

a.s. 2019 / 2020

PROGRAMMAZIONE PER COMPETENZE

**2° BIENNIO
CLASSI 3A - 3B - 3C**

Dipartimento : INFORMATICA E TELECOMUNICAZIONI
MATERIA : **INFORMATICA**

Docenti: Bartolacci M., Braccacini A., Mantini S., Piersigilli F.
ITP: Alfieri V.

OBIETTIVI rispetto alle COMPETENZE di CITTADINANZA	COMPETENZE IN AMBITO DISCIPLINARE	ABILITA'	CONOSCENZE	VERIFICHE rispetto alle competenze di ambito disciplinare
Imparare ad imparare Organizzare l'apprendimento scegliendo ed utilizzando varie fonti e modalità Comunicare Comprendere messaggi di genere diverso e con supporti diversi Rappresentare concetti utilizzando linguaggi diversi (verbale, matematico, simbolico-grafico) Risolvere problemi	Cogliere i motivi del crescente sviluppo dei calcolatori, comprendendo il loro ruolo Capire la logica dei sistemi di elaborazione Capire le modalità di interazione tra i principali dispositivi hardware di un elaboratore	Padronanza della terminologia informatica Saper scegliere il linguaggio più appropriato in base al tipo di problema Saper riconoscere in un Pc i principali dispositivi hardware e i loro collegamenti	MODULO 0: PREREQUISITI PER COMINCIARE Integrazione e consolidamento delle conoscenze di base MODULO 1: POTENZIAMENTO: PROBLEMI E ALGORITMI	Test d'ingresso Prova orale <u>Mese di settembre</u>

Affrontare situazioni problematiche facendo ipotesi, raccogliendo dati e proponendo soluzioni	Utilizzare le tecniche e le strategie del pensiero razionale per affrontare problemi ed elaborare opportune soluzioni	Saper risolvere semplici problemi di logica Saper progettare algoritmi Analizzare e confrontare algoritmi diversi per la soluzione dello stesso problema	Esempi di analisi e scomposizione di problemi elementari – Metodologia Top-Down Programmazione strutturata: esempi con applicazione a problemi pratici usando le strutture di controllo fondamentali della programmazione: sequenza, selezione, iterazione Teorema di “Jacopini – Bohm” Classificazione dei linguaggi ad alto livello (imperativo, funzionale, non procedurali/dichiarativi) Definizione di traduttore, interprete, compilatore	Soluzione di problemi reali producendo un Flow-Chart (case studies) con AlgoBuild o strumenti equivalenti in laboratorio. Codifica di una semplice applicazione. <u>Mese di ottobre</u>
Risolvere problemi Affrontare situazioni problematiche facendo ipotesi, raccogliendo dati e proponendo soluzioni Progettare Elaborare e realizzare progetti utilizzando le conoscenze apprese,	Saper sviluppare semplici applicazioni su singolo PC	Saper implementare algoritmi con un moderno linguaggio di programmazione	MODULO 2: IL LINGUAGGIO DI PROGRAMMAZIONE C# Caratteristiche fondamentali del linguaggio: variabili e strutture di controllo in C#	Laboratorio: Realizzare semplici applicazioni in C# in modalità console e modalità grafica <u>Mese di novembre</u>

verificando i risultati raggiunti				
Risolvere problemi Affrontare situazioni problematiche facendo ipotesi, raccogliendo dati e proponendo soluzioni Progettare Elaborare e realizzare progetti utilizzando le conoscenze apprese, verificando i risultati raggiunti	Saper applicare le tecniche per sviluppare semplici applicazioni Windows.	Saper implementare algoritmi che prevedano un'interfaccia grafica	MODULO 3: APPLICAZIONI WINDOWS FORM Introduzione alle applicazioni Windows Form Elementi di base di una interfaccia grafica: Form, Label, TextBox, Button Visualizzazione dei messaggi: uso del MessageBox Uso di più Form. Migliorare la comunicazione con l'utente: ListBox, ComboBox, RadioButton, CheckBox, PictureBox Controlli sull'input dei dati: Try- Catch	Realizzare applicazioni semplici che prevedano l'utilizzo degli elementi principali di una interfaccia grafica e l'accesso alla memoria di massa <u>Mese novembre-dicembre</u>
Risolvere problemi Affrontare situazioni problematiche facendo ipotesi, raccogliendo dati e proponendo soluzioni Progettare Elaborare e realizzare progetti utilizzando le conoscenze apprese,	Saper affrontare e risolvere problematiche di media complessità	Progettare ed implementare applicazioni secondo il paradigma studiato	MODULO 4: LINGUAGGIO C#: SOTTOPROGRAMMI E CARATTERISTICHE AVANZATE I sottoprogrammi in C#: il concetto di "metodo" (loro struttura ed utilizzo)	Realizzare applicazioni in C# e produrre relativa documentazione <u>Mese di dicembre e gennaio</u>

verificando i risultati raggiunti			Il passaggio dei parametri (passaggio per valore e per riferimento) Realizzazione di applicazioni strutturate con sottoprogrammi La ricorsione: concetti generali; applicazione a problemi matematici	
Risolvere problemi Affrontare situazioni problematiche facendo ipotesi, raccogliendo dati e proponendo soluzioni Progettare Elaborare e realizzare progetti utilizzando le conoscenze apprese, verificando i risultati raggiunti	Saper applicare metodologie standard per la soluzioni di problemi comuni	Saper implementare gli algoritmi classici adattandoli allo specifico problema	MODULO 4: LE STRUTTURE DATI SEMPLICI Le collezioni di dati: le strutture vettoriali (array monodimensionali) in C#; esempi di definizione ed uso. Il concetto di stringa; utilizzo di var. di tipo stringa Implementazione di algoritmi classici su vettori: ricerca, ricerca del max, scansione ... Algoritmi di Ordinamento (varie tecniche: bubble-sort, selezione) Algoritmo di ricerca binaria o dicotomica. Saper gestire gli eventi e le eccezioni.	Realizzare applicazioni in C# e produrre relativa documentazione <u>Mese gennaio/febbraio</u>

			Conoscere i controlli avanzati: DataGrid, TabControl, finestre di dialogo. Creare controlli a Run-Time.	
Risolvere problemi Affrontare situazioni problematiche facendo ipotesi, raccogliendo dati e proponendo soluzioni Progettare Elaborare e realizzare progetti utilizzando le conoscenze apprese, verificando i risultati raggiunti	Saper applicare le tecniche per sviluppare applicazioni complesse che prevedano la memorizzazione ed il recupero di dati strutturati	Saper implementare algoritmi che interagiscano con il "File System"	MODULO 5: STRUTTURE DATI COMPLESSE e FILE Strutture tipo matrici Definizione di file binari e di testo Accesso sequenziale a strutture su memoria di massa. Utilizzo dei file di tipo testo	Realizzare applicazioni complesse che prevedano l'accesso alla memoria di massa <u>Mese di marzo</u>
Risolvere problemi Affrontare situazioni problematiche facendo ipotesi, raccogliendo dati e proponendo soluzioni Progettare Elaborare e realizzare progetti utilizzando le conoscenze apprese, verificando i risultati raggiunti	Sapere utilizzare la programmazione orientata agli oggetti (OOP): - saper progettare semplici programmi usando il paradigma OOP con UML - saper implementare programmi attraverso paradigmi OOP	Scrivere programmi utilizzando classi ed oggetti, dopo aver definito semplici classi ed oggetti	MODULO 7: PROGRAMMAZIONE OOP Progettazione di Classi e Oggetti in UML. Definizione di classi e oggetti in C#.	Schematizzazione di un progetto con UML, individuando le classi, gli attributi e i metodi <u>Mese maggio</u>

Jesi 05/10/2019

Docenti: Bartolacci M., Mantini S., Piersigilli F.

ITP: Alfieri V. Braccacini A