SYLLABUS TRIENNIO

Anno scolastico:	**2014 – 2015**	**Classe:**	**4**	**Sez.:**	**G**
Disciplina:	**INFORMATICA**				
Articolazione:	**Informatica**				
Docente:	**prof. Mauro Malatesta**				
	(firma)				

Approvata dal Dipartimento di _____ il	__/__/_____
Approvata dal Consiglio di classe il ……....	__/__/_____

Obiettivi del corso [1]

Il docente di "Informatica" concorre a far conseguire allo studente, al termine del percorso quinquennale, i seguenti risultati di apprendimento relativi al profilo educativo, culturale e professionale: utilizzare, in contesti di ricerca applicata, procedure e tecniche per trovare soluzioni innovative e migliorative, in relazione ai campi di propria competenza; cogliere l'importanza dell'orientamento al risultato, del lavoro per obiettivi e della necessità di assumere responsabilità nel rispetto dell'etica e della deontologia professionale; orientarsi nella normativa che disciplina i processi produttivi del settore di riferimento, con particolare attenzione sia alla sicurezza sui luoghi di vita e di lavoro sia alla tutela dell'ambiente e del territorio; intervenire nelle diverse fasi e livelli del processo produttivo, dall'ideazione alla realizzazione del prodotto, per la parte di propria competenza, utilizzando gli strumenti di progettazione, documentazione e controllo; riconoscere gli aspetti di efficacia, efficienza e qualità nella propria attività lavorativa; utilizzare modelli appropriati per investigare su fenomeni e interpretare dati sperimentali; utilizzare le reti e gli strumenti informatici nelle attività di studio, ricerca ed approfondimento disciplinare.

I risultati di apprendimento sopra riportati, in esito al percorso quinquennale, costituiscono il riferimento delle attività didattiche della disciplina nel secondo biennio e quinto anno. La disciplina, nell'ambito della programmazione del Consiglio di classe, concorre in particolare al raggiungimento dei seguenti risultati di apprendimento, relativi all'indirizzo, espressi in termini di competenze:

- utilizzare le strategie del pensiero razionale negli aspetti dialettici ed algoritmici per affrontare situazioni problematiche elaborando opportune soluzioni
- sviluppare applicazioni informatiche per reti locali o servizi a distanza
- scegliere dispositivi e strumenti in base alle loro caratteristiche funzionali
- gestire progetti secondo le procedure e gli standard previsti dai sistemi aziendali di gestione della qualità e della sicurezza
- redigere relazioni tecniche e documentare le attività individuali e di gruppo relative a situazioni professionali

L'articolazione dell'insegnamento di "Informatica" in conoscenze e abilità è di seguito indicata quale orientamento per la progettazione didattica del docente in relazione alle scelte compiute nell'ambito della programmazione collegiale del Consiglio di classe

MODULO 1[2]

TITOLO[3]	**Programmazione ad oggetti**
DESCRIZIONE[4]	Il Modulo 1 introduce la programmazione orientata agli oggetti. Questo paradigma include la programmazione procedurale e ne estende le potenzialità. Si introduce il concetto di oggetto e di classe e si descrive l'implementazione di semplici classi. Si discute il ciclo di vita di un oggetto e le modalità di associazione degli attributi e dei metodi con gli indirizzi di memoria (binding). Si descrivono esempi di oggetti passati come parametri a metodi e oggetti restituiti come valori di ritorno.di metodi. Si illustrano in dettaglio le proprietà di incapsulamento, di ereditarietà (accennando ai vari tipi) e di polimorfismo (sotto forma di overloading e di overriding) e per ciascuna si propongono esempi applicativi.
OBIETTIVI[5]	• Paradigma ad oggetti • Oggetti reali e oggetti software • Attributi e metodi • Il concetto di classe • I metodi di default • Comunicazione tra oggetti • Ciclo di vita di un oggetto • Binding statico e dinamico • Incapsulamento • Ereditarietà • Polimorfismo • Variabili di classe • Metodi di classe • Oggetti come parametri • Oggetti come valori di ritorno • Gerarchia di classi • Relazioni in una gerarchia • Tipi di ereditarietà • Casting tra classi • Il polimorfismo • Overloading • Overriding • Classi astratte

UdA[6]	Titolo[7]	Tema[8]	Argomento[9]
T1A1	**Gli oggetti reali**	1.-Il paradigma procedurale	1.1-Descrivere l'astrazione sui dati e il suo scopo 1.2-Descrivere l'astrazione sulle procedure e il suo scopo 1.3-Richiamare le fasi di sviluppo della programmazione procedurale 1.4-Riconoscere il paradigma procedurale come orientato all'algoritmo
		2.-Il paradigma Object Oriented	2.1-Descrivere un oggetto come insieme di dati e operazioni 2.2-Descrivere il paradigma OO come orientato ai dati 2.3-Descrivere il compito della OOP come simulazione di oggetti reali 2.4-Riconoscere la necessità di un linguaggio OO 2.5-Descrivere la OOP come comunicazione tra oggetti 2.6-Descrivere la caratteristica di riusabilità degli oggetti
		3.-Gli oggetti reali	3.1-Descrivere oggetti reali in termini di loro proprietà 3.2-Definire lo stato di un oggetto 3.3-Descrivere gli oggetti reali in termini di comportamento 3.4-Descrivere il cambiamento di stato di un oggetto 3.5-Saper interpretare i messaggi fra oggetti reali 3.6-Riconoscere la presenza di un'interfaccia negli oggetti reali
T1A2	**Gli oggetti software**	1.-Simulare oggetti reali	1.1-Descrivere il compito della OOP 1.2-Saper simulare mediante la OOP un oggetto reale mediante attributi e metodi 1.3-Interpretare via software lo stato di un oggetto reale
		2.-Oggetit software	2.1-Fare esempi di oggetti software, descrivendone attributi

			e metodi 2.2-Descrivere gli attributi mediante nome, tipo e contenuto 2.3-Riconoscere l'importanza della privatezza degli attributi 2.4-Descrivere i metodi mediante funzioni con parametri 2.5-Riconoscere l'importanza della privatezza dei metodii 2.6-Conoscere gli strumenti sintattici della UML 2.7-Disegnare il diagramma di semplici oggetti software
T1B1	**Il concetto di classe**	1.-Classi e oggetti	1.1-Descrivere il significato e lo scopo di una classe 1.2-Saper fare esempi di oggetti della stessa classe 1.3-Distinguere i concetti di classe e oggetto 1.4-Descrivere lo scopo dell'interfaccia di una classe 1.5-Descrivere lo scopo dell'implementazione di una classe
		2.-Implementazione di classi	2.1-Implementare semplici classi di uso comune 2.2-Riconoscere l'importanza di distinguere membri privati e pubblici 2.3-Saper rappresentare una classe mediante UML
		3.-I metodi di default	3.1-Elencare il nome e lo scopo dei metodi di default 3.2-Saper implementare un costruttore con parametri 3.3-Saper implementare un costruttore senza parametri 3.4-Saper riconoscere la presenza di costruttori in overload 3.5-Conoscere e saper implementare metodi accessori 3.6-Conoscere e saper implementare metodi modificatori
T1B2	**Significato e propr. della OOP**	1.-Comunicazionetra oggetti	1.1-Saper interpretare la OOP come messaggi fra oggetti 1.2-Conoscere gli elementi della comunicazione 1.3-Individuare oggetto mittente e oggetto destinatario 1.4-Individuare messaggio e contenuto 1.5-Saper rappresentare una comunicazione tra oggetti mediante il diagramma di sequenza 1.6-La classe **Math** e i suoi metodi. 1.7-Esempi di comunicazione con la classe **Math**
		2.-Ciclo di vita di un oggetto	2.1-Conoscere il ciclo di vita di un oggetto 2.2-Sapere cosa significa creare un oggetto 2.3-Saper individuare e creare un oggetto 2.3-Sapere cosa vuol dire che un oggetto "muore" 2.4-Conoscere i compiti e il significato di *garbage collector*
		3.-Gloggetti e la memoria	3.1-Definire il concetto di binding statico e fare esempi 3 2-Definire il concetto di binding dinamico e fare esempi 3.3-Distinguere criticamente i due tipi di binding
		4.- Le proprietà della OOP	4.1-Elencare le proprietà della OOP 4.2-Descrivere l'incapsulamento e i suoi vantaggi 4.3-Descrivere l'ereditarietà e i suoi vantaggi 4.4-Descrivere il polimorfismo e i suoi vantaggi
T1C1	**Incapsulamento e tecniche OOP**	1.-L'incapsulamento	1.1-Descrivere i vantaggi dell'incapsulamento in termini idi protezione dei dati 1.2-Descrivere i vantaggi dell'incapsulamento in termini idi astrazione sui dati e sule procedure 1.3-Descrivere i vantaggi dell'incapsulamento in termini idi mascheramento dell'informazione

			1.4-Descrivere i vantaggi dell'incapsulamento in termini idi indipendenza degli oggetti 1.5-Descrivere i vantaggi dell'incapsulamento in termini idi riutilizzo del codice
		2.-Le variabili di classe	2.1-Conoscere i vari tipi di variabile (locali, istanza e di classe) 2.2-Descrivere l'utilità delle variabili di classe 2.3-Saper usare correttamente le variabili di classe
		3.-I metodi di classe	3.1-Descrivere l'utilità dei metodi di classe 3.2-Saper implementare correttamente i metodi di classe 3.3-Saper istanziare correttamente i metodi di classe
		4.-Gli oggetti nei metodi	4.1-Saper passare oggetti come parametri di metodi 4.2-Saper ricavare oggetti come valori di ritorno di metodi
		5.-La tecnica di sviluppo OO	5.1-Descrivere la metodologia di sviluppo OO 5.2-Conoscere e saper sviluppare la fase di OOA 5.3-Conoscere e saper sviluppare la fase di OOD
T1C2	**Ereditarietà e polimorfismo**	1.-L'ereditarietà	1.1-Descrivere i vantaggi dell'ereditarietà in termini di protezione e astrazione sui dati 1.2-Descrivere i vantaggi dell'ereditarietà in termini di riutilizzo del software 1.3-Riconoscere ed usare l'ereditarietà per espansione e per ridefinizione 1.4-Conoscere e saper usare lo specificatore **protected**
		2.-Gerarchia di classi	2.1-Saper interpretare e costruire una semplice gerarchia di classi 2.2-Conoscere l'effetto e saper usare i metodi di default nelle sottoclassi 2.3-Distinguere e usare le relazioni di generalizzazione e specializzazione 2.4-Distinguere e saper usare le relazioni di aggregazione e composizione
		3.-I tipi di ereditarietà	3.1.-Distinguere e saper usare ereditarietà pubblica e provata 3.2- Conoscere e saper distinguere ereditarietà singola e multipla
		4.-Il casting tra classi	4.1-Riconoscere e saper usare l'upcsasting 4.2-Riconoscere e saper usare il downcasting
		5.-Il polimorfismo	5.1-Conoscere le forme di overloading e di overriding 5.2-Saper implementare ed usare metodi in overloading 5.3-Saper implementare ed usare metodi in overriding
		6.-Le classi astratte	6.1-Definire cosa si intende per classe astratte 6.2-Esporre i vantaggi di creare classi astratte 6.3-Definire il concetto di metodo astratto 6.4-Esporre i vantaggi dell'uso di metodi astratti