Cos'è l'Informatica? Definizione, Impatto, Evoluzione

Prof. Marco Bernardo

Università degli Studi di Urbino Carlo Bo
Dipartimento di Scienze Pure e Applicate
Sezione di Informatica e Matematica

Scuola di Scienza, Ingegneria e Filosofia dell'Informazione


1506
UNIVERSITÀ
DEGLI STUDI
DI URBINO
CARLO BO
CORSO DI LAUREA TRIENNALE
INGEGNERIA
INFORMATICA
E SCIENZE
COMPUTAZIONALI
#NOPLACEFORBUGS
CORSO DI LAUREA MAGISTRALE
INFORMATICA
E
INNOVAZIONE
DIGITALE
#MAKEITDIGITAL
AI
IoT
https://informatica.uniurb.it/

Cosa Intendiamo per Informatica?

Cosa Intendiamo per Informatica?

- Informatica: studio dell'elaborazione automatica delle informazioni.
- La gente percepisce solo gli aspetti tecnologici, mentre chi ci lavora deve conoscere anche gli aspetti metodologici e scientifici.

Cosa Intendiamo per Informatica?

- Informatica: studio dell'elaborazione automatica delle informazioni.
- La gente percepisce solo gli aspetti tecnologici, mentre chi ci lavora deve conoscere anche gli aspetti metodologici e scientifici.
- Aspetti tecnologici – *Information & Communication Technology*: computer, sistemi operativi, basi di dati, reti di calcolatori, ...

Cosa Intendiamo per Informatica?

- Informatica: studio dell'elaborazione automatica delle informazioni.
- La gente percepisce solo gli aspetti tecnologici, mentre chi ci lavora deve conoscere anche gli aspetti metodologici e scientifici.
- Aspetti tecnologici – *Information & Communication Technology*: computer, sistemi operativi, basi di dati, reti di calcolatori, ...
- Aspetti metodologici – *Software Architecture & Engineering*: metodologie, linguaggi e ambienti di programmazione, ...

Cosa Intendiamo per Informatica?

- Informatica: studio dell'elaborazione automatica delle informazioni.
- La gente percepisce solo gli aspetti tecnologici, mentre chi ci lavora deve conoscere anche gli aspetti metodologici e scientifici.
- Aspetti tecnologici – *Information & Communication Technology*: computer, sistemi operativi, basi di dati, reti di calcolatori, ...
- Aspetti metodologici – *Software Architecture & Engineering*: metodologie, linguaggi e ambienti di programmazione, ...
- Aspetti scientifici – *Computer Science*: teoria della computazione, algoritmica, teoria degli automi, linguaggi formali, ...

Cosa Intendiamo per Computer?

Cosa Intendiamo per Computer?

- Computer: insieme di componenti elettromeccaniche programmabili per immettere, immagazzinare, elaborare ed emettere informazioni sotto forma di numeri, testi, immagini, audio e video.
- Ci sono tanti insiemi di componenti elettromeccaniche che possono eseguire *molteplici* funzioni (elettrodomestici, ascensori, veicoli, . . .) ma solo i computer ne possono eseguire un numero potenzialmente *illimitato*, cioè solo i computer sono macchine programmabili!

Cosa Intendiamo per Computer?

- Computer: insieme di componenti elettromeccaniche programmabili per immettere, immagazzinare, elaborare ed emettere informazioni sotto forma di numeri, testi, immagini, audio e video.
- Ci sono tanti insiemi di componenti elettromeccaniche che possono eseguire *molteplici* funzioni (elettrodomestici, ascensori, veicoli, . . .) ma solo i computer ne possono eseguire un numero potenzialmente *illimitato*, cioè solo i computer sono macchine programmabili!
- Hardware: insieme delle componenti elettromeccaniche che formano un computer, cioè le risorse computazionali disponibili (*parti fisiche*).
- Software: insieme dei programmi che possono girare su un computer, cioè istruzioni date alle risorse computazionali per eseguire compiti (*parti immateriali* – nome derivante dai telai Jacquard).

Quali Sono le Basi Teoriche?

- Dato un problema computazionale, la teoria della computazione studia l'esistenza di una soluzione algoritmica e, se questa c'è, quante risorse computazionali sono necessarie in termini di:
  - Tempo d'esecuzione.
  - Spazio di memoria.
  - Banda di comunicazione.

Quali Sono le Basi Teoriche?

- Dato un problema computazionale, la teoria della computazione studia l'esistenza di una soluzione algoritmica e, se questa c'è, quante risorse computazionali sono necessarie in termini di:
  - Tempo d'esecuzione.
  - Spazio di memoria.
  - Banda di comunicazione.
- Teoria della calcolabilità: problemi decidibili vs. indecidibili a seconda dell'esistenza o meno di una soluzione algoritmica.
- Teoria della complessità: problemi (decidibili) trattabili vs. intrattabili a seconda della quantità polinomiale o esponenziale di risorse.

Quali Sono le Basi Teoriche?

- Dato un problema computazionale, la teoria della computazione studia l'esistenza di una soluzione algoritmica e, se questa c'è, quante risorse computazionali sono necessarie in termini di:
  - Tempo d'esecuzione.
  - Spazio di memoria.
  - Banda di comunicazione.
- Teoria della calcolabilità: problemi decidibili vs. indecidibili a seconda dell'esistenza o meno di una soluzione algoritmica.
- Teoria della complessità: problemi (decidibili) trattabili vs. intrattabili a seconda della quantità polinomiale o esponenziale di risorse.
- Se un problema computazionale è decidibile, potrebbero esserci più algoritmi diversi che lo risolvono in modo *più o meno efficiente* (quicksort, mergesort, heapsort vs. insertsort, selectsort, bubblesort).

Cosa Intendiamo per Algoritmo?

Cosa Intendiamo per Algoritmo?

- Algoritmo: sequenza finita di passi, rappresentati in modo da essere intelligibili da un esecutore, che risolvono un problema computazionale nella sua generalità (cioè per ogni istanza dei suoi dati di ingresso).
- Il matematico persiano Muhammad ibn Musa al-Khwarizmi (780-850) fu l'autore di uno dei più antichi trattati di algebra (al-jabr) in cui descrisse come risolvere equazioni lineari e quadratiche.

Cosa Intendiamo per Algoritmo?

- Algoritmo: sequenza finita di passi, rappresentati in modo da essere intelligibili da un esecutore, che risolvono un problema computazionale nella sua generalità (cioè per ogni istanza dei suoi dati di ingresso).
- Il matematico persiano Muhammad ibn Musa al-Khwarizmi (780-850) fu l'autore di uno dei più antichi trattati di algebra (al-jabr) in cui descrisse come risolvere equazioni lineari e quadratiche.
- Sebbene la rappresentazione di un algoritmo abbia lunghezza finita, la durata della sua esecuzione può essere illimitata (p.e. cicli).
- L'esecutore non è necessariamente un agente elettromeccanico, può essere un agente biologico o un agente cyberfisico (p.e. ricette).

Cosa Intendiamo per Algoritmo?

- Algoritmo: sequenza finita di passi, rappresentati in modo da essere intelligibili da un esecutore, che risolvono un problema computazionale nella sua generalità (cioè per ogni istanza dei suoi dati di ingresso).
- Il matematico persiano Muhammad ibn Musa al-Khwarizmi (780-850) fu l'autore di uno dei più antichi trattati di algebra (al-jabr) in cui descrisse come risolvere equazioni lineari e quadratiche.
- Sebbene la rappresentazione di un algoritmo abbia lunghezza finita, la durata della sua esecuzione può essere illimitata (p.e. cicli).
- L'esecutore non è necessariamente un agente elettromeccanico, può essere un agente biologico o un agente cyberfisico (p.e. ricette).
- Programma: algoritmo espresso in un dato ling. di programmazione, il cui esecutore sarà un computer (software, applicazione, codice, ...).

Quali Sono gli Impatti?

- Impatto socio-economico dell'informatica a partire dagli anni 1950:
  - Eseguire calcoli complicati in tempi brevi.
  - Elaborare/recuperare/trasmettere grandi quantità di dati in tempi brevi.
  - Trasferire attività ripetitive dalle persone alle macchine.
  - Trasferire attività complesse dalle persone alle macchine.

Quali Sono gli Impatti?

- Impatto socio-economico dell'informatica a partire dagli anni 1950:
  - Eseguire calcoli complicati in tempi brevi.
  - Elaborare/recuperare/trasmettere grandi quantità di dati in tempi brevi.
  - Trasferire attività ripetitive dalle persone alle macchine.
  - Trasferire attività complesse dalle persone alle macchine.
- L'invenzione della stampa nel 1400 iniziò a favorire nel mondo una maggiore diffusione della conoscenza.

Quali Sono gli Impatti?

- Impatto socio-economico dell'informatica a partire dagli anni 1950:
  - Eseguire calcoli complicati in tempi brevi.
  - Elaborare/recuperare/trasmettere grandi quantità di dati in tempi brevi.
  - Trasferire attività ripetitive dalle persone alle macchine.
  - Trasferire attività complesse dalle persone alle macchine.
- L'invenzione della stampa nel 1400 iniziò a favorire nel mondo una maggiore diffusione della conoscenza.
- La rivoluzione industriale nel 1700 ampliò le nostre capacità fisiche attraverso l'introduzione di macchine automatiche.

Quali Sono gli Impatti?

- Impatto socio-economico dell'informatica a partire dagli anni 1950:
  - Eseguire calcoli complicati in tempi brevi.
  - Elaborare/recuperare/trasmettere grandi quantità di dati in tempi brevi.
  - Trasferire attività ripetitive dalle persone alle macchine.
  - Trasferire attività complesse dalle persone alle macchine.
- L'invenzione della stampa nel 1400 iniziò a favorire nel mondo una maggiore diffusione della conoscenza.
- La rivoluzione industriale nel 1700 ampliò le nostre capacità fisiche attraverso l'introduzione di macchine automatiche.
- La trasformazione digitale, basata su elettronica e informatica, sta estendendo in misura crescente le nostre capacità cognitive mediante dispositivi programmabili come computer e smartphone nonché la rapida propagazione di dati digitali via Internet.

- Un altro aspetto socio-economico è dato dal software open source.
- Comunità mondiali di persone che sviluppano e manutengono sw.

- Un altro aspetto socio-economico è dato dal software open source.
- Comunità mondiali di persone che sviluppano e manutengono sw.
- Essere open source (cioè ispezionabile) è un prerequisito tecnico per essere un software libero.
- Free Software Foundation fondata da Richard Stallman (1985):
  - Libertà di eseguire un programma per qualsiasi scopo.
  - Libertà di studiare un programma e di modificarlo.
  - Libertà di distribuire copie di un programma a chi ne abbia bisogno.
  - Libertà di migliorare un programma e di distribuirne pubblicamente i miglioramenti in modo tale che chiunque ne possa beneficiare.

- Un altro aspetto socio-economico è dato dal software open source.
- Comunità mondiali di persone che sviluppano e manutengono sw.
- Essere open source (cioè ispezionabile) è un prerequisito tecnico per essere un software libero.
- Free Software Foundation fondata da Richard Stallman (1985):
  - Libertà di eseguire un programma per qualsiasi scopo.
  - Libertà di studiare un programma e di modificarlo.
  - Libertà di distribuire copie di un programma a chi ne abbia bisogno.
  - Libertà di migliorare un programma e di distribuirne pubblicamente i miglioramenti in modo tale che chiunque ne possa beneficiare.
- Esempi: GNU/Linux, Firefox, WordPress, LibreOffice, Moodle, ...
- Distribuzione di software a pacchetti.

- Il pensiero computazionale è il contributo culturale dell'informatica, inteso come processo mentale finalizzato alla risoluzione di problemi.
- Espressione coniata da Jeannette Wing (2006).

- Il pensiero computazionale è il contributo culturale dell'informatica, inteso come processo mentale finalizzato alla risoluzione di problemi.
- Espressione coniata da Jeannette Wing (2006).
- Combinazione di:
  - Metodi caratteristici dell'informatica:
    - Analisi algoritmica dei problemi.
    - Rappresentazione digitale dei dati.
    - Sviluppo di soluzioni automatizzate.
  - Capacità intellettuali generali:
    - Affrontare la complessità.
    - Confrontare le alternative.

- Il pensiero computazionale è il contributo culturale dell'informatica, inteso come processo mentale finalizzato alla risoluzione di problemi.
- Espressione coniata da Jeannette Wing (2006).
- Combinazione di:
  - Metodi caratteristici dell'informatica:
    - Analisi algoritmica dei problemi.
    - Rappresentazione digitale dei dati.
    - Sviluppo di soluzioni automatizzate.
  - Capacità intellettuali generali:
    - Affrontare la complessità.
    - Confrontare le alternative.
- Variante computazionale del pensiero astratto della matematica: *il software è immateriale!*

Breve Panoramica Storica

- Quando inizia la storia dell'informatica?

Breve Panoramica Storica

- Quando inizia la storia dell'informatica?
- Pionieri tra 1600 e 1800 (tecnologia meccanica):
  - Blaise Pascal (1623–1662):
    - Macchina meccanica capace di fare addizioni e sottrazioni (1642).

Breve Panoramica Storica

- Quando inizia la storia dell'informatica?
- Pionieri tra 1600 e 1800 (tecnologia meccanica):
  - Blaise Pascal (1623–1662):
    - Macchina meccanica capace di fare addizioni e sottrazioni (1642).
  - Gottfried Wilhelm von Leibniz (1646–1716):
    - Macchina meccanica comprensiva di moltiplicazioni e divisioni (1672).
    - Sistema binario, characteristica universalis, calculus ratiocinator.

Breve Panoramica Storica

- Quando inizia la storia dell'informatica?
- Pionieri tra 1600 e 1800 (tecnologia meccanica):
  - Blaise Pascal (1623–1662):
    - Macchina meccanica capace di fare addizioni e sottrazioni (1642).
  - Gottfried Wilhelm von Leibniz (1646–1716):
    - Macchina meccanica comprensiva di moltiplicazioni e divisioni (1672).
    - Sistema binario, characteristica universalis, calculus ratiocinator.
  - Charles Babbage (1791–1871):
    - Difference Engine (1822): tabulazione di funzioni polinomiali e quindi approssimazione di funzioni logaritmiche e trigonometriche.
    - Analytical Engine (1834): archetipo di tutti i moderni computer sia per la sua architettura che per il suo insieme di istruzioni.

Breve Panoramica Storica

- Quando inizia la storia dell'informatica?
- Pionieri tra 1600 e 1800 (tecnologia meccanica):
  - Blaise Pascal (1623–1662):
    - Macchina meccanica capace di fare addizioni e sottrazioni (1642).
  - Gottfried Wilhelm von Leibniz (1646–1716):
    - Macchina meccanica comprensiva di moltiplicazioni e divisioni (1672).
    - Sistema binario, characteristica universalis, calculus ratiocinator.
  - Charles Babbage (1791–1871):
    - Difference Engine (1822): tabulazione di funzioni polinomiali e quindi approssimazione di funzioni logaritmiche e trigonometriche.
    - Analytical Engine (1834): archetipo di tutti i moderni computer sia per la sua architettura che per il suo insieme di istruzioni.
  - Ada Byron Lovelace (1815–1852):
    - Traduzione del e note sul progetto dell'Analytical Engine (1843).
    - Numeri di Bernoulli: *lei fu la prima persona a programmare nella storia!*

- Alcuni pionieri del 1900 (tecnologia elettronica):
  - Norbert Wiener (1894–1964):
    - Cibernetica: studio interdisciplinare di sistemi naturali e artificiali in termini di controllo, retroazione e comunicazione.

- Alcuni pionieri del 1900 (tecnologia elettronica):
  - Norbert Wiener (1894–1964):
    - Cibernetica: studio interdisciplinare di sistemi naturali e artificiali in termini di controllo, retroazione e comunicazione.
  - Claude Shannon (1916–2001):
    - Connessione tra circuiti elettrici e algebra di George Boole (1847).
    - Teoria dell'informazione: studio della codifica e della trasmissione di informazioni digitali, la cui unità elementare venne da lui chiamata bit.

- Alcuni pionieri del 1900 (tecnologia elettronica):
  - Norbert Wiener (1894–1964):
    - Cibernetica: studio interdisciplinare di sistemi naturali e artificiali in termini di controllo, retroazione e comunicazione.
  - Claude Shannon (1916–2001):
    - Connessione tra circuiti elettrici e algebra di George Boole (1847).
    - Teoria dell'informazione: studio della codifica e della trasmissione di informazioni digitali, la cui unità elementare venne da lui chiamata bit.
  - Alan Turing (1912–1954):
    - Macchina di Turing (MT): prima descrizione del concetto di algoritmo.
    - Macchina di Turing universale (MTU): visione del software in termini di schema di computazione non più cablato nell'hardware.
    - Crittografia e intelligenza artificiale.

- Alcuni pionieri del 1900 (tecnologia elettronica):
  - Norbert Wiener (1894–1964):
    - Cibernetica: studio interdisciplinare di sistemi naturali e artificiali in termini di controllo, retroazione e comunicazione.
  - Claude Shannon (1916–2001):
    - Connessione tra circuiti elettrici e algebra di George Boole (1847).
    - Teoria dell'informazione: studio della codifica e della trasmissione di informazioni digitali, la cui unità elementare venne da lui chiamata bit.
  - Alan Turing (1912–1954):
    - Macchina di Turing (MT): prima descrizione del concetto di algoritmo.
    - Macchina di Turing universale (MTU): visione del software in termini di schema di computazione non più cablato nell'hardware.
    - Crittografia e intelligenza artificiale.
  - John Von Neumann (1903–1957):
    - Dai plugboard computer agli stored program computer (MTU).
    - Dall'aritmetica decimale seriale all'aritmetica binaria parallela.
    - Teoria dei giochi e fisica quantistica.

- Pionieri italiani:
  - Adriano Olivetti (1901–1960):
    - Elea 9003 (1957): *primo computer interamente basato su transistor*, progettato alla Olivetti da Mario Tchou (1924–1961).
    - P101 (1964): *primo personal computer della storia*, progettato alla Olivetti da Pier Giorgio Perotto (1930–2002).

- Pionieri italiani:
  - Adriano Olivetti (1901–1960):
    - Elea 9003 (1957): *primo computer interamente basato su transistor*, progettato alla Olivetti da Mario Tchou (1924–1961).
    - P101 (1964): *primo personal computer della storia*, progettato alla Olivetti da Pier Giorgio Perotto (1930–2002).
  - CEP – Calcolatrice Elettronica Pisana (1961):
    - Primo computer scientifico progettato in Italia, all'Università di Pisa, un'iniziativa promossa da Enrico Fermi e Adriano Olivetti.

- Pionieri italiani:
  - Adriano Olivetti (1901–1960):
    - Elea 9003 (1957): *primo computer interamente basato su transistor*, progettato alla Olivetti da Mario Tchou (1924–1961).
    - P101 (1964): *primo personal computer della storia*, progettato alla Olivetti da Pier Giorgio Perotto (1930–2002).
  - CEP – Calcolatrice Elettronica Pisana (1961):
    - Primo computer scientifico progettato in Italia, all'Università di Pisa, un'iniziativa promossa da Enrico Fermi e Adriano Olivetti.
  - Federico Faggin (1941):
    - Progettò il primo microprocessore della storia alla Intel (1971).
    - Sviluppò i primi touchpad e touchscreen (fine anni 1980).

Evoluzione dell'Informatica

- Velocità di calcolo e capacità di memoria crescenti come pure costi e dimensioni calanti dell'hw (ENIAC vs. computer moderni):
  tubi a vuoto $\rightarrow$ transistor $\rightarrow$ circuiti integrati $\rightarrow$ VLSI $\rightarrow$ ...
- Esigenza di infrastrutture/protocolli sw e linguaggi di programmazione per supportare l'evoluzione tecnologica e la produttività dell'utenza.

Evoluzione dell'Informatica

- Velocità di calcolo e capacità di memoria crescenti come pure costi e dimensioni calanti dell'hw (ENIAC vs. computer moderni):
  tubi a vuoto $\rightarrow$ transistor $\rightarrow$ circuiti integrati $\rightarrow$ VLSI $\rightarrow \ldots$
- Esigenza di infrastrutture/protocolli sw e linguaggi di programmazione per supportare l'evoluzione tecnologica e la produttività dell'utenza.
- Sistemi operativi, basi di dati, reti di calcolatori (1969: Internet).
- Crescente complessità dei sistemi di elaborazione:
  sequenziali $\rightarrow$ *concorrenti* $\rightarrow$ *distribuiti* $\rightarrow$ *decentralizzati* $\rightarrow \ldots$

Evoluzione dell'Informatica

- Velocità di calcolo e capacità di memoria crescenti come pure costi e dimensioni calanti dell'hw (ENIAC vs. computer moderni):
  tubi a vuoto $\rightarrow$ transistor $\rightarrow$ circuiti integrati $\rightarrow$ VLSI $\rightarrow \ldots$
- Esigenza di infrastrutture/protocolli sw e linguaggi di programmazione per supportare l'evoluzione tecnologica e la produttività dell'utenza.
- Sistemi operativi, basi di dati, reti di calcolatori (1969: Internet).
- Crescente complessità dei sistemi di elaborazione:
  sequenziali $\rightarrow$ *concorrenti* $\rightarrow$ *distribuiti* $\rightarrow$ *decentralizzati* $\rightarrow \ldots$
- Mobile computing: dispositivi non vincolati a locazioni fisiche (IoT).
- Global computing: astrazione di un singolo computer globale accessibile ovunque in ogni momento (cloud, edge, fog).
- Autonomic computing: caratteristiche di autogestione per adattarsi a cambiamenti imprevisti (sensori, attuatori, politiche, AI).

- Linguaggi di programmazione:
  - *(Imperativi) procedurali*: Fortran, Cobol, Algol, Basic, Pascal, C, ...
  - *(Imperativi) a oggetti*: Simula, Smalltalk, C++, Java, C#, ...
  - *Dichiarativi (funzionali/logici)*: Lisp, Scheme, ML, Haskell, Prolog, ...
  - *Concorrenti*: Modula, Ada, Occam, Erlang, Scala, ...

- Linguaggi di programmazione:
  - *(Imperativi) procedurali*: Fortran, Cobol, Algol, Basic, Pascal, C, ...
  - *(Imperativi) a oggetti*: Simula, Smalltalk, C++, Java, C#, ...
  - *Dichiarativi (funzionali/logici)*: Lisp, Scheme, ML, Haskell, Prolog, ...
  - *Concorrenti*: Modula, Ada, Occam, Erlang, Scala, ...
- Altri linguaggi:
  - *Script*: shell, Perl/Raku, Tcl/Tk, Python, PHP, JavaScript, Ruby, ...
  - *Interrogazione*: SQL, ...
  - *Markup*: HTML, XML, LaTeX, ...
  - *Modellazione*: UML, AADL, ...
  - *Smart contract*: Solidity, Move, Vyper, ...

- Linguaggi di programmazione:
  - *(Imperativi) procedurali*: Fortran, Cobol, Algol, Basic, Pascal, C, ...
  - *(Imperativi) a oggetti*: Simula, Smalltalk, C++, Java, C#, ...
  - *Dichiarativi (funzionali/logici)*: Lisp, Scheme, ML, Haskell, Prolog, ...
  - *Concorrenti*: Modula, Ada, Occam, Erlang, Scala, ...
- Altri linguaggi:
  - *Script*: shell, Perl/Raku, Tcl/Tk, Python, PHP, JavaScript, Ruby, ...
  - *Interrogazione*: SQL, ...
  - *Markup*: HTML, XML, LaTeX, ...
  - *Modellazione*: UML, AADL, ...
  - *Smart contract*: Solidity, Move, Vyper, ...
- Supporto per quantum computing (esecuzione più veloce).
- Supporto per reversible computing (minore consumo energetico).

- Ibridazione con molte altre discipline:
  - Matematica computazionale.
  - Fisica computazionale.
  - Chimica computazionale.
  - Biologia computazionale.
  - Informatica giuridica.
  - Informatica umanistica.

- Ibridazione con molte altre discipline:
  - Matematica computazionale.
  - Fisica computazionale.
  - Chimica computazionale.
  - Biologia computazionale.
  - Informatica giuridica.
  - Informatica umanistica.
- Trasformazione del mondo economico, finanziario e monetario:
  - Finanza decentralizzata e smart contract.
  - Beni mobili e immobili trasformati in token digitali.
  - Criptovalute, stablecoin e central bank digital currency.

- Ibridazione con molte altre discipline:
  - Matematica computazionale.
  - Fisica computazionale.
  - Chimica computazionale.
  - Biologia computazionale.
  - Informatica giuridica.
  - Informatica umanistica.
- Trasformazione del mondo economico, finanziario e monetario:
  - Finanza decentralizzata e smart contract.
  - Beni mobili e immobili trasformati in token digitali.
  - Criptovalute, stablecoin e central bank digital currency.
- Intelligenza artificiale: risposta giusta o risposta più probabile?
  - Reti neurali.
  - Machine learning.
  - Deep learning.
  - Big data.
  - Digital twins.

Perché Teoria e Metodologia? Disastri Informatici

- Le tecnologie dell'informazione e della comunicazione sono *pervasive*.
- Malfunzionamenti, inefficienze, falle di sicurezza, interfacce scadenti: insoddisfazione dell'utenza e maggiori costi nella produzione sw.
- Perdite di vite umane e disastri ambientali se il sw è safety-critical: missioni spaziali, traffico aereo, centrali nucleari, ...
- `https://en.wikipedia.org/wiki/List_of_software_bugs`

Perché Teoria e Metodologia? Disastri Informatici

- Le tecnologie dell'informazione e della comunicazione sono *pervasive*.
- Malfunzionamenti, inefficienze, falle di sicurezza, interfacce scadenti: insoddisfazione dell'utenza e maggiori costi nella produzione sw.
- Perdite di vite umane e disastri ambientali se il sw è safety-critical: missioni spaziali, traffico aereo, centrali nucleari, ...
- `https://en.wikipedia.org/wiki/List_of_software_bugs`
- Metodologie per guidare progettazione, sviluppo e dispiegamento sw.
- Il testing del software non garantisce l'assenza di errori.
- È necessaria la verifica del software:
  - Annotazione del programma (logica di Floyd-Hoare, separation logic) e verifica deduttiva (precondizioni-postcondizioni, invarianti).
  - Modello del programma, formalizzazione delle proprietà (logica modale, logica temporale) e model ckecking (più generazione di controesempi).


1506
UNIVERSITÀ
DEGLI STUDI
DI URBINO
CARLO BO
CORSO DI LAUREA TRIENNALE
INGEGNERIA
INFORMATICA
E SCIENZE
COMPUTAZIONALI
#NOPLACEFORBUGS
CORSO DI LAUREA MAGISTRALE
INFORMATICA
E
INNOVAZIONE
DIGITALE
#MAKEITDIGITAL
AI
IoT
https://informatica.uniurb.it/