|  |
| --- |
| **Pensiero computazionale per la scuola secondaria inferiore: imparare a conoscere gli algoritmi** |

**Durata stimata:** 1 ora

**Età:** studenti delle scuole secondarie inferiori

**Obiettivi di apprendimento, abilità e competenze**

Gli obiettivi principali sono familiarizzare gli studenti con il pensiero computazionale e assisterli nello sviluppo del ragionamento matematico. L'obiettivo è ragionare sistematicamente su un problema e pensare a come un modo di risolverlo possa essere convertito in codice informatico. Gli studenti imparano a conoscere il concetto di algoritmo.

**Attività e ruoli**

Gli studenti giocheranno a giochi numerici e risponderanno a domande che stimolano il loro ragionamento matematico. In questa lezione lavoreranno in coppie. I compiti principali dell'insegnante sono fungere da istruttore, monitorare la classe durante l'attività e fornire assistenza quando necessario. Il feedback può essere fornito durante o alla fine dell'attività.

**Materiale occorrente**

Per eseguire questa attività con gli studenti è sufficiente una lavagna interattiva con connessione a Internet per dimostrare e consultare le attività su Scratch, computer portatili o da tavolo per giocare a un gioco su Scratch, più carta e penna.

**Spazio di apprendimento**

Aula scolastica e laboratorio informatico

**Descrizione dell'attività**

Giocare una versione di "Indovina il numero" con gli allievi in classe. Spiegare che si sta pensando a un numero compreso tra 0 e 127, invitando gli studenti, a turno, a cercare di indovinare il numero prescelto. Ogni volta, dire se il numero indicato è troppo alto, troppo basso o corretto. Utilizzare la lavagna ad ogni turno per scrivere i numeri proposti e se sono troppo alti o troppo bassi.

Far riflettere gli alunni su quanto le loro ipotesi siano state utili nell'aiutarli a indovinare rapidamente il numero sconosciuto. Potrebbe essere utile fornire qualche esempio. Ipotizziamo che la classe sappia che il numero è uguale o superiore a 64; se il giocatore successivo dice 96, le possibilità si dimezzano. Se gli studenti dicono 65, è probabile che non otterranno maggiori informazioni sul numero da indovinare.

Formare coppie di alunni per farle giocare una contro l'altra e cercare di trovare una buona strategia per vincere la partita nel minor numero possibile di tentativi. Dopo che gli alunni avranno avuto la possibilità di esercitarsi, chiedere loro di spiegare alla classe le strategie applicate, chiedendo agli altri alunni di commentare l'efficacia dei metodi proposti.

Mostrare agli alunni il programma Scratch su <https://scratch.mit.edu/projects/240290316/> e lasciare loro il tempo necessario a capire cosa fa il programma *senza* eseguirlo. C'è una coppia di alunni in grado di spiegare il programma alla classe? Il programma riproduce semplicemente il ruolo dell'insegnante nella prima versione del gioco, scegliendo un numero e poi rispondendo "troppo basso", "corretto" o "troppo alto" ad ogni tentativo. Spiegare che in precedenza è stato seguito un algoritmo e che il programma implementa semplicemente lo stesso algoritmo, ma sotto forma di codice. Potrebbe essere utile mostrare agli studenti l'algoritmo come diagramma di flusso.

Chiedere agli alunni di provare a elaborare la strategia più efficace per vincere il gioco nel minor numero di tentativi. Di cosa devono tenere traccia ad ogni turno? Su cosa dovrebbero basare i loro tentativi? Se il numero che ipotizzano è troppo basso, che numero dovrebbero dire dopo? Se è troppo alto, che numero dovrebbero dire dopo?

Chiedere agli studenti di pensare a un modo per programmare Scratch per indovinare velocemente il numero *pensato da loro*, invece del contrario. Se gli alunni non hanno molta esperienza di programmazione su Scratch, potrebbe essere meglio partire dalla versione da completare su <https://scratch.mit.edu/projects/240290555/>, ma se sono già abbastanza esperti potrebbero lavorare con il loro compagno per programmare qualcosa di simile. Una versione completa è disponibile su <https://scratch.mit.edu/projects/240290793/>.

Se gli studenti completano il compito, invitarli a modificare il codice in modo che tenga traccia del numero di tentativi utilizzati. Gli studenti possono modificare il codice e vedere quanto velocemente Scratch può indovinare numeri inferiori alle migliaia? O inferiori a un milione? In aggiunta si potrebbe modificare il codice affinché aggiorni gli elenchi dei numeri ipotizzati e la risposta del giocatore ad ogni turno.

In conclusione, analizzare come gli studenti hanno utilizzato i concetti del pensiero computazionale nella lezione: hanno dovuto usare la logica per ragionare sui numeri rimasti in ogni turno e su come Scratch dovrebbe rispondere all'input del giocatore. Il programma si basa su un classico algoritmo di ricerca: la ricerca binaria. Si scompone il problema originario dell'indovinare numeri compresi tra 0 e 127 in una serie di problemi progressivamente più piccoli (questo processo è chiamato algoritmo di "divide et impera"). Lo stato del problema è rappresentato dai numeri più piccoli e più grandi rimasti: questa astrazione è tutto ciò di cui il programma deve tenere traccia ad ogni turno.

**Nome dell'autore: Miles Berry**