Lorenzo Falchi

Jacopo Terenzi

Unità didattica: la ricorsione

A chi è rivolta questa attività?

Questa attività è rivolta a tutti gli studenti della scuola secondaria di secondo grado che hanno già acquisito conoscenze di base di informatica, in particolare quelle relative al pensiero computazionale. L'attività è particolarmente adatta agli studenti che frequentano un indirizzo di studi scientifico-tecnologico, in quanto la ricorsione è un concetto fondamentale in molti ambiti dell'informatica e della matematica.

Collocazione dell'attività

È principalmente inquadrabile nella disciplina di informatica, ma può essere integrata con altre discipline, come la matematica o la logica. In particolare, la matematica può essere utilizzata per fornire una base formale per il concetto di ricorsione. La logica può essere utilizzata per analizzare i principi alla base della ricorsione. Ad esempio, potrebbe essere utilizzata per introdurre il concetto di ricorsione attraverso la matematica, utilizzando esempi come la somma di una serie numerica o il calcolo del fattoriale di un numero. Successivamente, potrebbe essere utilizzata per esplorare i principi logici alla base della ricorsione, utilizzando esempi come la dimostrazione di teoremi matematici o la risoluzione di problemi di ragionamento. In questo modo, l'attività contribuisce a sviluppare le competenze degli studenti in diverse discipline, oltre che in informatica.

Può essere adattata/rivolta a studenti di diverse età e indirizzi?

Si, all'interno della nostra unità didattica, l'attività si compone di diversi passi proprio per riuscire a rivolgersi a studenti di diverse età ed indirizzi. Questo consente agli insegnanti di adattare l'attività alle esigenze specifiche dei propri studenti.

Perché questa specifica attività?

Si è scelto di realizzare questa specifica attività perchè la ricorsione è un argomento fondamentale dell'informatica e ha un'ampia gamma di applicazioni. È un concetto complesso che richiede una comprensione profonda del pensiero computazionale. È inoltre uno strumento fondamentale per affrontare argomenti quali la complessità. Gli algoritmi ricorsivi infatti possono essere molto efficienti, ma sono anche molto complessi da analizzare. Comprendere i principi alla base della ricorsione è essenziale per poter utilizzare questo potente strumento in modo efficace. La ricorsione contiene anche aspetti logici che possono aiutare l'apprendimento di concetti più generali che riguardano l'informatica: ad esempio, può essere utilizzata per esprimere il concetto di ricorsività, che è alla base di molti algoritmi e strutture dati. In generale ha in sè molti dei concetti utili per i percorsi di studi successivi alle scuole secondarie.

Una breve descrizione generale

L'attività si articola in tre passi così definiti: il primo passo si concentra sull'introduzione del concetto di ricorsione attraverso un esempio concreto (il famoso esercizio della torre di Hanoi), per introdurre e far interiorizzare il concetto agli studenti ma senza parlare approfonditamente di ricorsione. L'idea è quella di fornire la definizione formale e analizzarla tramite il suddetto esercizio. Il secondo passo riguarda l'implementazione dell'algoritmo ricorsivo legato al primo passo, in un linguaggio di programmazione semplice. Ciò per testare la comprensione e la capacità degli studenti di riprodurre e utilizzare la ricorsione. Questo secondo passo è inoltre importante per esplorare le applicazioni della ricorsione in ambito informatico e può essere seguito da esercizi di pratica via via di crescente difficoltà. Il terzo passo riguarda invece degli esercizi mirati.

Conoscenze richieste

L'attività della torre di Hanoi è pensata per non avere bisogno di nessuna particolare conoscenza pregressa, se non di avere dimestichezza con il ragionamento logico. Serve da starting point per introdurre la ricorsione.

Per quanto riguarda gli esercizi, è sufficiente che gli studenti abbiano basilari conoscenze di matematica e di un qualsiasi linguaggio di programmazione (nel nostro caso il linguaggio scelto è Java) come sapere definire e chiamare funzioni, utilizzare variabili (singole e array) e cicli (for e while).

Elenco dei contenuti

Torre di Hanoi
Ricorsione
Funzioni
Operazioni elementari
Variabili

Obiettivi

L'obiettivo principale di questa unità didattica è quello di introdurre il concetto di *ricorsione* e mostrarne vantaggi e svantaggi. Gli studenti dovranno essere in grado di capire quando è utile utilizzarla e saper risolvere semplici esercizi tramite funzioni ricorsive. Verranno accennati concetti come *complessità* e *confronto con l'iterazione*, per poter fornire un contesto quanto più completo possibile.

In particolare, facendo riferimento al documento "Proposta di Indicazioni Nazionali per l'insegnamento dell'Informatica nella Scuola" del CINI (https://www.consorzio-cini.it/images/Proposta-Indicazioni-Nazionali-Informatica-Scuola-numerata.pdf) i **traguardi di apprendimento** sono:

- T-S-2. riconosce che un algoritmo risolve un problema nella sua generalità;
- T-S-6. definisce, realizza e valida programmi e sistemi che modellano o simulano sistemi fisici o processi familiari del mondo reale o oggetto di studio nelle altre discipline;
- T-S-7. comprende quando la programmazione può costituire una soluzione vantaggiosa.

Gli **obiettivi** sono:

Ambito algoritmi:

- O-S-A-2. usare il ragionamento logico per valutare diversi algoritmi che risolvono lo stesso problema.

Ambito programmazione:

- O-S-P-3. utilizzare condizioni che usano un operatore logico;
- O-S-P-6. progettare e sviluppare programmi modulari che usano procedure e funzioni.

Ambito dati e informazione:

- O-S-D-1. valutare vantaggi e svantaggi di rappresentazioni alternative della stessa informazione.

Materiali e strumenti

- Per la torre di Hanoi:
  - Dischi di dimensioni diverse
  - Tre pioli
  - Un tavolo o una superficie piana
  - (Opzionale) Un calcolatore o un altro dispositivo per visualizzare la soluzione

- Per gli esercizi:
  - Un calcolatore o un altro dispositivo per scrivere e eseguire programmi
  - (Opzionale) Libri o slide per illustrare gli esercizi

Linguaggio scelto e motivazione

Il linguaggio scelto è java poichè risulta più comprensibile e utilizzabile anche avendo basilari conoscenze di programmazione rispetto a linguaggi più a basso livello come il C e non omette dettagli che in fase di apprendimento sono fondamentali, come invece fanno linguaggi più ad alto livello come python. In sostanza Java è un buon compromesso tra la facilità d'uso e la complessità. È abbastanza semplice da essere compreso da parte degli studenti, ma fornisce anche sufficienti dettagli per consentire loro di imparare i concetti fondamentali della programmazione.
Nota: per semplicità, tratteremo da ora in poi i termini *algoritmo* e *funzione* come sinonimi.

Introduzione

La ricorsione è un concetto di programmazione che consente di definire una funzione che richiama sé stessa. È un modo molto potente per risolvere problemi complessi. È molto più complicata della semplice iterazione ma allo stesso tempo è molto più efficiente sia in termini di spazio che di tempo.

La ricorsione si basa sul principio del divide et impera. Questo principio afferma che un problema complesso può essere risolto dividendolo in sotto problemi più piccoli, fino a quando i sotto problemi sono abbastanza semplici da essere risolti direttamente.

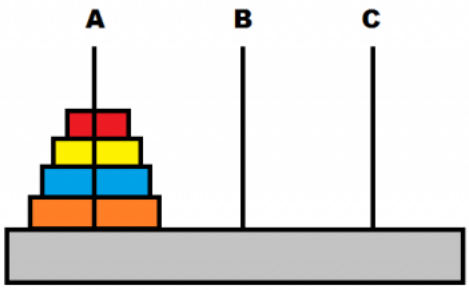
Fornisce le basi per molti argomenti futuri. La ricorsione si articola in due passi: il caso base in cui si fornisce una condizione di terminazione, tramite la quale l'algoritmo smetterà di fare chiamate a sé stesso e la chiamata ricorsiva, appunto, che è ciò che veramente distingue la ricorsione. Nella pratica, il passo base è una condizione che definisce il problema più semplice (si ricordi il principio del divide et impera), per esempio: nel calcolare il fattoriale di un numero, il caso base sarà che il numero ricevuto in input sia 1 (o anche 0, indifferentemente, riuscite a capire perché?). È fondamentale che il passo base sia significativo, poiché se fosse scritto in maniera non adeguata non permetterebbe alla funzione di terminare. La chiamata ricorsiva è invece fatta sullo stesso input di partenza ma diminuito (o aumentato), in quale modo: tornando all'esempio del fattoriale, volendo calcolare il fattoriale di 4 e non essendo nel caso base (per quanto visto precedentemente), l'algoritmo dovrà richiamare sé stesso diminuendo l'input di partenza, quindi non su 4 ma su 3. Nonostante abbiano comportamenti diversi, è comunque possibile combinare ricorsione e iterazione ma questo non sarà un argomento trattato in questa unità.

L'attività didattica è stata pensata in un certo modo e con un ordine ben preciso da seguire per i passi. Consigli sul come somministrare l'attività, sono presenti nella sezione "Guida per gli insegnanti".

Lezione 1: le torri di Hanoi

Costruire 3 piloni (da ora in poi detti *torri*) identici tra di loro. Inserire nella prima torre N dischi forati uno sopra l'altro, dal diametro diverso gli uni dagli altri. Impilarli dal più grande al più piccolo sulla prima torre. Scopo del gioco è *portarli tutti sulla torre di destra*, rispettando due regole:

a) si può muovere un solo disco alla volta;
b) un disco grande non può mai stare sopra un disco più piccolo.



Domande:

1)in base alla spiegazione precedente, posso spostare il disco rosso dal pilone A al pilone C?

2)se spostassi il disco rosso come scritto in precedenza, potrei inserirvi sopra (sulla stessa torre) il disco verde?

3)secondo voi, qual è il procedimento chiave da ripetere per arrivare alla soluzione?

4) ora provate voi, sperimentando in base alle risposte precedenti, come spostiamo tutti i dischi da una torre ad un'altra? (Se non avete riprodotto l'esercizio, simulate su un foglio di carta con tutti i passaggi)

Lezione 2: codice delle torri di Hanoi

Nota: prima di esporre l'esercizio su codice, accertarsi che gli studenti abbiano capito la teoria dietro l'esercizio della torre di Hanoi.

l'esercizio tratterà il codice dell'esercizio precedente sulle torri di Hanoi. Non verterà sulla scrittura di codice poiché troppo complesso ma gli studenti dovranno rispondere alle domande sul codice fornito. Presentiamo ora il codice in linguaggio java:

```
public class Main {
    static void towerOfHanoi(int n, char from_rod, char to_rod, char helper_rod)
    {
        if (n == 1)
        {
            System.out.println("Take disk 1 from rod " +  from_rod + " to rod " + to_rod);
            return;
        }
        towerOfHanoi(n-1, from_rod, helper_rod, to_rod);
        System.out.println("Take disk " + n + " from rod " +  from_rod + " to rod " + to_rod);
        towerOfHanoi(n-1, helper_rod, to_rod, from_rod);
    }

    public static void main(String args[])
    {
        int n = 5;
        towerOfHanoi(n,'A','C', 'B');
    }

}
```

Domande:

1)è una funzione ricorsiva o iterativa? Perché?

2)perché il primo if ha come condizione (n==1)? Facendo riferimento alla teoria della ricorsione, cosa rappresenta questo passo?

3)nel caso in cui fosse una funziona ricorsiva, se viene modificato, come e perché viene modificato il parametro in input? (es. aumenta la dimensione, diminuisce la dimensione, dimezza, rimane uguale..).

4)ora provate voi a scrivere la vostra versione su codice della torre di Hanoi ma con un numero a scelta di dischi e torri.

Guida per gli insegnanti
Consigli su come utilizzare il materiale didattico
Nota: questi sono consigli e in quanto tali vanno trattati per quello che sono. Le attività sono state pensate in un certo modo ma viene data all'insegnate totale libertà di adattarle alla classe.

L'attività della lezione 1 è pensata per essere fisicamente riprodotta in classe, ciò è caldamente consigliato per favorire un primo approccio all'argomento quanto più leggero e guidato possibile. In alternativa è possibile svolgere l'esercizio direttamente su carta o con l'ausilio di libri o slide. In questo modo la spiegazione e risoluzione dell'esercizio risulterà probabilmente più lunga, in quanto alcuni studenti potrebbero avere difficoltà ad immaginarsi l'esercizio senza perdere nessun dettaglio, ma comunque efficiente.
Al termine dell'attività accertarsi che tutti gli studenti abbiano capito quantomeno l'idea di base della ricorsione. All'occorrenza ritardare l'introduzione del secondo e **NON INTRODURLO** senza la certezza della comprensione del primo da parte della classe.

Per la lezione 2 si consiglia di mostrare visivamente agli studenti sia lo svolgimento dell'esercizio, attraverso slide, libri o disegni alla lavagna, sia lo stato delle torri dopo ogni spostamento di disco. Utilizzando materiale precedente riguardante l'iterazione, mostrare quanto più possibile le differenze tra i due approcci esaltando pregi e difetti di uno e dell'altro. Un suggerimento potrebbe essere mostrare la semplicità di scrittura di codice iterativo a discapito di un codice ricorsivo che a primo impatto potrebbe sembrare più criptico. Per quanto riguarda invece la ricorsione si suggerisce di esaltare la compattezza del codice a fronte della lunghezza di quello iterativo, inoltre cercare di enfatizzare il ragionamento che sta alla base di questa tipologia di algoritmo, poiché aiuta lo sviluppo del pensiero logico e computazionale.

È consigliato presentare le attività nell'ordine in cui sono presentate in questo documento. Precisiamo che sono divise in lezione 1 e 2 per questioni di praticità ma non è obbligatorio dividerle in giorni diversi. Il focus deve essere posto comunque sulla prima lezione in quanto getta la basi per l'apprendimento della seconda e per lo svolgimento degli esercizi presentati nella terza.

Alla fine della prima lezione, si consiglia di chiedere ad uno o più studenti di provare a spiegare il funzionamento delle torri al resto della classe. Chi deve spiegare, ha prima di tutto capito come deve essere fatto? Se sì, c'è stato qualcuno, tra quelli che non avevano capito, che ora è riuscito a capire? Se no, c'è stato qualcuno che, tra quelli che avrebbero dovuto sentire la spiegazione, è riuscito a spiegarlo a sua volta?

Per quanto riguarda la domanda 4 della seconda lezione, non è importante che l'esercizio produca l'output corretto, riprodurre l'esercizio della torre di Hanoi può risultare molto ostico, soprattutto se utilizzato come argomento di introduzione alla ricorsione. Gli studenti devono essere inoltre liberi di prendere spunto dal codice dell'esercizio 2. Accertarsi che l'idea di base degli algoritmi prodotti sia corretta (chiamate ricorsive, caso base ecc.).

Snodi e indicatori

Sono i **passaggi cognitivi** obbligati per arrivare alla meta, cioè per rispondere correttamente alle domande, svolgere un compito, risolvere un problema. Per questa attività gli snodi e gli indicatori individuati sono:

1)comprendere qual è il compito del caso base e quale quello del caso ricorsivo. Un indicatore per capire se lo studente ha compreso questo problema potrebbe essere il fatto che innanzitutto distingua i due passaggi, quindi provare a simulare graficamente l'algoritmo (anche mediante il disegno precedente), per avere una visione generale dell'esercizio.

2)capire cosa passare come parametro della funzione. Un indicatore in questo caso potrebbe essere il fatto che lo studente riesca a capire come modificare l'input iniziale, per poi passarlo alla chiamata ricorsiva.

Misconceptions

Presentiamo ora alcuni errori ("misconception") che potrebbero presentarsi durante la spiegazione della ricorsione. Presenteremo alcune misconception individuate da noi dopo un'attenta analisi dei documenti presenti in letteratura. Innanzitutto, gli studenti potrebbero erroneamente pensare che iterazione e ricorsione abbiano esattamente lo stesso comportamento. Ciò è sbagliato, infatti la ricorsione si basa sul divide et impera, cioè la prassi di dividere il problema principale in sotto problemi di più facile risoluzione per poi tornare all'inizio e risolvere il primo; l'iterazione è invece essenzialmente la ripetizione di istruzioni. Continuando con questo dualismo iterazione-ricorsione alcuni studenti potrebbero pensare che qualsiasi problema ricorsivo abbia una sua versione iterativa e viceversa ma ciò è sbagliato in quanto non possiamo affermarlo in generale. Si può certamente dire che quest'affermazione è vera in molti casi ma pensare che sia vera in assoluto è un errore in quanto ci sono problemi ricorsivi che non hanno una loro versione iterativa e viceversa (basti pensare a quei problemi che non possono essere divisi in sotto problemi). Passando agli esercizi veri e propri, gli studenti potrebbero pensare che il caso base non sia obbligatorio quando è invece fondamentale per permettere all'algoritmo di terminare oppure potrebbero pensare vada messo obbligatoriamente prima o obbligatoriamente dopo la chiamata ricorsiva quando in realtà non c'è una vera regola. Tra le altre cose, potrebbe esserci molta difficoltà nel cercare di capire l'ordine delle operazioni nelle chiamate ricorsive in successione; questo è probabilmente il problema maggiormente diffuso e di più difficile gestione. Il suggerimento è quello di riprendere la teoria dall'inizio, se serve anche dalla rappresentazione fisica dell'esercizio di Hanoi e cercare di capire dove il ragionamento degli studenti si distacca da come dovrebbe essere.

Esercizi

Proponiamo ora alcuni esercizi di difficoltà crescente. Saranno qui fondamentali conoscenze di base di matematica e programmazione. L'intenzione di questi esercizi è proprio combattere le misconception sopra descritte. Infatti, se risulta difficile (se non dopo un'attenta analisi e ricerca) individuare esercizi compatibili con la ricorsione ma non con l'iterazione (o viceversa), gli esercizi mettono in risalto il concetto di divide et impera, della divisione in sotto problemi e delle chiamate ricorsive. Mostrano come il caso base sia fondamentale (è presente in tutti gli esercizi e deve esserlo anche in quelli proposti successivamente dall'insegnante) e, tramite questi, l'insegnante dovrebbe mostrare che non è importante che venga messo prima o dopo la chiamata ricorsiva. Abbiamo ritenuto le chiamate ricorsive in successione essere un argomento troppo avanzato per questa trattazione, per rinforzare questo concetto si consiglia di riprendere l'esercizio fornito sulle torri di Hanoi. Per coerenza si consiglia di mantenere il linguaggio Java.

Esercizio1:
Scrivere un programma che dato un numero calcola la somma dei primi N numeri pari positivi in maniera iterativa. Riscrivere l'algoritmo in versione ricorsiva.
Nota: focus di questo esercizio è la misconception secondo cui iterazione e ricorsione lavorano allo stesso modo. Bisogna qui cercare di rendere ben visibili la differenza tra un modo e l'altro di scrivere l'algoritmo. Secondo il **Block Model**, possiamo utilizzare questo esercizio per capire nel complesso come si comportano e quale struttura hanno i due algoritmi, con focus, ovviamente, in quello ricorsivo (Text Surface / Macrostructure).

Esercizio2:
Scrivere il codice di una funzione ricorsiva f(n) che restituisce 0 nel caso n sia dispari o zero, 1+f(n/2) altrimenti.
Nota: l'obiettivo qui è la distinzione tra il caso base e il passo ricorsivo, cercando di arginare la relativa misconception. Utilizzando lo strumento del Block model, si consiglia di usare questo strumento per mettere in risalto la relazione tra i due blocchi (Text Surface / Relationships).

Esercizio3:
Scrivere il codice di una funzione ricorsiva f(n) che restituisce:
– n nel caso n sia minore di 10,
– il risultato di f applicata alla somma delle cifre di n se n è pari,
– f(n-1) altrimenti.
Individuare il caso base e provare a spostarlo in diversi punti dell'algoritmo. Funziona sempre?
Nota: questo esercizio è pensato e proposto per far sperimentare agli studenti la possibilità di spostare il caso base in diversi punti del codice (ed arginare la relativa misconception). Secondo il Block Model, può essere classificato come esercizio per capire la sequenza di chiamate di metodi (Program Execution / Relationships).

Rubrica valutativa

Forniamo una rubrica valutativa liberamente modificabile dal docente.

	Livello 1	**Livello 2**	**Livello 3**
Concetti fondamentali	Lo studente non ha compreso il concetto base della ricorsione.	Lo studente ha una comprensione superficiale dei concetti fondamentali della ricorsione.	Lo studente ha una comprensione approfondita dei concetti fondamentali della ricorsione.
Capacità di risoluzione dei problemi	Lo studente non è in grado di risolvere problemi semplici utilizzando la ricorsione.	Lo studente è in grado di risolvere problemi semplici utilizzando la ricorsione.	Lo studente è in grado di risolvere problemi complessi utilizzando la ricorsione.
Codifica	Lo studente non è in grado di scrivere codice ricorsivo che funzioni correttamente per problemi semplici.	Lo studente è in grado di scrivere codice ricorsivo che funzioni correttamente per problemi semplici.	Lo studente è in grado di scrivere codice ricorsivo che funzioni correttamente per problemi complessi.