Computational Thinking

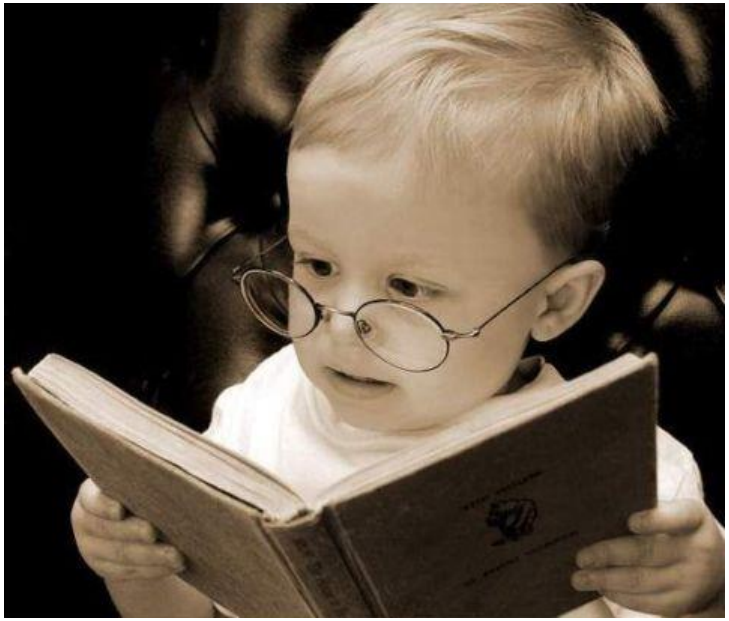
Pensiero computazionale

Luigi Ferrari

IIS Italo Calvino – Genova

Agosto 2015

Abilità di base: leggere



Abilità di base: scrivere

Abilità di base: far di conto

La nuova (?) abilità di base: pensare in modo computazionale

Legami

Il pensiero computazionale prende a prestito concetti e strumenti propri dell'informatica per trovare soluzioni innovative e creative ai problemi di ogni giorno.

Attenzione!

Non significa che l'uomo deve pensare come i computer!

```
.model small
.stack
.data
message   db "Hello world, I'm learning Assembly !!!", "\$"

.code

main   proc
   mov   ax,seg message
   mov   ds,ax

   mov   ah,09
   lea   dx,message
   int   21h

   mov   ax,4c00h
   int   21h
main   endp
end main
```

```
#include <stdio.h>

main()

{
    printf("hello, world\n");
}
```

I computer sono stupidi e noiosi

Solo grazie agli umani possono diventare strumenti utili ed interessanti!

Molto di più…

Il pensiero computazionale è quello che adotta un informatico quando affronta un problema.

Esercitare il pensiero computazionale è molto di più di saper scrivere righe di codice!

La moltiplicazione

Un informatico direbbe che la moltiplicazione è una somma ripetuta e quindi frutto di un'iterazione, uno dei concetti fondamentali alla base del pensiero computazionale.

Inoltre direbbe che, nonostante per la moltiplicazione valga la proprietà commutativa, in termini di efficienza (pratica fortemente ricercata da chi applica il pensiero computazionale) è preferibile sommare tre volte sei piuttosto che sei volte tre.

Una definizione

Comprende:

- il pensiero algoritmico, MA E’ PIU’ DI QUESTO
- il ragionamento in parallelo, MA E’ PIU’ DI QUESTO
- parole come dati, stato, comportamento, interazione, progettazione, MA E’ PIU’ DI QUESTO

Uniti si vince

Il computational thinking sfrutta la potenza e i limiti dei processi di elaborazione, siano essi eseguiti dall'uomo o da una macchina.

I metodi e i modelli computazionali ci permettono di risolvere problemi e progettare sistemi che nessuno di noi sarebbe in grado di affrontare da solo.

Come si pensa in modo computazionale?

Occorre rispondere a qualche domanda:

- Quali sono potenza e limiti dell'intelligenza dell'uomo e del computer?
- Dato un problema, quanto è difficile da affrontare?
- In che modo può essere risolto?
- In che modo la tecnologia può essere applicata al problema?
- Quali strategie computazionali potrebbero essere impiegate?

Idee di base

sviluppare dei modelli

e

simulare dei problemi.

Sviluppare dei modelli, perché?

Simulare dei problemi, perché?

Terminologia di base

Le parole sono il mezzo per comunicare le idee

Terminologia di base

Per introdurre il computational thinking è necessario proporre un'idea dei concetti chiave dell'informatica.

Possiamo partire da alcune definizioni date «alla buona»

Algoritmo (procedura, funzione)

Un algoritmo è un insieme di regole che descrive come risolvere un problema.

Un algoritmo può essere descritto come un programma, pseudo-codice, o una spiegazione passo-passo (volendo anche una ricetta).

Dato (variabile, database, coda…)

Il dato è una informazione che è parte del problema, rivela come questa informazione è organizzata e acceduta.

La distanza fra due città vicine è un dato per il problema del calcolo delle distanze fra le città.

Astrazione (concettualizzazione, modularizzazione)

L’astrazione è l’estrapolazione di proprietà importanti e la generalizzazione delle relazioni.

Tutti noi abbiamo un concetto astratto di automobile: qualsiasi bambino di pochi anni è in grado di distinguere una automobile da un treno, ma anche da un camion o da un autobus.

Il concetto di veicolo è una generalizzazione dei concetti di auto, camion, autobus, treno…

Oggetto

Un’entità che ha certe proprietà e può eseguire certe azioni.

Un essere umano, una automobile, una applicazione di un calcolatore sono tutti oggetti.

Processo

L’esecuzione di un algoritmo.

Un processo potrebbe essere un essere umano, una comunità, un programma in esecuzione che sta eseguendo qualche task.

Sistema

Un sistema è un gruppo di processi o oggetti che interagiscono.

Un sistema potrebbe essere una rete di computer, uno stormo di uccelli, una comunità virtuale, cosi come entità più vaste come sistemi economici e biologici.

Iterazione (loop, ricorsione, ciclo)

L’iterazione riguarda la ripetizione di una o più operazioni fino al raggiungimento di una condizione desiderata.

Nella progettazione e sviluppo di un gioco, un esempio di iterazione si presenta nella costruzione delle versioni via via più efficienti del gioco, dal prototipo iniziale, alla versione beta, alla versione da commercializzare.

Categorie di computational thinking

Categorie di computational thinking

Computazione è un termine abbastanza generale che comprende diversi task, concetti e tecniche.

Analogamente, computational thinking coinvolge un ampio insieme di approcci e abilità.

Per capire il computational thinking distinguendo differenze e somiglianze fra esempi specifici in campi diversi, è utile definire diverse categorie di computational thinking.

Categorie di computazione

I principi di computazione possono essere organizzati in sette categorie, ognuna delle quali enfatizzi una visione univoca sulla computazione.

Distinguiamo quindi *computazione, comunicazione, coordinazione, memoria, automatizzazione, valutazione, e design*.

Computazione

E’ l’esecuzione di un algoritmo, un processo che inizia da uno stato iniziale contenente l’algoritmo e i dati di input, e passa attraverso un numero indeterminato di stati intermedi fino allo stato finale, che determina il raggiungimento dello scopo.

Keywords: *stato e transizione di stato, algoritmo, programma, ricerca esaustiva, backtracking, ricorsione e iterazione, albero di decisione, randomizzazione, complessità, calcolabilità.*

Comunicazione

E’ la trasmissione di informazioni da un processo (o oggetto) ad un altro.

Keywords: *informazione e sua rappresentazione, messaggio, mittente/destinatario, protocollo di comunicazione, compressione di un messaggio, cifratura di un messaggio, correzione d’errore, canale di comunicazione, encoder/decoder, rumore, autenticazione.*

Coordinazione

E’ il controllo (attraverso la comunicazione, per esempio) del tempo di computazione dei processi partecipanti al fine di ottenere lo scopo desiderato.

Keywords: *processi/agenti interagenti, protocolli inter-processo che includono protocolli di comunicazione, sincronizzazione, eventi e gestione degli eventi, dipendenza, concorrenza.*

Memoria

E’ la codifica e l’organizzazione dei dati un modo da rendere efficienti le ricerche e le altre operazioni.

Keywords: *immagazzinamento, gerarchia e organizzazione dei dati, manipolazione dei dati come i.e. inserimenti/accodamento/rimozione, localizzazione dei dati e caching, rappresentazione virtuale, sistemi di naming, riferimenti assoluti e relativi.*

Automatizzazione

E’ il trasferimento delle computazioni nei sistemi fisici che le eseguono.

Keywords: *mapping di un algoritmo in un oggetto di una computazione fisica, meccanizzazione, si applica a processi ripetitivi e offre esecuzioni consistenti, senza errori, veloci e poco costose computazionalmente.*

Valutazione

E’ l’analisi statistica, numerica e sperimentale dei dati.

Keywords: *visualizzazione, analisi dei dati, statistica, data mining, simulazione, esperimento computazionale.*

Design

E’ l’organizzazione (usando astrazione, modularizzazione, aggregazione, scomposizione) di un sistema, di un processo, di un oggetto, ecc.

Keywords: *astrazione, livelli di astrazione, modellazione, modularità, occultazione dell’informazione, classe, architettura, aggregazione, pattern, struttura sottostante.*

Tecniche di Computational Thinking

4 Tecniche

1. Decomposizione
2. Riconoscimento di pattern
3. Generalizzazione
4. Definizione di algoritmi

Decomposizione

Abilità di suddividere un problema in più sottoproblemi di dimensione ridotta, così da poter spiegare chiaramente un procedimento ad un'altra persona o a un computer, o anche soltanto prendere nota per noi stessi.

Decomporre un problema porta di solito al riconoscimento di pattern e alla generalizzazione, e dunque anche alla capacità di progettare un algoritmo.

Esempio: *Quando assaggiamo un piatto nuovo e cerchiamo di identificare i diversi ingredienti in base al loro sapore, lo stiamo decomponendo nei suoi singoli ingredienti.*

Riconoscimento di pattern

Abilità di intravvedere somiglianze o differenze comuni che aiuteranno a fare predizioni o porteranno a scorciatoie.

Il riconoscimento di pattern è solitamente alla base del problem solving e della progettazione di algoritmi.

Esempio: *I bambini identificano pattern di comportamento nella reazione dei propri genitori e degli insegnanti ai propri comportamenti, per determinare cosa è giusto e cosa è sbagliato. Proprio questi pattern condizioneranno i loro comportamenti futuri.*

Generalizzazione di pattern e astrazione

Abilità di filtrare l'informazione non necessaria per la risoluzione di un certo tipo di problema e generalizzare invece l'informazione necessaria.

La generalizzazione di pattern e l'astrazione permettono di rappresentare un'idea o un processo in termini generali, affinché li si possa utilizzare per risolvere altri problemi della stessa natura.

Esempio: *In matematica possiamo generalizzare le formule usando delle variabili invece dei numeri, così da poterle utilizzare per risolvere problemi che coinvolgono valori sempre diversi.*

Progettazione di algoritmi

L’abilità nello sviluppare una soluzione passo-passo per un dato problema.

La progettazione di algoritmi è spesso basata sulla decomposizione e sull’identificazione di pattern che aiutino la risoluzione di un problema.

In informatica, così come in matematica, gli algoritmi sono spesso scritti in modo astratto, cioè utilizzando delle variabili al posto di numeri specifici.

Esempio: *Quando uno chef scrive la ricetta di un piatto sta semplicemente progettando e creando un algoritmo che altri potranno seguire nel cercare di replicare (più o meno fedelmente!) quel piatto.*

Concetti Computazionali

Concetto	Descrizione
Sequenza	Una serie di passaggi in una azione
Loops	Eseguire la stessa sequenza più volte
Parallelismo	Far accadere le cose contemporaneamente
Eventi	Una causa determina un effetto
Condizioni	Prendere decisioni in base alle condizioni
Operatori	Espressioni matematiche e logiche
Dati	Memorizzare, recuperare ed aggiornare valori

Pratiche Computazionali

Pratica	Descrizione
Produrre per iterazioni ed incrementi	Sviluppare una parte per volta, provare e svilupparne ancora
Testare e rimuovere gli errori	Essere sicuri che le cose funzionino, cercare e risolvere gli errori
Riciclare e mescolare	Creare qualcosa partendo da materiale creato da altri
Astrarre e rendere modulare	Creare qualcosa di grande mettendo insieme una collezione di piccole parti

Attitudini del pensiero computazionale

- Esprimere se stessi: una persona dotata di pensiero computazionale vede nella tecnologia uno strumento per esprimere se stessi, la propria creatività e dire qualcosa di sé agli altri.
- Essere connessi: saper comunicare e lavorare con gli altri per raggiungere un obiettivo o una soluzione condivisa.
- Porre domande: saper sviluppare una mente vigile grazie alla quale è sempre viva la domanda di come un oggetto incontrato nel mondo reale possa funzionare.

Competenze da acquisire

In tre ambiti:

- Problem solving
- Computer programming
- Computational Thinking

Competenza 1: Problem solving

- è presente **in ogni disciplina**!

- è una attività mentale di ampio spettro; comprende:

Razionalizzazione di esigenze e **Formalizzazione dei problemi**

- è la più complessa fra tutte le funzioni mentali: è un processo cognitivo di ordine superiore che richiede la **capacità di coordinare e utilizzare** diverse abilità.

- viene attivato quando un organismo o un sistema intelligente artificiale ha il problema di **trasformare una esigenza in una soluzione**.

Competenza 2: Computer Programming

Ovvero: **l'aspetto linguistico dell'Informatica**

Conoscenza di più linguaggi (naturali e/o artificiali) = vantaggio per

- comunicare
- comprendere

Linguaggio di programmazione = strumento per sviluppare le abilità e le competenze di **astrazione**.

Il computer può essere istruito e può apprendere dall'esperienza.

Competenza 3: Computational Thinking

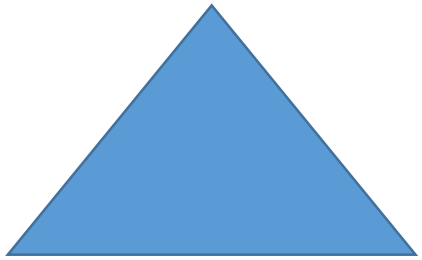
Si fonda sulla potenza e sui limiti dei procedimenti automatizzabili (eseguiti da uomini o da computer).

La conoscenza di metodi e modelli computazionali (e la possibilità di usarli in modo effettivo) consente di affrontare problemi che gli uomini da soli non potrebbero neppure prendere in considerazione.

Con questo termine si indica un insieme di attitudini, abilità e competenze (applicabili universalmente) che ognuno (non solo gli informatici) dovrebbe essere interessato ad apprendere e saper usare.

Il crogiuolo delle competenze

Computational Thinking



Problem Solving ***Computer Programming***

Riepilogo

- Il pensiero computazione è la nuova abilità di base
- Computazione, comunicazione, coordinazione, memoria, automazione, valutazione, design sono le categorie
- Decomposizione, riconoscimento di pattern, generalizzazione, definizione di algoritmi sono le tecniche
- Problem solving, computer programming, computation thinking sono le competenze richieste a chi opera nel settore informatico (e non solo!)

Grazie per l'attenzione.