|  |
| --- |
| ***Premessa:*** |
| La disciplina “Informatica” concorre a far conseguire allo studente al termine del percorso quinquennale i seguenti risultati di apprendimento relativi al profilo educativo, culturale e professionale dello studente **(PECUP)**:   * 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.   **Competenze Disciplinari**   * 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.   La metodologia utilizzata sarà strettamente connessa all'individuazione degli obiettivi e dei contenuti, di particolare importanza è la scelta delle strategie operative e dei mezzi più idonei alla realizzazione del processo di apprendimento da parte degli alunni. In tale ottica, gli strumenti della mediazione didattica terranno conto del contesto e assicureranno la sostenibilità delle attività proposte e un generale livello di inclusività.  Per quanto non espressamente esplicitato si fa riferimento ai Verbali di dipartimento ratificati dal Collegio dei Docenti per il corrente anno scolastico (standard di valutazione, verifica trasversale per “Assi culturali” e classi parallele, attività interdisciplinari, visite, viaggi, sopralluoghi aziendali, attività formative extracurricolari, P.T.O.F.) |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Modulo 1: il linguaggio Java** | | | | **Periodo: Settembre - Dicembre** | | |
| **Competenze disciplinari** | **Livelli di apprendimento** | **Conoscenze** | **Abilità** | | **Indicazioni**  **Metodologiche** | **Attività e Verifiche** |
| Sviluppare i problemi codificando l’algoritmo risolutivo con un linguaggio di programmazione.  Costruire programmi eseguibili dal computer e controllare l’esecuzione del programma. | **A - Livello di apprendimento avanzato** | Struttura generale di un programma in linguaggio Java.  La sintassi del linguaggio Java.  Codifica di programmi in linguaggio Java utilizzando le strutture di controllo e le strutture dati: vettori, vettori paralleli, vettori di record, matrici. | Saper scrivere programmi utilizzando in modo corretto la sintassi del linguaggio Java.  Codificare e validare gli algoritmi. | | Lezione dialogata.  Problem solving e lavoro individuale e/o di gruppo.  Esercitazioni individuali e/o di gruppo. | Interrogazione individuale.  Verifica formativa.  Test online.  Prove pratiche. |
| **B - Livello di apprendimento intermedio** | Struttura generale di un programma in linguaggio Java.  La sintassi del linguaggio Java.  Codifica di programmi in linguaggio Java utilizzando le principali strutture di controllo e le strutture dati: vettori, vettori di record, matrici. | Saper scrivere semplici programmi utilizzando in modo corretto la sintassi del linguaggio Java.  Codificare e validare gli algoritmi. | |
| **C - Livello di apprendimento base** | Struttura generale di un programma in linguaggio Java.  La sintassi del linguaggio Java.  Codifica di semplici programmi in linguaggio Java utilizzando semplici strutture di controllo e le strutture dati: vettori, matrici. | Saper scrivere semplici programmi utilizzando in modo corretto la sintassi del linguaggio Java. | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Modulo 2: IL PARADIGMA AD OGGETTI** | | | | **Periodo: Ottobre – Febbraio** | | |
| **Competenze disciplinari** | **Livelli di apprendimento** | **Conoscenze** | **Abilità** | | **Indicazioni**  **Metodologiche** | **Attività e Verifiche** |
| Progettare ed implementare algoritmi utilizzando il paradigma di programmazione OOP e diverse strutture di dati. | **A - Livello di apprendimento avanzato** | Programmazione a oggetti: concetti generali.  Le classi, gli oggetti e i metodi.  Rappresentazione in UML.  Visibilità dei membri.  La classe vettore e vettori di oggetti.  Ereditarietà.  Polimorfismo e overloading.  Riscrittura di metodi nelle classi derivate.  Elementi di C++/Java funzionali agli argomenti precedenti. | Applicare il concetto di astrazione per modellare le classi.  Usare la OOP per scrivere programmi complessi.  Individuare la specializzazione e la generalizzazione di una classe.  Applicare i concetti di ereditarietà e polimorfismo.  Definire gerarchie di classi.  Scrivere programmi utilizzando in modo corretto la sintassi del linguaggio C++/Java relativa alla OOP. | | Lezione dialogata.  Problem solving e lavoro individuale e/o di gruppo.  Esercitazioni individuali e/o di gruppo. | Interrogazione individuale.  Verifica formativa.  Test online.  Prove pratiche. |
| **B - Livello di apprendimento intermedio** | Programmazione a oggetti: concetti generali.  Le classi, gli oggetti e i metodi.  Rappresentazione in UML.  Visibilità dei membri.  La classe vettore.  Ereditarietà.  Elementi di C++/Java funzionali agli argomenti precedenti. | Applicare il concetto di astrazione per modellare le classi.  Usare la OOP per scrivere programmi.  Individuare la specializzazione e la generalizzazione di una classe.  Scrivere programmi utilizzando in modo corretto la sintassi del linguaggio C++/Java relativa alla OOP. | |
| **C - Livello di apprendimento base** | Programmazione a oggetti: concetti generali.  Le classi, gli oggetti e i metodi.  Rappresentazione in UML.  Visibilità dei membri.  La classe vettore.  Elementi di C++/Java funzionali agli argomenti precedenti. | Saper modellare una semplice classe. Scrivere semplici programmi utilizzando in modo corretto la sintassi del linguaggio C++/Java relativa alla OOP. | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Modulo 3: STRUTTURE DATI DINAMICHE** | | | | **Periodo: Marzo - Aprile** | | |
| **Competenze disciplinari** | **Livelli di apprendimento** | **Conoscenze** | **Abilità** | | **Indicazioni**  **Metodologiche** | **Attività e Verifiche** |
| Progettare ed implementare algoritmi utilizzando il paradigma di programmazione OOP e diverse strutture di dati. | **A - Livello di apprendimento avanzato** | Le variabili dinamiche.  La gestione dinamica della memoria.  Le liste concatenate: definizione, inserimento di un nodo, cancellazione di un nodo, ricerca di un elemento.  Vari tipi di liste lineari.  Strutture dati ad accesso limitato: pile e  code. | Saper definire dinamicamente le variabili.  Saper operare con i puntatori.  Acquisire le tecniche di implementazione delle liste lineari.  Implementare i vari tipi di liste lineari.  Riconoscere le situazioni adeguate all’utilizzo di liste lineari.  Saper scrivere le primitive di gestione di una pila e di una coda.  Riconoscere le situazioni di utilizzo di pile e code. | | Lezione dialogata.  Problem solving e lavoro individuale e/o di gruppo.  Esercitazioni individuali e/o di gruppo. | Interrogazione individuale.  Verifica formativa.  Test online.  Prove pratiche. |
| **B - Livello di apprendimento intermedio** | Le variabili dinamiche.  La gestione dinamica della memoria.  Le liste concatenate: definizione, inserimento di un nodo, cancellazione di un nodo, ricerca di un elemento.  Strutture dati ad accesso limitato: pile e  code. | Saper definire dinamicamente le variabili.  Saper operare con i puntatori.  Acquisire le tecniche di implementazione delle liste lineari.  Riconoscere le situazioni adeguate all’utilizzo di liste lineari.  Saper scrivere le primitive di gestione di una pila e di una coda.  Riconoscere le situazioni di utilizzo di pile e code. | |
| **C - Livello di apprendimento base** | Le variabili dinamiche.  La gestione dinamica della memoria.  Le liste concatenate: definizione e semplici operazioni.  Strutture dati ad accesso limitato: pile e  Code e semplici operazioni di inserimento e cancellazione di un nodo. | Saper definire dinamicamente le variabili.  Saper scrivere le primitive di gestione di semplici liste lineari.  Saper scrivere le primitive di gestione di una pila e di una coda. | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Modulo 4: I FILE** | | | | **Periodo: Maggio - Giugno** | | |
| **Competenze disciplinari** | **Livelli di apprendimento** | **Conoscenze** | **Abilità** | | **Indicazioni**  **Metodologiche** | **Attività e Verifiche** |
| Comprendere la possibilità di conservare le informazioni in memoria di massa.  Individuare la struttura dati più idonea a risolvere un determinato problema.  Sviluppare algoritmi che elaborano dati memorizzati sulle memorie di massa. | **A - Livello di apprendimento avanzato** | Gli archivi: organizzazione.  File di testo e binari: gestione, lettura, creazione, utilizzo per l'I/O.  Operazioni sui file.  File di testo in C++: definizione, apertura e chiusura di file.  File binari in C++: definizione, apertura e chiusura di file.  Operazioni di ricerca dati in un file sequenziale e/o random. | Scrivere, leggere e ricercare dati da un file.  Effettuare l’accesso diretto ai dati.  Creare archivi mediante file di record.  Progettare applicazioni che utilizzano file. | | Lezione dialogata.  Problem solving e lavoro individuale e/o di gruppo.  Esercitazioni individuali e/o di gruppo. | Interrogazione individuale.  Verifica formativa.  Test online.  Prove pratiche. |
| **B - Livello di apprendimento intermedio** | Gli archivi: organizzazione.  File di testo e binari: gestione, lettura, creazione, utilizzo per l'I/O  Operazioni sui file.  File di testo in C++: definizione, apertura e chiusura di file.  File binari in C++: definizione, apertura e chiusura di file.  Operazioni di ricerca dati in un file sequenziale. | Scrivere, leggere e ricercare dati da un file.  Effettuare l’accesso diretto ai dati.  Creare archivi mediante file di record. | |
| **C - Livello di apprendimento base** | Gli archivi: organizzazione.  File di testo e binari: gestione, lettura, creazione, utilizzo per l'I/O  Operazioni sui file.  File di testo in C++: definizione, apertura e chiusura di file.  File binari in C++: definizione, apertura e chiusura di file. | Scrivere e leggere dati da un file. | |