

CLASSE: 1 e 2 Liceo Scienze Applicate

MATERIA: Informatica

COMPETENZE CHIAVE DI CITTADINANZA
relative all'intero curriculum della disciplina

1. competenza alfabetica funzionale (saper comunicare in forma scritta e orale)
2. competenza digitale (conoscere e utilizzare programmi di videoscrittura, fogli di calcolo e presentazioni)
3. competenza personale, sociale e capacità di imparare ad imparare (applicare strategie efficaci di apprendimento; lavorare in gruppo in maniera costruttiva)
4. competenza imprenditoriale (risolvere problemi)

COMPETENZE	ABILITA'	CONOSCENZE
<ul style="list-style-type: none">● Riconoscere la struttura di un sistema di elaborazione● Distinguere le tipologie dei computer in base alle attività e alle caratteristiche● Riconoscere il ruolo dei componenti di un sistema di elaborazione● Convertire numeri e codici rappresentati secondo sistemi di numerazione diversi	<ul style="list-style-type: none">● Identificare i componenti hardware di un computer● Utilizzare in modo appropriato la terminologia tecnica● Confrontare le caratteristiche tecniche principali dei singoli componenti● Codificare e decodificare numeri e codici	Introduzione all'architettura del computer <ul style="list-style-type: none">● L'architettura ed i componenti fondamentali di un computer● Le memorie: tipologie e capacità● Sistema di numerazione decimale binario ottale esadecimale.● Il software
<ul style="list-style-type: none">● Utilizzare le procedure necessarie per gestire le impostazioni	<ul style="list-style-type: none">● Utilizzare in modo appropriato la terminologia tecnica	Il funzionamento del Sistema Operativo <ul style="list-style-type: none">● Riconoscere il ruolo dei sistemi operativi e quali sono i tipi più diffusi

<p>dello schermo e del desktop</p> <ul style="list-style-type: none"> • Utilizzare le tecniche di drag and drop per gestire i file e le cartelle • Essere in grado di estrarre le caratteristiche del computer in uso • Gestire i file system di windows e di Linux Ubuntu 	<ul style="list-style-type: none"> • Riconoscere le caratteristiche principali del sistema operativo 	<ul style="list-style-type: none"> • Conoscere le caratteristiche principali del desktop di windows e di linux • Saper distinguere i tipi di file in base all'estensione • Riconoscere il significato e la struttura delle directory • Identificare i principali elementi dell'interfaccia grafica di windows e linux ubuntu. • Identificare il significato dei caratteri jolly
<ul style="list-style-type: none"> • Conoscere e saper utilizzare Microsoft word. • Conoscere e saper utilizzare PowerPoint • Conoscere e saper utilizzare excel, saper applicare formule e funzioni corrette in base al contesto 	<ul style="list-style-type: none"> • Realizzare documenti word e lettere circolari • Realizzare presentazioni multimediali con PowerPoint • Applicare le funzioni condizionali ai fogli di lavoro in situazioni complesse. 	<p>Elaborazione di documenti digitali</p> <ul style="list-style-type: none"> • Elementi fondamentali del documento Word / Google Documenti • Elementi fondamentali di una presentazione PowerPoint/Google Slides • Riconoscere le caratteristiche del foglio di calcolo di Excel o dei fogli di Google (Google sheets) • Orientarsi nell'ambiente di lavoro Excel/Fogli di Google inserendo formule e funzioni per la produttività personale.
<ul style="list-style-type: none"> • Descrivere la soluzione di semplici problemi mediante algoritmi. • capire il concetto di programma e codice sorgente 	<ul style="list-style-type: none"> • Utilizzare i diagrammi di flusso per codificare gli algoritmi • Saper scrivere compilare ed eseguire un primo semplice programma. 	<p>Principi base di programmazione</p> <ul style="list-style-type: none"> • Concetto di Linguaggio di programmazione • Concetto di algoritmo • Conoscere la simbologia dei diagrammi di flusso.
<ul style="list-style-type: none"> • Editare testare e collaudare un programma in Python • Effettuare l'input dei dati • Effettuare l'output dei dati • Scrivere programmi con 	<ul style="list-style-type: none"> • Installare e configurare l'ambiente di sviluppo (Pycharm) • Disporre l'output sullo schermo • Utilizzare le variabili nei programmi • Commentare il codice del programma • Utilizzare variabili intere, reali e bool 	<p>Programmare in Python</p> <ul style="list-style-type: none"> • Conoscere i diversi formati di un programma • Descrivere le diverse fasi di sviluppo di un programma • Comprendere il concetto di variabile • Conoscere le istruzioni di comunicazione con l'utente

<p>più istruzioni in sequenza.</p>	<ul style="list-style-type: none"> • Utilizzare gli operatori / e % sui numeri interi 	<ul style="list-style-type: none"> • Comprendere l'importanza del commento del codice
<ul style="list-style-type: none"> • Riconoscere quando utilizzare selezioni annidate • Codificare la selezione semplice e doppia • Effettuare l'annidamento delle istruzioni • Utilizzare le variabili di tipo bool • Utilizzare gli operatori logici 	<ul style="list-style-type: none"> • Scrivere codice con istruzioni condizionali • Scrivere codice con blocchi di istruzioni annidate • Combinare più selezioni con condizioni logiche composte • Scrivere codice utilizzando l'istruzione di selezione multipla 	<p>Blocco di Selezione (if then else elif)</p> <ul style="list-style-type: none"> • Conoscere l'istruzione di selezione semplice e doppia • Conoscere le variabili di tipo bool e gli operatori logici • Comprendere il concetto di annidamento
<ul style="list-style-type: none"> • Saper scegliere il tipo di iterazione adeguato alle diverse situazioni. • Codificare l'iterazione finita ed indefinita. • Codificare programmi con cicli annidati. 	<ul style="list-style-type: none"> • Scrivere programmi con selezioni e iterazioni annidate. • Progettare programmi con cicli e cicli annidati. • Utilizzare diverse tipologie di iterazione nello stesso programma. 	<p>Blocco di Iterazione (for,while)</p> <ul style="list-style-type: none"> • Comprendere il concetto di iterazione. • Conoscere le diverse tipologie di iterazione
<ul style="list-style-type: none"> • Definire una funzione in Python. • Distinguere tra parametri formali e parametri attuali. 	<ul style="list-style-type: none"> • Scrivere programmi in Python suddividendoli in moduli tramite le funzioni. • Utilizzare funzioni predefinite nei programmi. • Scrivere funzioni ricorsive. • Trasformare funzioni iterative in funzioni ricorsive. 	<p>Le Funzioni</p> <ul style="list-style-type: none"> • Comprendere il concetto di funzione nella programmazione. • Conoscere il concetto di invocazione di una funzione e di passaggio dei parametri. • Conoscere i Record di Attivazione ed il loro funzionamento. • Comprendere il concetto di ricorsione e la differenza rispetto all'iterazione.

<ul style="list-style-type: none"> ● Saper definire una stringa in Python e saperla manipolare. 	<ul style="list-style-type: none"> ● Scrivere programmi in Python che lavorano con le stringhe. ● Utilizzare le principali operazioni e funzioni con le stringhe. 	Le stringhe <ul style="list-style-type: none"> ● Conoscere la struttura dati Stringa. ● Comprendere l'utilità del tipo di dato Stringa ● Conoscere quali operazioni sono possibili su una stringa.
--	---	--

N.B.: le conoscenze devono essere riferite alle Linee Guida per i Licei del 2010 - DPR 89

CLASSE: **3 e 4 Liceo Scienze Applicate**

MATERIA: **Informatica**

COMPETENZE CHIAVE DI CITTADINANZA

relative all'intero curriculum della disciplina

1. competenza alfabetica funzionale (saper comunicare in forma scritta e orale)
2. competenza digitale (conoscere e utilizzare programmi per lo sviluppo e il testing di programmi)
3. competenza personale, sociale e capacità di imparare ad imparare (applicare strategie efficaci di apprendimento; lavorare in gruppo in maniera costruttiva)
4. competenza imprenditoriale (risolvere problemi)

COMPETENZE	ABILITA'	CONOSCENZE
<ul style="list-style-type: none">• Saper creare una lista in Python saperla utilizzare ed iterare.• Comprendere la sintassi di List Comprehension e sapere quando utilizzarla.	<ul style="list-style-type: none">• Scrivere programmi/funzioni in Python che lavorano con le Liste.• Utilizzare le principali operazioni e funzioni con le Liste.• Utilizzare la sintassi di list comprehension per costruire Liste.	Le liste <ul style="list-style-type: none">• Conoscere il tipo di dato Lista.• Comprendere l'utilità del tipo di dato Lista.• Conoscere quali operazioni sono possibili su una lista e in che contesti viene utilizzata.• Conoscere la sintassi di List Comprehension.
<ul style="list-style-type: none">• Saper quando è necessario l'utilizzo di una tupla in un programma Python.	<ul style="list-style-type: none">• Fare "unpacking" di tuple.	Tuple <ul style="list-style-type: none">• Conoscere il tipo di dato Tupla.
<ul style="list-style-type: none">• Saper creare un dizionario in Python, saperlo utilizzare ed iterare.• Comprendere la sintassi di Dictionary Comprehension e	<ul style="list-style-type: none">• Scrivere programmi/funzioni in Python che lavorano con i Dizionari.• Utilizzare le principali operazioni e funzioni con i dizionari.• Utilizzare la sintassi di Dictionary	Dizionari <ul style="list-style-type: none">• Conoscere il tipo di dato Dizionario.• Comprendere l'utilità del tipo di dato Dizionario.• Conoscere quali operazioni sono possibili su un Dizionario e in che contesti viene utilizzato.

sapere quando utilizzarla.	Comprehension per costruire Dizionari.	<ul style="list-style-type: none"> Conoscere la sintassi di Dictionary Comprehension.
<ul style="list-style-type: none"> Saper creare un set in Python, saperlo utilizzare ed iterare. 	<ul style="list-style-type: none"> Scrivere programmi/funzioni in Python che lavorano con i Set. Utilizzare le principali operazioni e funzioni sui Set. 	Set <ul style="list-style-type: none"> Conoscere il tipo di dato Set. Comprendere l'utilità del tipo di dato Set. Conoscere quali operazioni sono possibili su un Dizionario e in che contesti viene utilizzato.
<ul style="list-style-type: none"> Saper aprire un file in lettura ed in scrittura mediante la primitiva open Saper usare le funzioni dell'oggetto file in Python. 	<ul style="list-style-type: none"> scrivere programmi che facciano input e output da e su file. 	Lettura/Scrittura su file (Python) <ul style="list-style-type: none"> Conoscere le diverse modalità di apertura di un file e la loro differenza.
<ul style="list-style-type: none"> Definire in Python una classe con attributi e metodi. Classificare classi e relazioni tra esse. Riconoscere e progettare la gerarchia tra classi. 	<ul style="list-style-type: none"> Usare la progettazione orientata agli oggetti per i programmi complessi. Applicare il concetto di astrazione per modellare le classi. Individuare la specializzazione e la generalizzazione di una classe. Applicare i concetti di ereditarietà e di polimorfismo nello sviluppo di programmi. 	Programmazione ad Oggetti <ul style="list-style-type: none"> Conoscere gli elementi teorici del paradigma di programmazione ad oggetti. Comprendere il concetto di astrazione nell'ambito della programmazione ad oggetti. Comprendere il concetto di ereditarietà e di polimorfismo.

N.B.: le conoscenze devono essere riferite alle Linee Guida per i Licei del 2010 - DPR 89

CLASSE: **5 Liceo Scienze Applicate**

MATERIA: **Informatica**

COMPETENZE CHIAVE DI CITTADINANZA¹

relative all'intero curriculum della disciplina

1. competenza alfabetica funzionale (saper comunicare in forma scritta e orale)
2. competenza digitale (conoscere e utilizzare programmi di videoscrittura, fogli di calcolo e presentazioni)
3. competenza personale, sociale e capacità di imparare ad imparare (applicare strategie efficaci di apprendimento; lavorare in gruppo in maniera costruttiva)
4. competenza imprenditoriale (risolvere problemi)

COMPETENZE	ABILITA'	CONOSCENZE
<ul style="list-style-type: none">• Utilizzare lo schema dei dati E-R• Utilizzare il modello logico dei dati• Utilizzare gli operatori relazionali• Rispettare le regole di integrità• Effettuare ricerche nelle tabelle• Interrogare il database mediante query• Definire un database con il DBMS MySQL• Realizzare relazioni tra tabelle	<ul style="list-style-type: none">• Applicare le gerarchie di generalizzazione• Utilizzare le potenzialità di una base di dati relazionale• Applicare le regole di normalizzazione• Progettare basi di dati relazionali	Database <ul style="list-style-type: none">• Comprendere le necessità dei database• Conoscere i vantaggi di un DBMS• Individuare le entità e le relazioni all'interno di una situazione complessa• Acquisire la conoscenza degli aspetti funzionali e organizzativi di una base di dati• Conoscere il concetto di dipendenza funzionale• Conoscere le motivazioni alla base della normalizzazione• Conoscere il significato di database• Riconoscere il ruolo dei DBMS• conoscere la struttura di tabelle record e campi• conoscere il significato di relazione e di chiave primaria

<ul style="list-style-type: none"> • realizzare query personalizzate 		
<ul style="list-style-type: none"> • Convertire in base 10 un numero relativo rappresentato in complemento a due. • Convertire in base 10 un numero reale rappresentato in virgola fissa ed in virgola mobile • Saper distinguere errore relativo ed errore assoluto, saper arrotondare e troncare. • Codificare l'algoritmo babilonese e di newton per il calcolo della radice quadrata. • Codificare il metodo di bisezione per trovare gli zeri di una funzione. 	<ul style="list-style-type: none"> • Saper rappresentare un numero relativo in complemento a due • Saper rappresentare un numero reale in virgola mobile. • Calcolare errore assoluto ed errore relativo. • Saper implementare il calcolo numerico della radice quadrata di un numero • Saper implementare il metodo di bisezione. 	<p>Algoritmi del calcolo numerico</p> <ul style="list-style-type: none"> • Comprendere le diverse rappresentazioni dei numeri nel calcolatore. • Comprendere il concetto di errore assoluto ed errore relativo dovuto all'approssimazione. • Calcolo approssimato della radice quadrata. • Calcolo approssimato della radice di un'equazione con il metodo di bisezione

<ul style="list-style-type: none"> • Classificare le reti in base alla topologia • Individuare i diversi dispositivi di rete • Saper classificare le reti in base ai mezzi trasmissivi • Classificare le tecniche di trasferimento dell'informazione. 	<ul style="list-style-type: none"> • Riconoscere le funzioni in relazione ai diversi livelli protocollari • Confrontare il modello ISO/OSI con il modello TCP/IP • Delineare i compiti del modello ISO/OSI e TCP/IP • Scomporre una rete in sottoreti 	<p>Le Reti</p> <ul style="list-style-type: none"> • Conoscere gli elementi fondamentali di una rete • Conoscere le topologie di rete • Acquisire il concetto di protocollo • Conoscere come si è sviluppata internet ed il protocollo TCP/IP • Il confronto tra i livelli ISO/OSI e TCP/IP • I 4 strati del modello TCP/IP e le loro funzioni • La struttura degli indirizzi IP • Le classi degli indirizzi IP • Differenze tra indirizzamento pubblico e privato.
---	---	--

¹ – ² vedi file dei materiali

N.B.: le conoscenze devono essere riferite alle Linee Guida per i Licei del 2010 - DPR 89