Corso Formazione
Coding e Pensiero Computazionale
CRIAD @ Casa Bufalini

[modulo-01]

INTRODUZIONE AL CODING E PENSIERO COMPUTAZIONALE IN OTTICA COSTRUZIONISTA

OBIETTIVI DEL MODULO

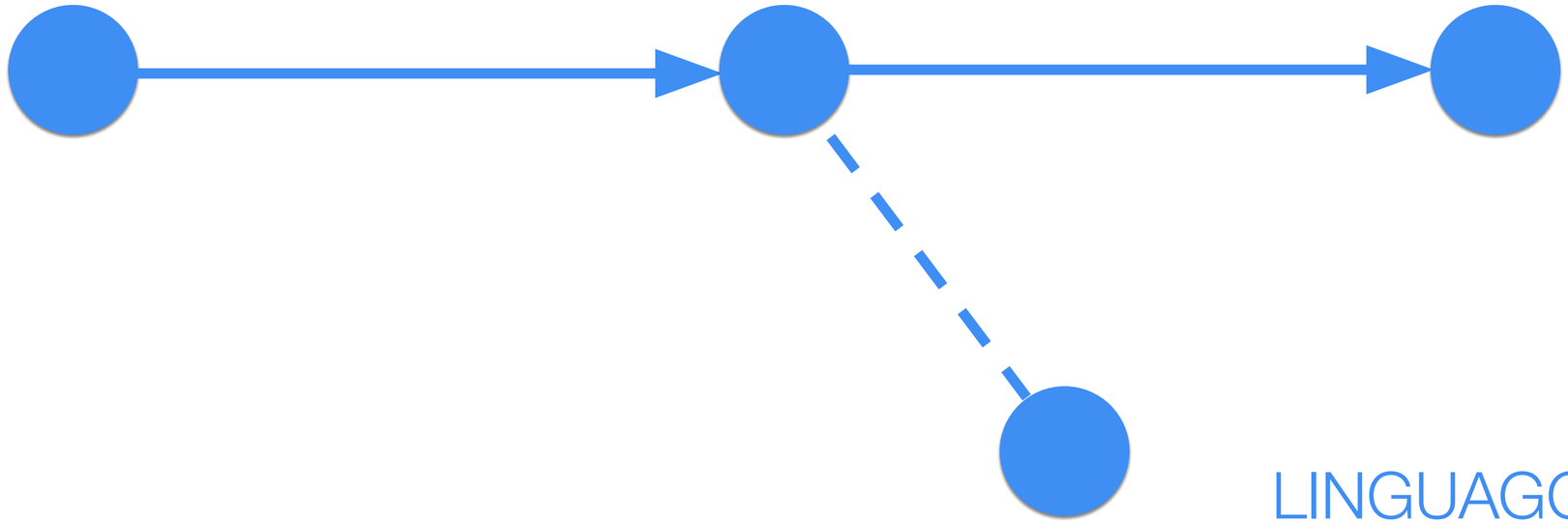
- Fornire un quadro di riferimento in relazione a come inquadrare il ruolo del computer e dell'informatica nella didattica, adottando allo scopo un approccio costruzionista.
  - Introduzione di Snap! come linguaggio / l’ambiente / tecnologia a supporto del corso
- Introdurre concetti principali relativamente al pensiero computazionale e coding in ottica costruzionista e i principi generali con cui inquadrare la progettazione e lo sviluppo di micromondi, considerando Snap! come piattaforma di riferimento.

PUNTI PRINCIPALI

RUOLO del
COMPUTER
E INFORMATICA
NELLA DIDATTICA

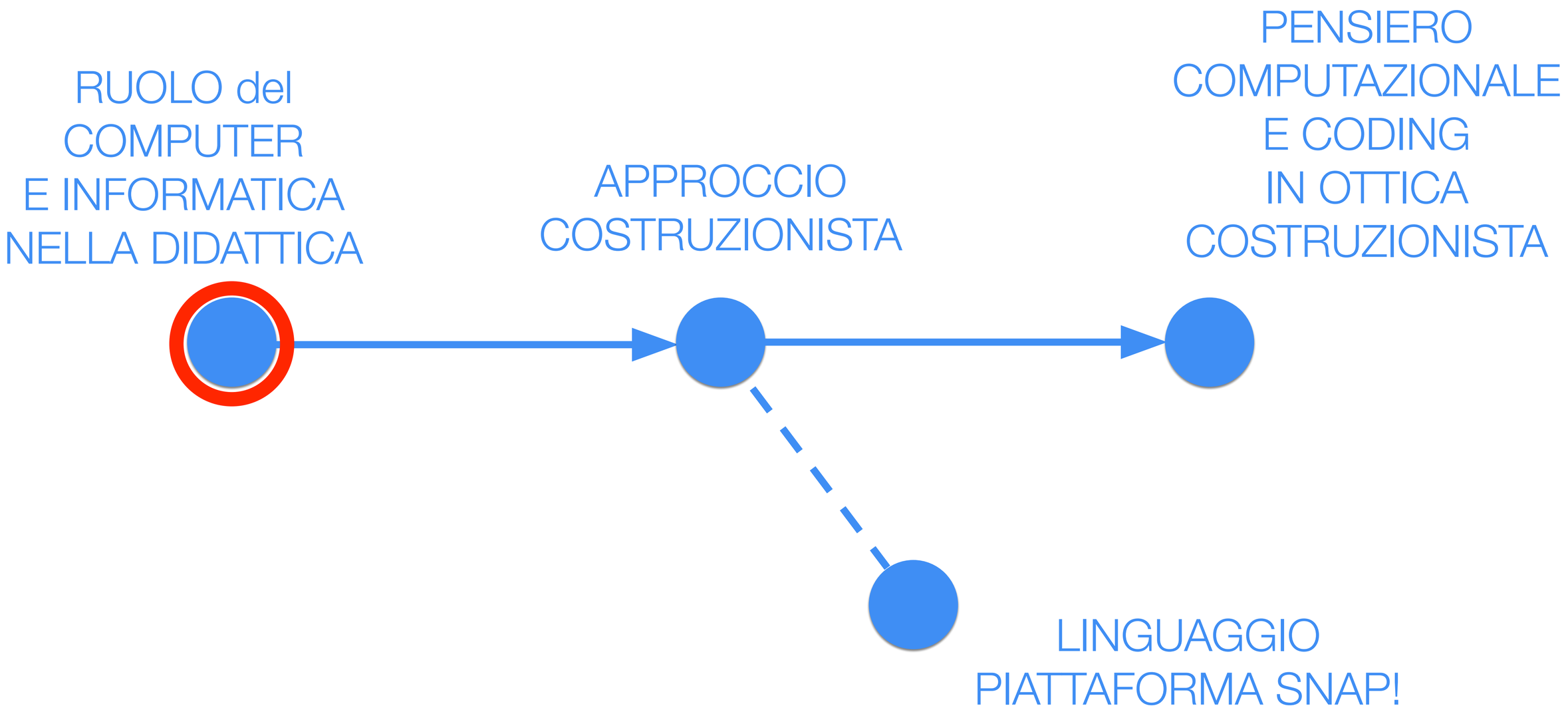
APPROCCIO
COSTRUZIONISTA

PENSIERO
COMPUTAZIONALE
E CODING
IN OTTICA
COSTRUZIONISTA



LINGUAGGIO
PIATTAFORMA SNAP!

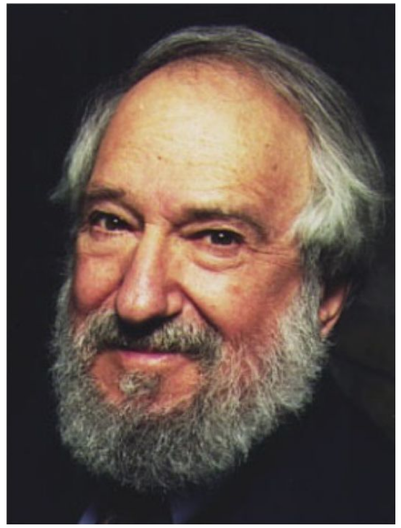
PUNTI PRINCIPALI



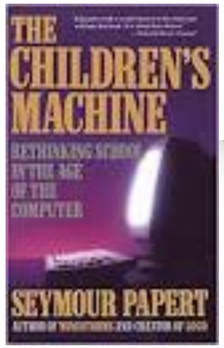
RUOLO COMPUTER E INFORMATICA NELLA DIDATTICA

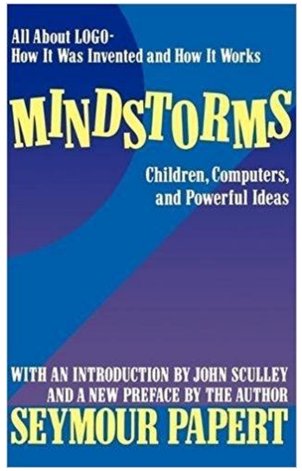
- Domanda di partenza (Papert)
  - In che modo aiutiamo i giovani ad apprendere
  - Studio dell'apporto che la tecnologia informatica può dare ai processi di *insegnamento* e *apprendimento*
    - questioni strettamente correlate ma distinte

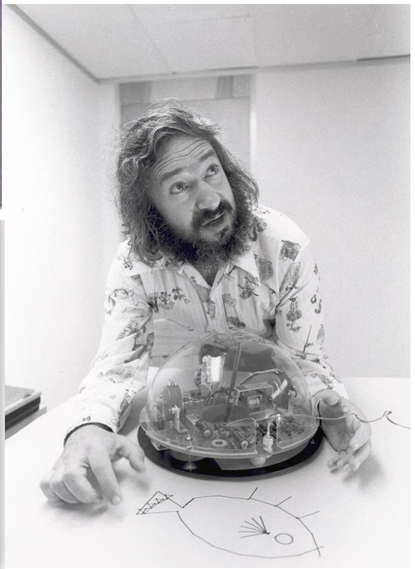
SEYMOUR PAPERT (1928-2016)



- Matematico, filosofo, pedagogista e informatico
- Ha lavorato prima in Europa con Piaget, poi è divenuto professore al MIT (Massachusetts Institute of Technology)
- Padre del costruzionismo
- Inventore del linguaggio LOGO







MIT LIFELONG KINDERGARTEN GROUP

- **Mitch Resnick**
  - Technologies for Lifelong Kindergarten (1996)
  - Programmable Bricks - Toys to think with (1996)
  - Linguaggio/piattaforma Scratch (2009)

TECNOLOGIE DIDATTICHE

- Tecnologie didattiche:
  - insieme di **artefatti** (macchine e strumenti) utilizzati per realizzare processi di insegnamento-apprendimento;
  - insieme di **metodi** sistematici basati sull'applicazione di conoscenze scientifiche, impiegati nella progettazione e nella gestione di processi di insegnamento-apprendimento.

MODALITA' UTILIZZO COMPUTER

- Computer **tutor**
  - quando è il computer e quindi chi l'ha programmato a guidare il processo di insegnamento, attraverso la presentazione di contenuti, la proposta di esercitazioni, seguendo metodologie **CAI** (Computer Aided Instruction) e **CBT** (Computer Based Training). Taglio addestrativo. Paradigma *dell'istruzione*.
- Computer **tool**
  - quando il computer serve principalmente ad emulare la funzionalità di un altro strumento (es. editing con word processor, foglio elettronico, …). Paradigma *dell'espressione*.
- Computer **tutee**
  - quando il computer è un sostegno nel trasformare problemi astratti e di difficile comprensione in problemi *concreti* e *trasparenti*. Paradigma *della costruzione*.

RUOLO DEL COMPUTER

- La terza tipologia è quella che ci interessa, inquadrando il computer e tecnologie informatiche come **ambiente di apprendimento**, *a disposizione sia di chi deve insegnare sia di chi deve apprendere* (Papert, Capponi)

- Non è (solo) una macchina di informazioni o per gestire informazioni
  - più utile di un semplice strumento per domande e risposte, per la memorizzazione di dati e per il calcolo
  - è una macchina per ***eseguire progetti,*** *per costruire*

RUOLO DEL COMPUTER COME AMBIENTE DI APPRENDIMENTO

- Vera essenza: potere di ***simulare*** (Capponi, Parisi)
  - rendere *il formale* concreto e personale (Papert)
- Il computer spinge lo studente a costruire, a fare ipotesi, a esplorare, sperimentare, apprendere in maniera responsabile (nel senso di un personale processo di controllo sul progetto che egli sta costruendo) ed autonoma
- Rende possibile compiere azioni didattiche precluse agli strumenti tradizionali
  - es: esplorare strutture atomo, simulare effetti clima,…

“COMPUTER SUL BANCO”

- Differenza fra uso del computer in laboratorio e un suo utilizzo in classe
  - l'ora settimanale in laboratorio vede l'informatica come una disciplina a sé stante e quindi senza una particolare influenza sul resto delle attività
  - mentre l'avere un computer sul banco offre all'allievo la possibilità di utilizzarlo ogni volta che lo ritenga necessario, indipendentemente dalla materia seguita.

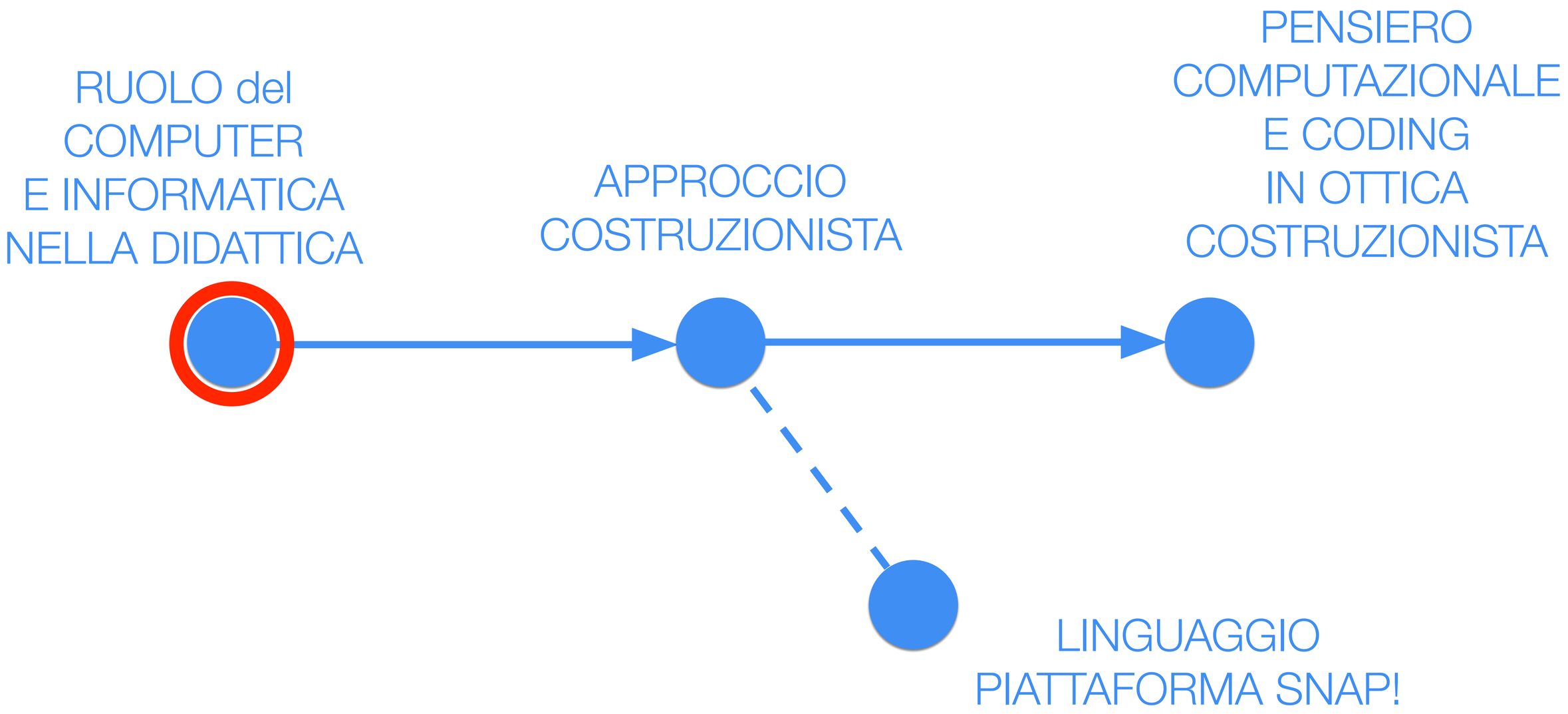
AMBIENTE DI APPRENDIMENTO INTERATTIVO

- **Interattività** è il grado in cui il fruitore può entrare in rapporto con lo strumento didattico ed essere *attivo* nel processo di apprendimento
  - interattività sia *comportamentale*
    - possibilità che lo studente ha di poter intervenire operativamente sullo strumento di cui è fruitore
  - interattività sia *mentale*
    - la capacità dello strumento informatico di favorire riflessioni personali, riorganizzazioni mentali, etc

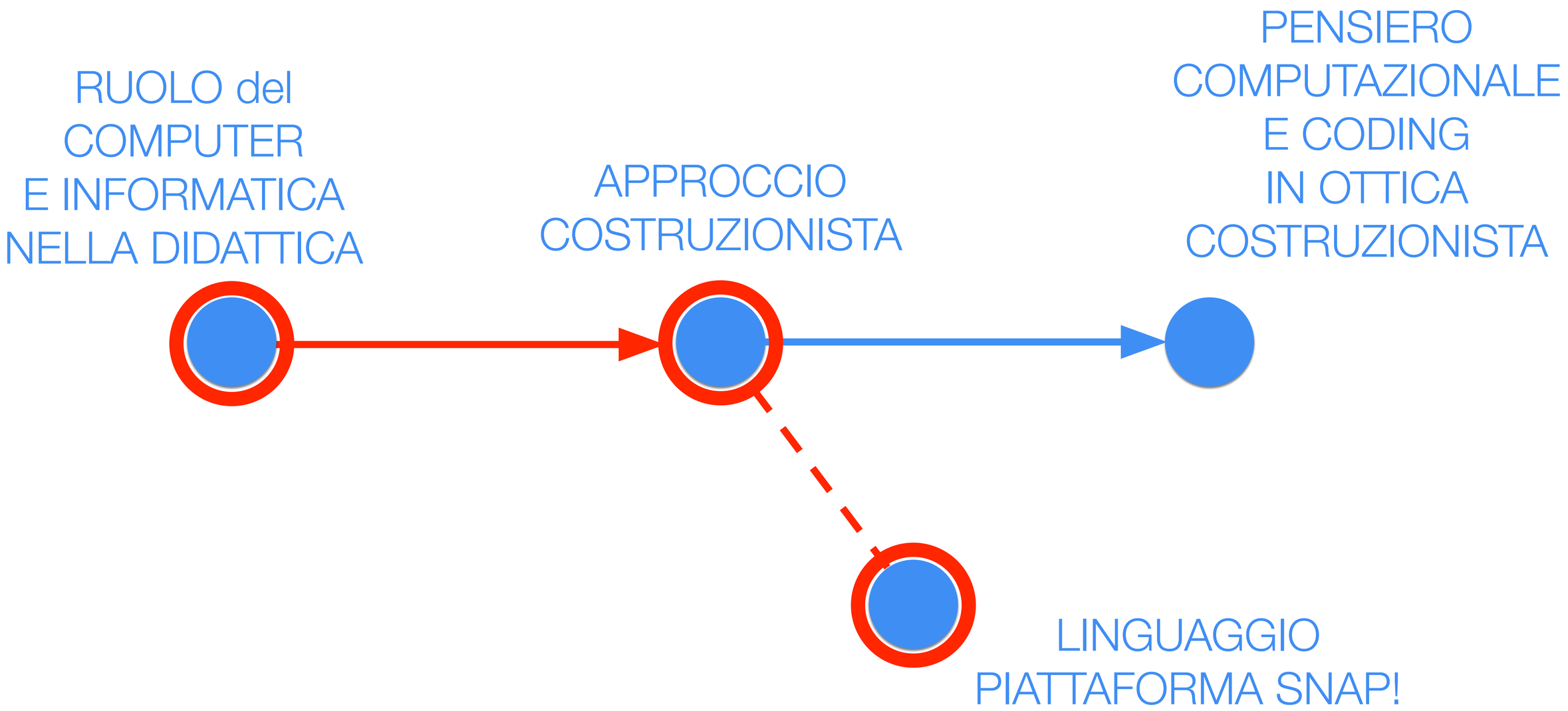
DIDATTICA, “MATETICA” E INFORMATICA PEDAGOGICA

- Visione della **didattica** come ambito conoscitivo che si occupa criticamente *dell'allestimento, consolidamento e valutazione di ambienti di apprendimento,* cioè di specifici contesti risultanti da opportune integrazioni di artefatti culturali e tecnologici (Calvani)
  - concezione student-centered del processo didattico
  - apprendimento come ***costruzione attiva***, da parte del soggetto, della propria conoscenza
- Visione dell'informatica "pedagogica" e "matetica" (Papert)
  - lato insegnante: strumento per costruire ambienti di apprendimento

PUNTI PRINCIPALI



PUNTI PRINCIPALI



APPROCCIO COSTRUZIONISTA

- Parte da una visione "costruttivista" e **costruttivismo** (Piaget)
  - le conoscenze non possono essere semplicemente "trasmesse" o "convogliate già pronte" a un'altra persona. E' sempre un processo di costruzione soggettiva, attiva.

- **Costruzionismo** assegna una particolare *importanza al ruolo delle costruzioni reali* a supporto di quelle presenti nella mente (Papert)
  - l'ambiente di apprendimento deve permettere di supportare una costruzione concreta a partire da quella mentale
  - connotazione di *set da costruzioni*
    - da intendersi in senso letterale, come il set dei Lego

LINGUAGGI DI PROGRAMMAZIONE E COMPUTER NELL'APPROCCIO COSTRUZIONISTA

- I linguaggi di programmazione e i relativi ambienti => "set" da cui possono creare i programml
  - continuità fra progettazione e attualizzazione di un progetto

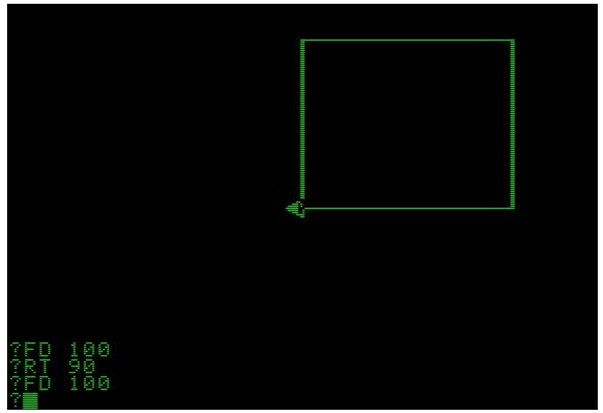
- Computer => ambiente che consente ad ogni singolo studente di esplorare liberamente *nuovi modelli di apprendimento* e di ***costruire attivamente*** le proprie conoscenze in maniera responsabile e controllata
  - apprendimento attivo => è l'allievo a programmare il computer, non viceversa (come prevede l'approccio CAI)
  - uso del computer come ambiente per costruire il proprio sapere => ruolo di **responsabilità** e **creatività** all'allievo

LINGUAGGIO LOGO

- Linguaggio **Logo** come linguaggio a supporto del costruzionismo (Wally Feurzeig, Papert - ~1980)
  - programma = elenco di istruzioni con le quali far muovere una tartaruga su un piano, con la possibilità di eseguire azioni come tracciare linee, emettere suoni, stampare messaggi..

- Strumento che consente ai bambini (scuole elementari, medie) di utilizzare il computer per ottenere *rapidamente risultati concreti*, ma utilizzando principi matematici e logici rigorosi
  - disegni, musica, poesie…

- L'allievo avverte di avere *il controllo del computer*



DA LOGO A SCRATCH E SNAP!

- Piattaforma Scratch (https://scratch.mit.edu/)
  - eredita filosofia e principi del logo, in chiave moderna
  - basata su programmazione visuale
  - piattaforma più utilizzata per coding nelle scuole

- Piattaforma Snap! (https://snap.berkeley.edu/)
  - estensione di Scratch, più configurabile e flessibile

- Inquadrabili come
  - linguaggi/piattaforme di programmazione (coding)
  - linguaggi/piattaforme per costruire ed eseguire *micromondi*

IL CONCETTO DI MICROMONDO

- La costruzione di un programma o un applicazione su queste piattaforme avviene a partire da un ambiente di partenza che può essere visto come “**micromondo**”
  - *un piccolo universo, un “microcosmo”, una realtà delimitata, un luogo confinati e sicuro da esplorare in cui è possibile creare applicazioni sottoforma di percorsi di apprendimento, animazioni, simulazioni geometriche, storytelling, etc.*

<ESEMPI IN SNAP!>

MICROMONDO COME AMBIENTE DI APPRENDIMENTO

c

- Micromondo come ambiente di apprendimento da allestire da parte dell'insegnante
  - ambiente aperto in cui viene incoraggiato *un apprendimento per scoperta*, non direttivo, sulla base di soluzioni di natura euristica
    - luogo di esplorazione di alternative, verifiche di ipotesi e intuizioni, sperimentazione di azioni e strategie
    - "self-image of learner-as-researcher, in genuine joint research activity"
- **Un ambiente ove far praticare esperienze logico-intuitive in contatto sia le idee dell'informatica, sia con quelle della matematica, della linguistica, delle discipline in generale**

MICROMONDI E DISCIPLINE

- Micromondi e discipline, materie
  - micromondi opportunamente progettati a supporto dell'apprendimento di una specifica disciplina/materia…
  - ma anche della loro *integrazione*, come *luogo di creazione e ragionamento multi- e inter- disciplinare*

- L'allievo programma il computer in un linguaggio che favorisce la riflessione sul pensiero e stabilisce un'intima relazione con le idee fondamentali della scienza, matematica e arte nella costruzione di modelli intellettuali (Reggini)
  - ausilio per *abbattere le fobie verso le materie* (matofobia) e tossine culturali varie ("è portato per"), creando opportune condizioni/ambienti di apprendimento
  - ausilio per abbattere il muro fra cultura umanistica e scientifica

APPROCCIO METODOLOGICO PROMOSSO DA LOGO (E SNAP!)

- Obiettivi di base [Reggini]
  - esercitare un processo di **analisi**
    - nella suddivisione dei problemi in piccole parti, nella rappresentazione
  - esercitare la **sintesi**
    - nel costruire procedure di alto livello di complessità, combinando sotto-procedure più elementari

- Ciò consente di:
  - allenare la mente agli aspetti logici del pensiero
  - sviluppare e perfezionare la capacità di risoluzione dei problemi
  - capire la natura dei problemi, compresi quelli irrisolti, la loro formulazione e l'approccio a nuovi tipi di problemi.

APPROCCIO METODOLOGICO PROMOSSO DA LOGO (E SNAP!)

- Principio di ***continuità***'
  - primo passo per comprendere un concetto è la possibilità di integrarlo con conoscenze precedenti
- Principio di ***potenza***
  - ambiente di apprendimento deve consentire a chi apprende di concepire progetti carichi di significato, che non avrebbe mai pensato prima
- Principio di ***risonanza culturale***
  - ciò che viene appreso deve avere senso all'interno dell'ambiente sociale nel quale ci troviamo

STILE DI APPRENDIMENTO “BRICOLAGE”

- Due stili di apprendimento che riguardano l'approccio delle persone ai problemi in generale:
  - *planning* e planners - coloro che hanno l'abitudine di allestire un piano di lavoro per qualunque cosa
  - ***bricolage*** e *bricoleur* o tinkerer - coloro che invece procedono per tentativi ed errori
    - costituito da un avvicinarsi gradualmente alla meta*, un procedere per gradi* governare quella situazione che Papert chiama **indeterminatezza gestita**
- Anche lo stile bricolage è guidato da una intenzione, da una meta sia pur vaga
  - Intenzione e interpretazione si alternano nel produrre significati, schemi, usi.
  - Intenzione a priori e interpretazione a posteriori portano all'emergenza continua di strutture e significati

STILE DI APPRENDIMENTO “BRICOLAGE”

- Lo stile bricolage è fondamentale per irrobustire la capacità di muoversi in condizioni di **incertezza**, di *parziale* comprensione del problema o della soluzione, sia per adattare l'esistente a nuove circostanze
  - importanza di saper amministrare capacità, adattare le abilità alle situazioni contingenti, scegliere strategie trasversali - “*learning by making”*

- Il linguaggio LEGO (e SCRATCH/SNAP! ancora di più) spinge all'uso di un metodo di lavoro molto simile al bricolage, con la preziosa caratteristica aggiuntiva di non fare distinzione fra istruzioni (blocchi) messe a disposizione dal linguaggio stesso e istruzioni personali create dall'utente

- Per chi è solito pianificare, ogni *errore* è un passo nella direzione sbagliata, mentre coloro che usano il bricolage *navigano attraverso correzioni continue - l'interazione è fondamentale, non è un monologo del soggetto ma una conversazione soggetto-oggetto.*

COSTRUZIONE INCREMENTALE E RUOLO DELL'ERRORE

- Impatto sul metodo con cui affrontare un problema / apprendimento
  - molti allievi sono bloccati nell'atto di imparare perché hanno un modello di apprendimento per cui "o si è capito tutto" o "non si è capito”...
  - programmare in LOGO e Scratch/Snap! assomiglia molto all'attività del bricoleur, in cui si promuove un processo di **costruzione incrementale** della soluzione

- In questo: ruolo significativo e positivo *dell’errore*
  - **l’errore** *("bug")* diviene di punto di attenzione per riflettere su come proseguire e quale problema specifico risolvere per proseguire, stimolando un processo di eliminazione dell'errore (**debugging**).

<ESEMPIO IN SNAP!>

SUL RUOLO DELL'ERRORE E DEL "DEBUGGING"

- La costruzione di teorie "errate" è in realtà la costruzione di ***teorie di transizione*** che devono essere "corroborate" (Popper)
  - sono un mezzo per allenare le facoltà cognitive
  - per sviluppare e far emergere le abilità indispensabili alla costruzione di teorie più ortodosse
  - lo studente a fronte di un errore viene messo nella condizione di *controllarlo* e di stabilire con lui un rapporto, in un certo senso, amichevole (Montessori)

- In LOGO / Snap! l'errore e il processo della sua rimozione (debugging) è parte costitutiva del processo di apprendimento e di costruzione della conoscenza
  - l'allievo che programma è incoraggiato a studiare *il bug*, non a cancellarlo in fretta dalla sua memoria (come spesso avviene nelle materie..)
  - l'eliminazione degli errori consente poi allo studente di *constatare i propri progressi*

RIMOZIONE ERRORI E ORGANIZZAZIONE MODULARE

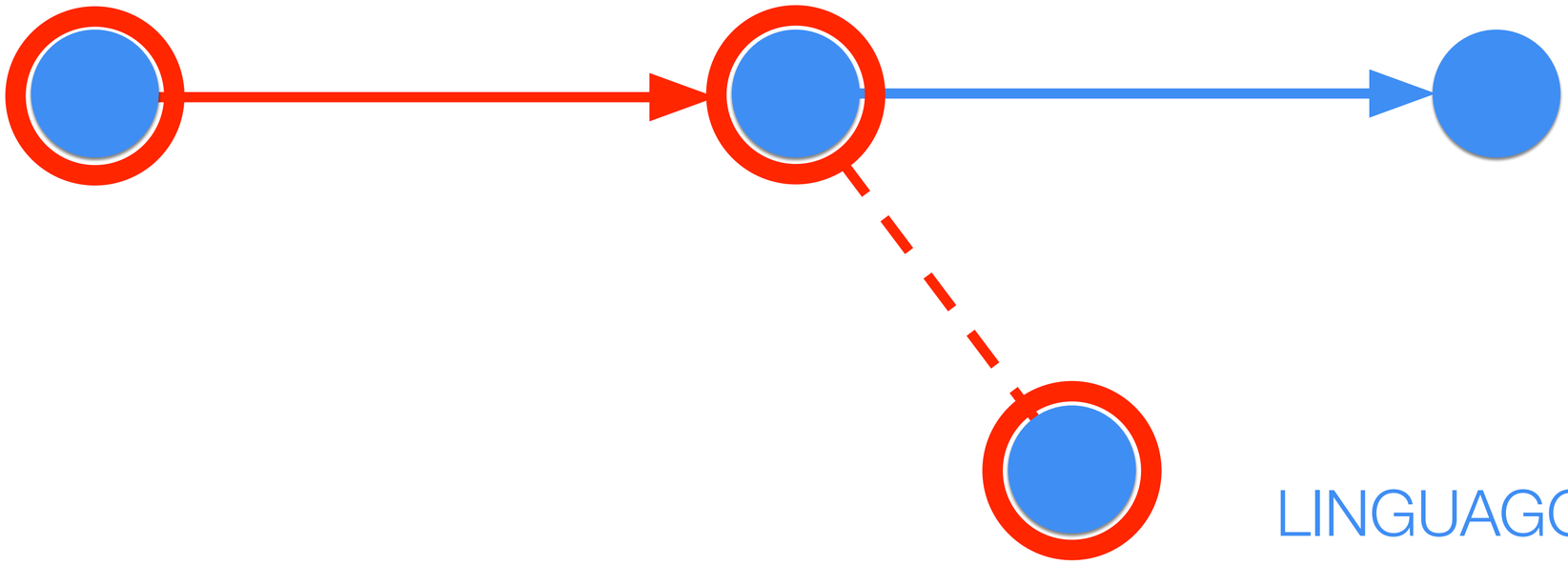
- La ricerca e individuazione degli errori è più semplice se un programma è organizzato in modo modulare e strutturato, ad esempio organizzato in procedure e sotto-procedure
  - *programmazione strutturata*
- In realtà suddividere un problema (e la sua risoluzione) in parti semplici ha una valenza più generale, a livello di problem solving e capacità di progettazione
  - gli studenti spesso hanno difficoltà nella risoluzione di un problema perché vogliono affrontarlo tutto in una volta, mentre - nella maggior parte dei casi - *il processo di soluzione è più facile se si considerano che alcune parti del problema possono essere risolte separatamente, per poi riunirle in una prospettiva generale* (Papert)

PUNTI PRINCIPALI

RUOLO del
COMPUTER
E INFORMATICA
NELLA DIDATTICA

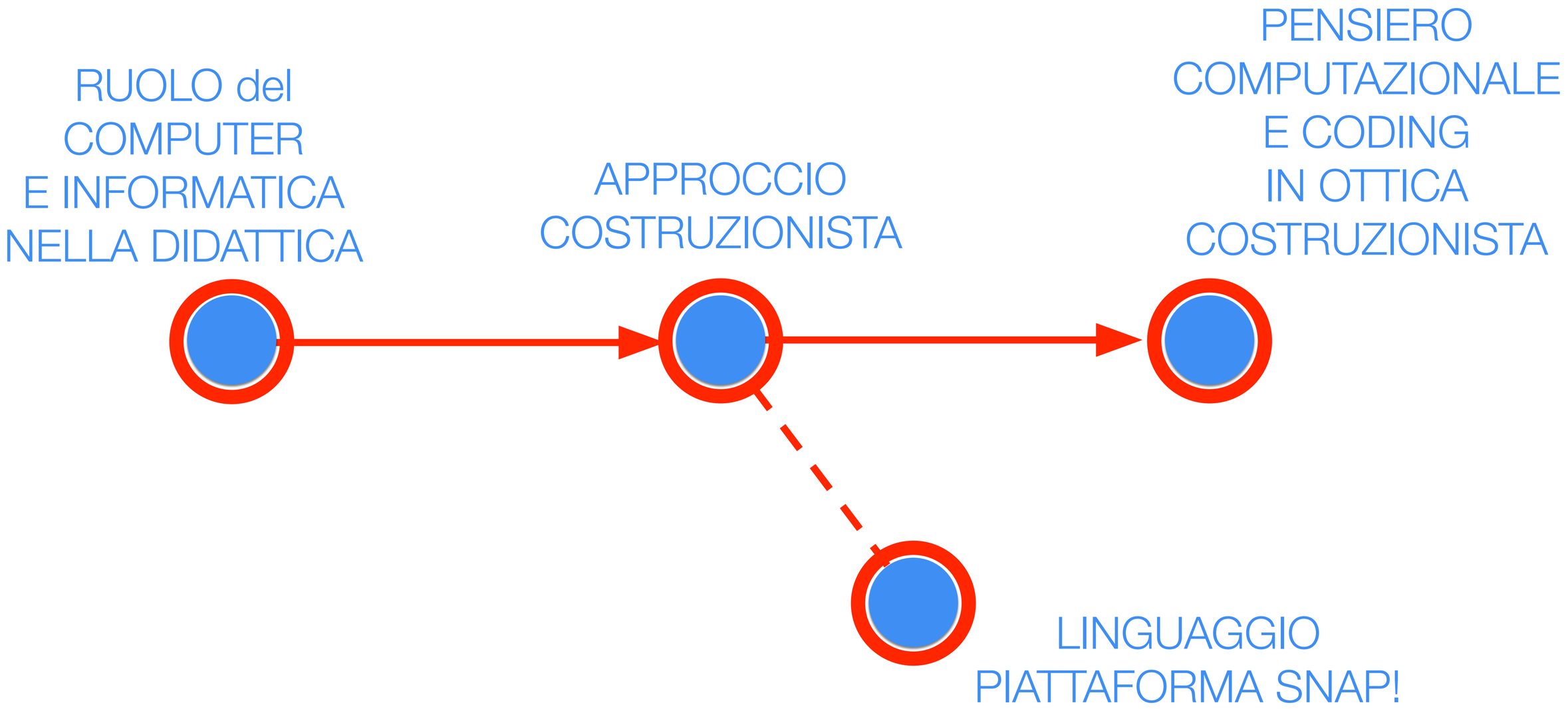
APPROCCIO
COSTRUZIONISTA

PENSIERO
COMPUTAZIONALE
E CODING
IN OTTICA
COSTRUZIONISTA



LINGUAGGIO
PIATTAFORMA SNAP!

PUNTI PRINCIPALI



“PENSIERO COMPUTAZIONALE”

- Definizioni in letteratura
  - Insieme di mental skills e pratiche per (Denning):
    - progettare computazioni che possano essere eseguite da computer che eseguono il compito per noi
    - spiegare e interpretare il mondo come un insieme complesso di processi relativi alla rappresentazione, elaborazione e trasmissione dell’informazione
  - Modo di pensare volto a risolvere problemi, progettare sistemi, comprendere il comportamento umano basandosi su concetti fondamentali dell’Informatica (Wing)
  - Processo mentale di formulazione di problemi e delle loro soluzioni in una forma che sia effettivamente eseguibile da un esecutore che processa l’informazione (Nardelli)

PENSIERO COMPUTAZIONALE

- Analizzare i dati, scomporre il problema in problemi più semplici
- Astrarre liberandosi di dettagli inutili, generalizzare per risolvere classi di problemi
- Scegliere modo migliore per rappresentare i dati, immaginare algoritmi efficienti secondo metriche diverse
- Scrivere il codice in modo che sia sintetico, leggibile, modulare, manutenibile
- Testare e raffinare il codice scritto

PENSIERO COMPUTAZIONALE

CONCETTI

Sequenza,
Costrutti iterativi,
Costrutti condizionali,
Variabili,
Operatori,
Espressioni
Eventi,
…

METODI

Decomposizione
Astrazione
Generalizzazione

PRATICHE

Essere incrementali e iterativi
Testing e debugging
Riuso e mixing
Attenzione all’efficienza
calcolabilità e complessità
…

“SOFT SKILLS”

Esprimere se stessi,
Connettersi,
Farsi domande,
Saper gestire la complessità e i problemi difficili.
Tollerare l'ambiguità e i problemi aperti

CODING, PROGRAMMI, ALGORITMI

- **coding** = Ideazione e scrittura di un programma
- **programma** (nel caso più semplice) = sequenza di istruzioni per raggiungere un certo obiettivo, svolgere un certo compito
- I programmi implementano *algoritmi*
  - **algoritmo** = insieme ordinato di passi o strategia per risolvere operazionalmente un problema o per costruire un sistema

CODING, PROGRAMMI, ALGORITMI

- I programmi sono basati su un **linguaggio di programmazione**
  - usato per descrivere, implementare gli algoritmi
- I programmi sono *eseguiti* da un **ambiente di esecuzione**
  - "macchina" o esecutore

LINGUAGGI DI PROGRAMMAZIONE

- Sono linguaggi *formali*
  - caratterizzati da regole esatte rigorose che permettono di scrivere il programma in maniera non ambigua
- Due macro-aspetti
  - **sintassi** - regole di come le istruzioni devono essere scritte, vocabolario
  - **semantica** - definisce cosa fa il programma, quando mandato in esecuzione da un certo esecutore (es: computer)

LINGUAGGI VISUALI E A BLOCCHI

- Linguaggi **visuali** vs **testuali**
  - equivalenti dal punto di vista espressivo
- Linguaggi visuali **a blocchi** (es: Scratch, Snap!, Blocky)
  - ispirazione dalle costruzioni (Lego)
  - istruzioni come blocchi componibili
    - “affordance” - incastri, colore
  - abbattono alcuni elementi di complessità, astraendo

LINGUAGGI E MICROMONDI

- Ogni linguaggio definisce un ***livello di astrazione***
  - insieme dei concetti di prima classe usati per esprimersi, rappresentare, costruire
- Possiamo definirci il nostro linguaggio, al giusto livello di astrazione per un certo tipo di scopo, di dominio
  - in letteratura informatica si parla di *Domain Specific Language*
- Collegamento con i micromondi
  - *ogni micromondo implicitamente o esplicitamente definisce un linguaggio*

PROSSIMI PASSI

- Progettazione e programmazione di micromondi e programmi
  - comprensione e applicazione dei principali concetti, metodi e pratiche del pensiero computazionale e coding
  - costruzione concreta di micromondi utili in discipline diverse
- Snap! come linguaggio/piattaforma di riferimento

RIFERIMENTI BIBLIOGRAFICI SELEZIONATI

- Massimo Capponi. *Un giocattolo per la mente. L'informatica cognitiva di Seymour Papert.* Morlacchi editore, 2008

- Seymour Papert. *Mindstorms*, Basic Books, New York, 1980 - tr. it. *Mindstorms. Bambini, computers e creatività,* 1984

- Seymour Papert. *The Children's Machine. Rethinking School in the Age of the Computer*, Basic Books, New York (1993) - tr. it: I bambini e il computer, Rizzoli, Milano, 1994

- Reggini (1984). Logo: ali per la mente, Mondadori