **TECH Università Tecnologica** indica la qualifica ottenuta nel Master Privato, e riunisce tutti i requisiti comunemente richiesti da borse di lavoro, concorsi e commissioni di valutazione di carriere professionali.

Titolo: **Master Privato in Programmazione di Videogiochi**
N° Ore Ufficiali: **1.500 O.**



*Se lo studente dovesse richiedere che il suo diploma cartaceo sia provvisto di Apostille dell'Aia, TECH EDUCATION effettuerà le gestioni opportune per ottenerla pagando un costo aggiuntivo.

futuro
salute fiducia persone
educazione informazione tutor
garanzia accreditamento insegnamento
istituzioni tecnologia apprendimento
comunità impegno
attenzione personalizzata inn
conoscenza presente qualità
formazione online
sviluppo istituzioni
classe virtuale lingu

tech università
tecnologica

Master Privato
Programmazione
di Videogiochi

- » Modalità: online
- » Durata: 12 mesi
- » Titolo: TECH Università Tecnologica
- » Dedizione: 16 ore/settimana
- » Orario: a scelta
- » Esami: online

Master Privato

Programmazione di Videogiochi

