Image

Storia della programmazione

Le origini: Antiche forme di programmazione

Nel corso della storia, l'umanità ha sviluppato diverse forme di programmazione molto prima dell'avvento dei computer moderni. Queste prime forme di programmazione coinvolgevano l'utilizzo di dispositivi meccanici per eseguire calcoli e automatizzare compiti specifici. Uno dei dispositivi più antichi utilizzati per la programmazione era l'abaco, un semplice strumento di calcolo composto da perline o pietre mobili disposte su fili o aste.

Oltre all'abaco, anche i telai da tessitura hanno rappresentato un importante passo verso la programmazione. I tessitori utilizzavano sistemi di cartoni forati chiamati "carte perforate" per creare disegni complessi sui tessuti. Queste carte perforate fungevano da programmi, indicando al telaio da tessitura i passi specifici da seguire per creare i disegni desiderati.

Ada Lovelace e il suo contributo pionieristico alla programmazione

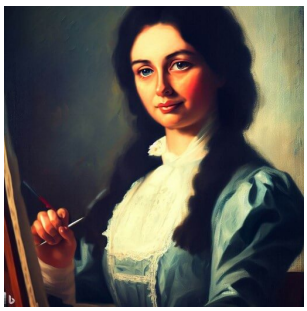
Ada Lovelace, nata Ada Byron, è considerata la prima programmatrice al mondo. Nel XIX secolo, Ada lavorò con Charles Babbage, un matematico e inventore britannico, sul suo progetto di macchina analitica, un precursore del computer moderno. Durante la sua collaborazione con Babbage, Ada scrisse quello che viene considerato il primo algoritmo mai creato.

Il suo algoritmo, sviluppato per essere eseguito sulla macchina analitica, era progettato per calcolare i numeri di Bernoulli. Tuttavia, Ada andò oltre e riconobbe il potenziale delle macchine analitiche per elaborare non solo numeri, ma anche simboli e concetti astratti. Le sue intuizioni la portarono a considerare le macchine analitiche come strumenti capaci di svolgere compiti molto più complessi di

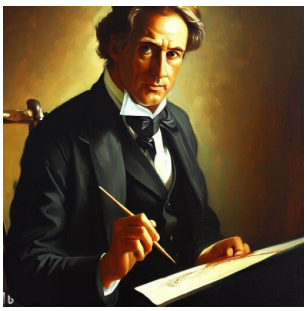
semplici calcoli matematici.

L'opera di Ada Lovelace ha avuto un impatto significativo sullo sviluppo della programmazione moderna. Il suo lavoro pionieristico ha contribuito a gettare le basi per l'idea di utilizzare le macchine per svolgere compiti diversi dalla semplice elaborazione numerica, aprendo la strada alla programmazione come la conosciamo oggi.

Ada Lovelace (by DALL·E)



Charles Babbage (By DALL·E)

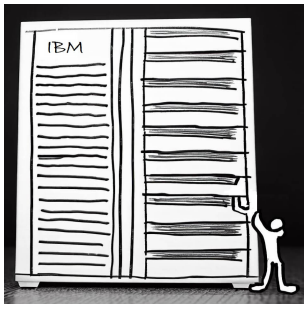


I computer primitivi: ENIAC, UNIVAC e l'avvento dell'elettronica digitale

Negli anni '40, il mondo vide l'emergere dei primi computer elettronici. Uno dei primi computer di grande rilevanza fu l'ENIAC (Electronic Numerical Integrator and Computer), sviluppato negli Stati Uniti durante la Seconda Guerra Mondiale. L'ENIAC fu il primo computer completamente elettronico e fu utilizzato per calcoli complessi e per scopi militari.

Successivamente, l'UNIVAC (Universal Automatic Computer) divenne uno dei primi computer commerciali. Fu progettato per scopi più generali rispetto all'ENIAC e segnò un importante passo avanti nell'evoluzione dei computer. L'UNIVAC introduceva l'uso di memorie magnetiche, consentendo una maggiore capacità di archiviazione dei dati.

L'avvento dell'elettronica digitale rese possibile la realizzazione di computer più compatti, potenti ed efficienti. Questo passaggio fondamentale nell'evoluzione dei computer aprì la strada a nuove applicazioni e possibilità di calcolo.



Linguaggi di programmazione dei primi computer: Assembly, linguaggi a basso livello

I primi computer venivano programmati utilizzando linguaggi di basso livello, come l'Assembly. L'Assembly era un linguaggio di programmazione molto vicino al linguaggio macchina, basato su istruzioni specifiche per l'hardware del computer. I programmatori dovevano scrivere il codice Assembly direttamente in base alle specifiche del computer e delle sue istruzioni.

I linguaggi di basso livello erano potenti ma richiedevano un'alta competenza tecnica e una conoscenza dettagliata dell'architettura del computer. Gli sviluppatori dovevano gestire direttamente i registri, le istruzioni di memoria e le operazioni di input/output.

Nonostante le sfide presenti, *l'utilizzo di linguaggi di basso livello ha permesso ai programmatori di sfruttare appieno le potenzialità dei primi computer*, aprendo la strada a nuove applicazioni e avanzamenti nell'elaborazione dei dati.

Durante gli anni '50 e '60, l'industria informatica ha conosciuto l'era dei computer mainframe, macchine potenti e di grandi dimensioni che fornivano servizi di elaborazione dati a grandi organizzazioni. Questa lezione esplorerà l'importanza dei computer mainframe e l'introduzione del linguaggio di programmazione FORTRAN.

L'era dei computer mainframe

Gli anni '50 e '60 sono stati caratterizzati dall'uso diffuso dei computer mainframe. Questi enormi sistemi informatici, spesso custoditi all'interno di grandi sale dedicate, erano utilizzati da aziende, istituzioni governative e organizzazioni scientifiche per elaborare grandi quantità di dati e risolvere problemi complessi.

I computer mainframe erano dotati di potenti processori, capacità di archiviazione notevole e supportavano un gran numero di utenti simultanei. Essi fornivano servizi di calcolo, archiviazione e

condivisione dei dati su larga scala, rappresentando un elemento chiave nell'evoluzione della tecnologia informatica.

Il linguaggio di programmazione FORTRAN

Nel contesto dell'era dei computer mainframe, il linguaggio di programmazione FORTRAN (Formula Translation) ha svolto un ruolo fondamentale. Creato da un team di ricercatori presso IBM negli anni '50, FORTRAN è stato il primo linguaggio di programmazione ad alto livello, progettato specificamente per calcoli scientifici e ingegneristici.

FORTRAN semplificava la scrittura di programmi complessi, offrendo una sintassi più vicina all'inglese e introducendo funzionalità di calcolo avanzate. Questo linguaggio ha consentito ai ricercatori di scrivere programmi più facilmente comprensibili e di sfruttare al meglio le potenzialità dei computer mainframe per risolvere problemi scientifici e di ingegneria.

L'evoluzione dei linguaggi di programmazione: COBOL, ALGOL, Lisp

Con il passare degli anni, l'industria informatica ha assistito a un'evoluzione dei linguaggi di programmazione, mirata a soddisfare le esigenze di diverse applicazioni e settori. Questa sezione della lezione esplorerà tre importanti linguaggi di programmazione che hanno contribuito a definire l'evoluzione della programmazione.

1. COBOL (Common Business-Oriented Language): COBOL è stato sviluppato negli anni '60 ed è stato progettato per l'elaborazione di dati aziendali e finanziari. È stato il primo linguaggio di programmazione ad alto livello ad essere orientato al mondo degli affari, offrendo funzionalità specifiche per la gestione dei dati tabulari e delle operazioni bancarie.

2. ALGOL (ALGOrithmic Language): ALGOL è stato sviluppato negli anni '60 ed è stato uno dei primi linguaggi di programmazione ad alto livello a introdurre concetti fondamentali della programmazione strutturata. ALGOL ha influenzato lo sviluppo di molti altri linguaggi successivi ed è stato ampiamente utilizzato nel campo della ricerca scientifica e dell'informatica teorica.

3. Lisp (LISt Processing): Lisp è stato sviluppato negli anni '50 ed è noto per essere il secondo linguaggio di programmazione ad alto livello mai creato. Lisp si distingue per il suo approccio basato su liste e per la sua flessibilità nell'elaborazione simbolica. È stato ampiamente utilizzato nel campo dell'intelligenza artificiale e ha influenzato lo sviluppo di molti altri linguaggi di

programmazione.

Rivoluzione informatica: l'avvento dei personal computer

Negli anni '70, l'avvento dei personal computer ha segnato una vera e propria rivoluzione informatica. Questa lezione esplorerà l'importanza dei personal computer e l'introduzione del linguaggio di programmazione BASIC. Con l'introduzione dei personal computer, i computer sono diventati accessibili al grande pubblico. Queste macchine, compatte e convenienti, hanno portato la potenza del calcolo e dell'elaborazione dati nelle case e negli uffici di persone comuni. L'era dei personal computer ha aperto nuove possibilità e ha contribuito a diffondere l'informatica a livello globale. I personal computer erano dotati di processori più piccoli ma potenti, capacità di archiviazione migliorata e un'interfaccia utente intuitiva che consentiva a chiunque di interagire con la macchina. L'importanza dei personal computer risiede nella democratizzazione dell'informatica, offrendo alle persone la possibilità di eseguire calcoli, creare documenti e persino scrivere programmi direttamente dal proprio computer domestico.

Il linguaggio di programmazione BASIC

Con l'avvento dei personal computer, è emersa la necessità di un linguaggio di programmazione accessibile e intuitivo. Il linguaggio BASIC (Beginner's All-purpose Symbolic Instruction Code) è stato sviluppato per soddisfare questa esigenza. Progettato da John Kemeny e Thomas Kurtz nel 1964, BASIC è stato creato per essere semplice da imparare e da utilizzare, consentendo a chiunque di

scrivere programmi senza dover affrontare complessità e sintassi esoteriche.

BASIC ha fornito un punto di ingresso per molte persone nel mondo della programmazione, consentendo loro di sperimentare e imparare a creare programmi personalizzati per i loro computer. Questo linguaggio ha giocato un ruolo significativo nell'educazione informatica e nella diffusione della programmazione tra il grande pubblico. Molti appassionati di computer degli anni '70 e '80 hanno iniziato a scrivere i loro primi programmi in BASIC, aprendo la strada a una nuova generazione di programmatori.

Introduzione dei linguaggi di programmazione di alto livello come C e Pascal.

Con il passare degli anni, i linguaggi di programmazione hanno continuato a evolversi per soddisfare le esigenze sempre crescenti degli sviluppatori e delle applicazioni informatiche. Questa sezione della lezione esplorerà l'introduzione dei linguaggi di programmazione di alto livello come C e Pascal.

Linguaggio di programmazione C

Negli anni '70, il linguaggio di programmazione C è stato introdotto da Dennis Ritchie presso i laboratori Bell. C è stato progettato per fornire un linguaggio di programmazione efficiente e flessibile, adatto per lo sviluppo di sistemi operativi e applicazioni di basso livello. C ha introdotto concetti come puntatori e gestione manuale della memoria, fornendo un alto livello di controllo e prestazioni.

Il linguaggio C ha avuto un impatto significativo nell'industria informatica ed è diventato uno dei linguaggi di programmazione più popolari di tutti i tempi. È stato utilizzato per lo sviluppo di sistemi operativi come Unix e ha influenzato il design di molti altri linguaggi successivi, come C++, Java e Python.

Linguaggio di programmazione Pascal

Nel 1970, il linguaggio di programmazione Pascal è stato sviluppato da Niklaus Wirth. Pascal era un linguaggio di programmazione strutturato progettato per favorire la chiarezza e la comprensibilità del codice. È stato ampiamente utilizzato nell'insegnamento dell'informatica e nella scrittura di programmi scientifici e didattici.

Pascal ha introdotto molti concetti importanti della programmazione strutturata, come le procedure e le funzioni, che hanno contribuito a migliorare l'organizzazione e la manutenibilità del codice. Anche

se Pascal non è diventato un linguaggio di programmazione di largo uso come C, ha svolto un ruolo significativo nello sviluppo di linguaggi successivi come Delphi. L'introduzione dei linguaggi di programmazione di alto livello come C e Pascal ha portato a una maggiore espressività e facilità di sviluppo. Questi linguaggi hanno aperto nuove possibilità per gli sviluppatori e hanno contribuito a definire le basi della programmazione moderna. Oggi, molti dei principi introdotti da questi linguaggi sono ancora ampiamente utilizzati e continuano a influenzare il mondo della programmazione.

L'evoluzione della programmazione orientata agli oggetti

La programmazione orientata agli oggetti è emersa negli anni '60 e '70 come approccio innovativo alla progettazione del software. Si basa sul concetto di "oggetti" che racchiudono dati e comportamenti correlati. La POO offre un modo modulare e organizzato per creare programmi complessi, consentendo una maggiore riusabilità del codice e una migliore gestione delle interazioni tra gli oggetti.

L'introduzione di linguaggi di programmazione specifici per la POO ha contribuito alla sua adozione diffusa. Un esempio chiave è C++, sviluppato da Bjarne Stroustrup negli anni '80. C++ è una versione estesa del linguaggio C che incorpora concetti di programmazione orientata agli oggetti come classi, ereditarietà e polimorfismo. C++ è diventato un linguaggio di programmazione molto popolare e ha aperto la strada a una nuova era nella programmazione.

Successivamente, nel 1995, Java è stato introdotto da Sun Microsystems. Java è stato progettato per essere portabile, sicuro e versatile. È stato uno dei primi linguaggi di programmazione a utilizzare una macchina virtuale, consentendo ai programmi scritti in Java di essere eseguiti su diverse piattaforme senza dover essere riscritti. Java ha trovato ampia applicazione nello sviluppo di applicazioni web, software di sistema e dispositivi mobili.

Negli ultimi anni, Python ha guadagnato popolarità come linguaggio di programmazione orientato agli oggetti. Python è noto per la sua sintassi chiara e leggibile, rendendolo adatto sia per i principianti che per i professionisti. Ha una vasta gamma di librerie e framework che lo rendono adatto per una varietà di applicazioni, dal data analysis all'intelligenza artificiale.

L'impatto del World Wide Web sulla programmazione e lo sviluppo dei linguaggi web

L'avvento del World Wide Web ha avuto un profondo impatto sulla programmazione e lo sviluppo dei linguaggi web. Questa sezione della lezione esplorerà come il web ha influenzato la programmazione

e introdotto linguaggi chiave come HTML, CSS e JavaScript.

L'HTML (HyperText Markup Language) è il linguaggio di markup fondamentale per la creazione di pagine web. È stato sviluppato da Tim Berners-Lee negli anni '90 come linguaggio standard per la condivisione di informazioni ipertestuali. HTML definisce la struttura e il contenuto di una pagina web, consentendo di organizzare e formattare testo, immagini, collegamenti ipertestuali e altri elementi.

CSS (Cascading Style Sheets) è un linguaggio di fogli di stile utilizzato per controllare l'aspetto e il layout delle pagine web. CSS fornisce una separazione tra la struttura (HTML) e la presentazione (CSS) di una pagina, consentendo di definire stili, colori, dimensioni e posizionamento degli elementi.

JavaScript è un linguaggio di programmazione interpretato che aggiunge interattività e funzionalità dinamiche alle pagine web. È stato sviluppato da Brendan Eich nel 1995 e offre la possibilità di manipolare il contenuto HTML, gestire eventi, creare animazioni e molto altro. JavaScript ha svolto un ruolo chiave nello sviluppo di applicazioni web complesse e interattive.

L'avvento del World Wide Web ha creato una domanda sempre crescente di sviluppatori web e ha spinto allo sviluppo continuo di nuovi linguaggi, framework e tecnologie per soddisfare le esigenze degli utenti e delle aziende. HTML, CSS e JavaScript sono diventati fondamentali per la creazione di pagine web moderne e l'esperienza utente interattiva.

Programmazione funzionale: concetti e linguaggi come Lisp, Haskell e Scala

Nel campo della programmazione, sono emersi diversi paradigmi che offrono approcci innovativi per lo sviluppo del software. Questa lezione esplorerà tre paradigmi di programmazione moderni: la programmazione funzionale, la programmazione logica e i paradigmi emergenti come la programmazione reattiva e la programmazione parallela. La programmazione funzionale è un paradigma che si basa sul concetto di funzioni pure e immutabilità dei dati. Invece di modificare lo stato dei dati, si applicano funzioni per ottenere nuovi risultati. Questo approccio promuove il

ragionamento dichiarativo e la composizione delle funzioni per risolvere i problemi.

Lisp è uno dei primi linguaggi di programmazione funzionale, sviluppato negli anni '50. Si distingue per la sua flessibilità e potenza espressiva, consentendo di manipolare il codice come dati. Lisp ha influenzato molti altri linguaggi di programmazione funzionale successivi.

Haskell è un linguaggio di programmazione funzionale puro, caratterizzato da un forte sistema di tipi statico e da una sintassi elegante. Haskell offre strumenti per la gestione dei casi di eccezione e delle operazioni di input/output in modo funzionale. È ampiamente utilizzato in ambiti come l'elaborazione dei linguaggi naturali, la crittografia e l'intelligenza artificiale.

Scala è un linguaggio che combina elementi della programmazione funzionale e della programmazione orientata agli oggetti. Scala è compatibile con la piattaforma Java e offre funzionalità avanzate come la gestione degli errori tramite il tipo di dato "Option" e la concorrenza con gli attori.

Programmazione logica: Prolog e il ragionamento automatizzato

La programmazione logica è un paradigma che si basa sulla logica formale e sul ragionamento automatizzato. Un linguaggio rappresentativo di questo paradigma è Prolog. Prolog consente di definire fatti e regole logiche e di interrogare la base di conoscenza per ottenere risposte. Questo paradigma si presta particolarmente bene alla risoluzione di problemi basati su regole e relazioni.

Prolog ha trovato applicazioni in diversi campi, come l'intelligenza artificiale, la programmazione di agenti intelligenti e l'elaborazione del linguaggio naturale. Offre un approccio dichiarativo alla programmazione, in cui l'attenzione si concentra sulla specifica del problema piuttosto che sull'implementazione dettagliata.

Paradigmi emergenti: programmazione reattiva e programmazione parallela

Oltre ai paradigmi tradizionali, sono emersi nuovi paradigmi che affrontano sfide specifiche nell'ambito dello sviluppo del software.

La programmazione reattiva si concentra sulla gestione degli eventi e sulla reattività del sistema. Con l'aumento dell'interazione utente e dei dispositivi connessi, è diventato sempre più importante sviluppare sistemi che possano rispondere in modo rapido ed efficiente agli eventi. Linguaggi come RxJava e ReactJS sono ampiamente utilizzati per la programmazione reattiva.

La programmazione parallela si occupa della suddivisione dei compiti in sottocompiti che possono essere eseguiti contemporaneamente su più processori o core di elaborazione. Questo paradigma è diventato cruciale per sfruttare al massimo le prestazioni dei moderni sistemi di elaborazione parallela e distribuita. Framework come OpenMP, CUDA e MPI consentono la programmazione parallela in diversi contesti.

Questa lezione e' stata realizzata grazie al contributo di:

Baobab.school
Risorse per la scuola
https://www.baobab.school

FrancescoBelloni.it
software development
Siti web a Varese
https://www.francescobelloni.it