Programmazione curricolare per competenze, abilità e conoscenze

DISCIPLINA : INFORMATICA – Classe PRIMA

COMPETENZE	ABILITÀ	CONOSCENZE
Comprendere le ragioni che hanno prodotto lo sviluppo scientifico e tecnologico nel tempo, in relazione ai bisogni e alle domande di conoscenza dei diversi contesti, con attenzione critica alle dimensioni tecnico-applicative delle conquiste scientifiche Essere consapevole delle potenzialità e dei limiti delle tecnologie nel contesto culturale e sociale in cui vengono applicate Essere in grado di utilizzare criticamente e consapevolmente gli strumenti informatici e telematici nelle attività di studio Utilizzare le strategie del pensiero negli aspetti didattici e algoritmici per affrontare situazioni problematiche elaborando opportune soluzioni Analizzare dati e interpretarli sviluppando deduzioni e ragionamenti sugli stessi anche con l'ausilio di rappresentazioni grafiche Saper usare le tecniche di formalizzazione algoritmica in relazione all'analisi dei dati e alla modellizzazione di specifici problemi scientifici	Comprendere la struttura logico-funzionale e fisica di un computer Saper creare e gestire file e cartelle con Windows Saper operare con numeri binari e convertire numeri decimali in basi diverse Comprendere e utilizzare le tecniche per la rappresentazione dei dati all'interno di un computer Riconoscere e saper utilizzare con consapevolezza gli strumenti che ruotano attorno al mondo di Internet Saper impostare documenti di testo formattandoli adeguatamente. Saper disporre oggetti diversi all'interno di documenti testuali. Saper realizzare fogli di calcolo usando formule e funzioni. Realizzare grafici su dati relativi a fogli di calcolo. Realizzare presentazioni. Comprendere i passi logici necessari alla risoluzione di un problema, scomponendolo in sottoproblemi. Effettuare una corretta analisi dei dati coinvolti nel problema (input, output e lavoro) Individuare e formalizzare strategie risolutive secondo formalismi specifici Saper impostare l'algoritmo risolutivo del problema dato, utilizzando correttamente le tre strutture per il controllo del flusso.	(AC) Informazioni, dati e loro codifica. Sistemi di numerazione. ▪ Il sistema binario ▪ Conversione di numeri da un sistema di numerazione ad un altro ▪ La rappresentazione dei caratteri: ASCII e UNICODE ▪ La rappresentazione delle immagini ▪ La rappresentazione dei suoni ▪ La compressione dei dati (AC) Architettura e componenti hardware e software di un computer. ▪ Architettura del computer (Von Neumann) (SO) Funzioni di un sistema operativo, processi, gestione della memoria e file system. ▪ Ruolo dei SO ▪ Programmi e processi ▪ Architettura di un SO (onion skin) ▪ L'interfaccia grafica (Windows in lab) ▪ Il file system (Windows in lab) ▪ Il software: copyright e licenze d'uso (IS) Terminologia legata alla rete e ai servizi di Internet. Il cloud e i servizi di cloud computing ▪ Internet e il web ▪ La posta elettronica ▪ La sicurezza informatica (DE) Tipologia dei programmi applicativi, software di produttività e tipi di documenti. Tecniche di gestione e formattazione di documenti testuali e di calcolo. Formule e funzioni per impostare calcoli. Tecniche per realizzare presentazioni efficaci. ▪ Le applicazioni per il word processing: azioni di base su un testo

		▪ La gestione delle immagini ▪ Le tabelle ▪ Le applicazioni di foglio di calcolo: formule e funzioni ▪ Applicazione di alcune funzioni ▪ Subtotali e copie speciali ▪ I grafici ▪ Le applicazioni per fare presentazioni ▪ Creare pagine master (modelli) ▪ Inserire immagini, disegni e schemi ▪ Progettare diapositive per l'uso interattivo (AL) Concetto di algoritmo. Fasi risolutive di un problema e loro rappresentazione. ▪ Analisi, astrazione e modello del problema ▪ L'algoritmo e le sue proprietà ▪ Rappresentazione di algoritmi ▪ Concetto di variabile ▪ Teorema di Jacopini-Bohm ▪ Problemi con sequenza ▪ Problemi con selezione ▪ Problemi con iterazione ▪ Concetto di contatore ▪ La tabella di traccia

DISCIPLINA : INFORMATICA – Classe SECONDA

COMPETENZE	ABILITÀ	CONOSCENZE
Individuare le analogie e le differenze tra diversi linguaggi di programmazione Avere una sufficiente padronanza di un linguaggio di programmazione per sviluppare semplici ma significative applicazioni di calcolo in ambito scientifico	Saper tradurre un algoritmo in linguaggio C Saper impostare e comprendere la struttura di base di un programma C Saper gestire un IDE per lo sviluppo di applicazioni C Riuscire a codificare programmi corretti con dati, istruzioni, operatori e strutture di controllo Saper codificare in linguaggio C algoritmi che si servono del costrutto di selezione e di iterazione Saper scegliere fra i vari costrutti iterativi Realizzare algoritmi che fanno uso di strutture dati	(AL) Fondamenti di programmazione. ▪ Storia della programmazione ▪ Dal linguaggio naturale al linguaggio macchina ▪ Paradigma imperativo: linguaggio procedurale e linguaggio orientato agli oggetti ▪ Linguaggio interpretati e linguaggi compilati ▪ Nascita ed evoluzione del linguaggio C ▪ Sintassi del C: casting delle variabili, tipi di dati, definizione di funzione e di libreria ▪ Utilizzo di un IDE per la scrittura di programmi in C

	Implementare vettori in C Realizzare algoritmi che usano tecniche di ricerca e di ordinamento di vettori Saper gestire le stringhe e i vettori di caratteri in C	▪ Struttura di base di un programma in C: lettura e scrittura di dati in un programma ▪ Operatori aritmetici utilizzati nel C:+,-,:,*,% ▪ Operatori di assegnamento e di incremento e decremento ▪ Sintassi dell'istruzione if -else ▪ Operatori relazionali ▪ Operatori logici ▪ Selezioni annidate ▪ Selezione multipla (switch) ▪ Strutture iterative: while, do-while, for ▪ Il concetto di struttura dati ▪ La struttura degli array monodimensionali ▪ Le tecniche di gestione degli array ▪ Gli array bidimensionali ▪ Le stringhe come array di caratteri ▪ Il terminatore di stringa ▪ Le sequenze di escape

DISCIPLINA : INFORMATICA – Classe TERZA

COMPETENZE	ABILITÀ	CONOSCENZE
Utilizzare le strategie del pensiero negli aspetti dialettici e algoritmici per affrontare situazioni problematiche elaborando opportune soluzioni Utilizzare il linguaggio di programmazione per organizzare e valutare informazioni qualitative e quantitative Conoscere il paradigma di programmazione ad oggetti. Utilizzare la memoria di massa attraverso i file.	Conoscere le caratteristiche del linguaggio Python. Scrivere programmi utilizzando le strutture di controllo. Saper elaborare testo utilizzando il tipo string. Saper codificare programmi che utilizzano le strutture dati. Saper riconoscere la struttura dati adeguata allo scopo del programma. Saper scomporre un problema in sottoproblemi e risolverlo attraverso le funzioni Implementare funzioni in Python Saper gestire il passaggio dei parametri Saper riconoscere problemi che necessitano di ricerca e di ordinamento. Saper implementare gli opportuni algoritmi. Saper progettare e codificare classi seguendo i principi della	(AL) Fondamenti di programmazione. ▪ Principali caratteristiche del linguaggio Python ▪ Le variabili ▪ Input/output ▪ Le strutture condizionali e iterative (if, else, elif, while, for) ▪ Il tipo string (accesso, slicing, funzioni e metodi) ▪ Liste ▪ Liste bidimensionali ▪ Set ▪ I vantaggi delle funzioni ▪ Le funzioni: dichiarazione, chiamata e struttura ▪ Parametri della funzione ▪ La funzione main ▪ Ambito di visibilità ▪ I moduli (math e random) ▪ Bubble Sort ▪ Ricerca sequenziale e binaria ▪ Uso di funzioni built-in per l'ordinamento e per la ricerca ▪ OOP: caratteristiche

	programmazione ad oggetti. Saper utilizzare le classi all'interno di programmi. Saper leggere e scrivere file. Saper organizzare dati frutto di elaborazione di file in strutture dati.	▪ Le classi come astrazioni ▪ Costruire gli oggetti ▪ Attributi e metodi ▪ Informazioni private e pubbliche ▪ Incapsulamento ▪ Il ruolo dei file ▪ Aprire e chiudere un file ▪ Leggere e scrivere su un file ▪ La struttura dati dizionario

DISCIPLINA : INFORMATICA – Classe QUARTA

COMPETENZE	ABILITÀ	CONOSCENZE
Utilizzare le strategie del pensiero negli aspetti didattici e algoritmici per affrontare situazioni problematiche elaborando opportune soluzioni Utilizzare il linguaggio e i metodi della matematica per organizzare e valutare informazioni qualitative e quantitative Saper creare semplici ipertesti con tabelle, testi, form usando tag HTML. Saper definire lo stile degli elementi HTML usando CSS. Applicare i concetti fondamentali del modello relazionale delle basi di dati	Costruire oggetti software utilizzando la metodologia OOP Comprendere il meccanismo dell'ereditarietà. Integrare le conoscenze pregresse con la OOP. Conoscere la storia del web e l'evoluzione fino ad oggi. Conoscere i principali linguaggi utilizzati nel mondo del web. Saper comprendere la differenza tra linguaggi di programmazione e linguaggi di markup. Conoscere la struttura di una pagina web. Utilizzare un linguaggio di scripting per rendere interattive le pagine web. Comprendere e analizzare le differenze fra sistema informativo e sistema informatico Saper modellare una realtà analizzando tutti gli aspetti ritenuti essenziali per una corretta applicazione di un appropriato livello di astrazione Saper riconoscere entità e attributi all'interno di una realtà di interesse Saper porre in relazione varie entità utilizzando le associazioni più opportune Saper impostare appropriati vincoli di integrità	(AL) Fondamenti di programmazione. ▪ I principi della OOP ▪ Modellare semplici classi ▪ Ereditarietà (DE) Ecosistema del web ▪ Introduzione e storia del WWW ▪ La progettazione di siti web ▪ Il linguaggio HTML (formattazione, testo, punti elenco, tabelle, immagini, form) ▪ Il collegamento tra le pagine (link) ▪ CMS ▪ I fogli di stile ▪ Il linguaggio CSS (selettori, carattere, colori, bordi, box model) ▪ Il design responsive ▪ Il linguaggio Javascript: caratteristiche generali ed elementi di base. (BD) Basi di dati ▪ Introduzione ai database ▪ Progettazione concettuale e logica ▪ Modello ER: entità e attributi ▪ Modello ER: identificativi univoci ▪ Modello ER: associazioni ▪ Documentazione di un modello ER ▪ Modello relazionale: relazione e chiavi ▪ Mapping da ER a relazionale ▪ Vincoli di integrità ▪ Normalizzazione ▪ Algebra relazionale

	Tradurre uno schema concettuale in uno schema relazionale Applicare consapevolmente gli operatori dell'algebra relazionale al fine di interrogare la base di dati Saper normalizzare schemi non normalizzati Riconoscere le varie istruzioni SQL in funzione delle operazioni da svolgere Saper impostare correttamente le istruzioni SQL per costruire query semplici e query annidate Saper far uso corretto di funzioni di aggregazione	▪ Caratteristiche di SQL ▪ DDL ▪ DML ▪ QL: la select ▪ JOIN (inner e outer join) ▪ Funzioni di aggregazione e clausola GROUP BY ▪ Interrogazioni annidate

DISCIPLINA : INFORMATICA – Classe QUINTA

COMPETENZE	ABILITÀ	CONOSCENZE
Avere una visione di insieme delle tecnologie e delle applicazioni nella trasmissione di dati sulle reti Comprendere le basi del calcolo numerico. Essere in grado di stimare la complessità computazionale in tempo di un semplice algoritmo. Saper confrontare gli algoritmi in base alle classi di complessità.	Saper spiegare i principi di base del funzionamento delle reti Riconoscere le architetture e i dispositivi di rete Individuare i diversi livelli nei modelli ISO/OSI e TCP/IP Scrivere programmi relativi ai principali algoritmi di calcolo numerico. Applicare i principi della computazione a problemi di calcolo. Scrivere programmi di simulazione in relazione allo studio e alle indagini di varie discipline.	(RC) Reti di computer ▪ La comunicazione fra computer ▪ Gli elementi fondamentali di una rete ▪ Componenti hardware: dispositivi e mezzi fisici ▪ Criteri per la classificazione delle reti: estensione, architettura, topologia ▪ I protocolli di comunicazione ▪ Il modello di riferimento OSI ▪ Le LAN ▪ Il livello fisico: i mezzi trasmissivi e la codifica di linea ▪ Il livello datalink: sottolivelli LLC e MAC ▪ Le LAN wireless (IS) Struttura dell'Internet e dei servizi di rete. ▪ Le origini di Internet ▪ La suite di protocolli TCP/IP ▪ Lo strato Internet del TCP/IP ▪ Gli indirizzi IP ▪ L'accesso remoto a Internet ▪ I protocolli del livello di trasporto: TCP e UDP ▪ Le applicazioni di rete ▪ Il protocollo HTTP, FTP, la posta elettronica ▪ Il DNS

		(CS) Computazione, calcolo numerico e simulazione ▪ La qualità e la complessità degli algoritmi ▪ I numeri macchina ▪ Gli errori e l'attendibilità dei risultati ▪ I principali algoritmi del calcolo numerico. (CS) Introduzione all'intelligenza artificiale ▪ Breve storia dell'intelligenza artificiale ▪ I sistemi esperti ▪ L'importanza e la quantità dei dati (Big Data) ▪ Il machine learning ▪ Gli alberi decisionali (Decision Tree)

LICEO SCIENTIFICO FULCIERI PAULUCCI DI CALBOLI – FORLÌ

GRIGLIA DI VALUTAZIONE PROVA SCRITTA

INDICATORI	LIVELLI DI VALUTAZIONE	PUNTEGGIO
Conoscenze	Individua e applica le regole più efficaci alla soluzione	4
	Esprime discreta conoscenza di regole e principi	3
	Coglie in modo sostanziale le regole e i principi	**2**
	Coglie solo in parte le regole e i principi	1,5
	Non individua o non applica regole e principi	1
Correttezza e completezza dei procedimenti **Capacità logiche e argomentative nella scelta della strategia risolutiva ottimale**	Affronta con padronanza e originalità lo sviluppo della tematica proposta, che risolve in tutte le sue parti	3,5
	Svolge in modo corretto e completo il tema proposto	3
	Svolge l'elaborato in modo sostanzialmente corretto	**2,5**
	Svolge l'elaborato in modo essenziale ma con imprecisioni o mancanze	1
	Commette gravi errori nell'esecuzione delle operazioni richieste	0,5
Uso del formalismo e del linguaggio specifico	Padroneggia il linguaggio e la simbologia	2,5
	Utilizza in modo appropriato il linguaggio e la simbologia	2
	Utilizza in modo sostanzialmente corretto il linguaggio e la simbologia	**1,5**
	Utilizza in modo generico il linguaggio e la simbologia	1
	Utilizza in modo inadeguato il linguaggio e la simbologia	0,5
PUNTEGGIO COMPLESSIVO		

COGNOME E NOME:

CLASSE …….. DATA …….. VOTO ……

GRIGLIA DI VALUTAZIONE PROVA ORALE

INDICATORI	LIVELLI DI VALUTAZIONE	PUNTEGGIO
		(Tot. 10)
Conoscenze	Complete e approfondite	5
	Precise	4
	Essenziali	**3**
	Parziali	**2**
	Lacunose	1
	Mancanti	0,5
Esposizione	Espressione curata, linguaggio specifico preciso	2
	Espressione sicura, lessico corretto	1,75
	Discorso e lessico sostanzialmente corretti	**1,5**
	Discorso incerto, lessico non sempre corretto	1
	Lessico del tutto inadeguato	0,5
Organizzazione delle strategie risolutive e capacità di creare collegamenti	Organizzazione e applicazione autonoma delle conoscenze acquisite	3
	Organizzazione completa delle conoscenze acquisite	2,5
	Organizzazione essenziale dei contenuti	**1,5**
	Limitata organizzazione dei contenuti	1
	Carente organizzazione e mancata applicazione dei contenuti	0,5
PUNTEGGIO COMPLESSIVO		

COGNOME E NOME:

CLASSE …….. **DATA …….** **VOTO ……**

PROGRAMMAZIONE DISCIPLINARE di INFORMATICA

per OBIETTIVI MINIMI

Classe PRIMA

- Conoscere la differenza tra hardware e software
- Conoscere le caratteristiche architetturali di un computer
- Saper rappresentare i dati con i numeri binari
- Saper effettuare conversioni tra sistema decimale e sistema binario
- Conoscere la classificazione dei software
- Conoscere le principali funzioni del Sistema Operativo
- Conoscere la struttura di un foglio di testo e le sue funzioni base
- Conoscere la struttura di un foglio di calcolo e le sue funzioni base
- Conoscere la struttura di una presentazione multimediale e le sue funzioni base
- Acquisire la definizione e le caratteristiche di un algoritmo
- Conoscere i fondamenti delle programmazione strutturata (teorema di Bohm-Jacopini)
- Saper realizzare il diagramma a blocchi di un algoritmo che risolva semplici problemi usando le strutture di controllo di sequenza, selezione e iterazione

Classe SECONDA

- Acquisire il concetto di linguaggio di programmazione
- Conoscere la differenza tra linguaggi di alto livello e basso livello
- Comprendere la differenza tra compilatore e interprete
- Comprendere il concetto di variabile
- Conoscere le istruzioni di comunicazione con l'utente
- Editare e testare un programma in C
- Saper utilizzare variabili intere, reali e bool
- Saper utilizzare gli operatori / e % sugli interi
- Conoscere l'istruzione di selezione semplice e doppia
- Conoscere e saper utilizzare gli operatori logici
- Saper scrivere codice con blocchi di selezione annidati
- Conoscere le diverse strutture di iterazione (precondizionale, postcondizionale, enumerativa) e saperle codificare in C
- Saper scrivere programmi con iterazioni e selezioni

- Saper scrivere programmi con iterazioni annidate
- Generare numeri casuali
- Conoscere le caratteristiche degli array monodimensionali (vettori)
- Saper individuare i problemi che necessitano dell'utilizzo di vettori
- Sapere scrivere, leggere, ricercare, manipolare dati in un vettore

Classe TERZA

- Conoscere le caratteristiche del linguaggio Python.
- Scrivere programmi utilizzando le strutture di controllo.
- Saper elaborare testo utilizzando il tipo string.
- Saper codificare programmi che utilizzano la struttura dati lista.
- Saper riconoscere la struttura dati adeguata allo scopo del programma.
- Saper scomporre un problema in sottoproblemi e risolverlo attraverso le funzioni
- Implementare funzioni in Python
- Saper gestire il passaggio dei parametri
- Saper riconoscere problemi che necessitano di ricerca e di ordinamento.
- Saper implementare gli opportuni algoritmi.
- Saper progettare e codificare classi seguendo i principi della programmazione ad oggetti.
- Saper utilizzare le classi all'interno di programmi.
- Saper leggere e scrivere file.
- Saper organizzare dati frutto di elaborazione di file in strutture dati.

Classe QUARTA

- Costruire oggetti software utilizzando la metodologia OOP
- Comprendere il meccanismo dell'ereditarietà.
- Integrare le conoscenze pregresse con la OOP.
- Conoscere la storia del web e l'evoluzione fino ad oggi.
- Conoscere i principali linguaggi utilizzati nel mondo del web.
- Saper comprendere la differenza tra linguaggi di programmazione e linguaggi di markup.
- Conoscere la struttura di una pagina web.
- Comprendere la necessità dei database

- Conoscere i vantaggi di un DBMS
- Saper individuare le entità e le relazioni tra le entità all'interno di una situazione complessa
- Saper progettare lo schema concettuale dei dati E-R
- Conoscere il modello logico relazionale dei dati
- Saper effettuare il mapping da modello concettuale a modello logico relazionale
- Conoscere le regole di integrità
- Conoscere le caratteristiche del linguaggio SQL (DDL, DML e QL)
- Saper interrogare un database mediante query semplici e su più tabelle (JOIN)

INDICAZIONI METODOLOGICHE PER le VERIFICHE FINALI DEI CORSI / SPORTELLI di recupero e sostegno al termine del trimestre

Prova/e : - indicare la durata e la tipologia

Una prova scritta della durata di 1 ora, contenente brevi esercizi di lettura e/o completamento di codice, la richiesta di qualche definizione e almeno un esercizio di progettazione di codice, dato un problema assegnato

INDICAZIONI METODOLOGICHE PER le VERIFICHE degli alunni con sospensione di giudizio

Prova/e : - indicare la durata e la tipologia

Una prova scritta della durata di 1 ora, contenente brevi esercizi di lettura e/o completamento di codice, e almeno un esercizio di progettazione di codice, dato un problema assegnato

Una prova orale della durata di 15 min, volta a discutere l'elaborato scritto e a verificare le conoscenze teoriche legate agli argomenti oggetto della prova scritta

PROGETTI DI ARRICCHIMENTO O DI AMPLIAMENTO DELL'OFFERTA FORMATIVA STRETTAMENTE CONNESSI ALLA DISCIPLINA

1. **Robotica (Arduino):** il progetto si articola in diverse attività che vengono portate avanti in parallelo da gruppi di alunni differenti. Fra queste vengono solitamente proposte:
    a. le Olimpiadi Robotiche, organizzate da MakersLab Forlì, in cui è richiesto di assemblare uno o più robot per svolgere competizioni di Line Follower, Robo-Calcio, Mini-Sumo, Robo-Labirinto.
    b. le Campionati nazionali di robotica, organizzate dal MIUR, richiedono la progettazione e la realizzazione di un sistema robotizzato ed interagente con gli esseri umani secondo specifiche ed un tema forniti ogni anno dagli organizzatori.

2. **Olimpiadi di Informatica:** il progetto si pone come obiettivo la valorizzazione delle eccellenze e l'approfondimento degli aspetti algoritmici della disciplina di informatica. Segue le indicazioni e le

attività definite a livello nazionale da un accordo tra MIUR ed AICA (Associazione Italiana per l'Informatica ed il Calcolo Automatico. Vengono proposte due diverse competizioni:

a. Gara a squadre
b. Olimpiadi individuali

3. **Olimpiadi di Problem Solving:** il progetto si pone come obiettivo promuovere la diffusione del Pensiero Computazionale tramite attività coinvolgenti che si applicano alle diverse discipline scolastiche. Le gare sono aperte ai soli studenti del primo biennio.