

Olimpíada Informática Argentina
Categoría “Programación”

Temario orientativo por nivel

Aclaraciones:

- Los conocimientos que se listan, son **orientativos** para los alumnos, docentes, y entrenadores, y son **no excluyentes** de otros conocimientos que se podrían incorporar ocasionalmente en los problemas de los Certámenes.
- Estos conocimientos son acumulativos para los distintos niveles (Se considera que los temas de cada nivel incluyen los temas correspondientes a los niveles anteriores).
- El temario está pensado como una orientación general de los temas que **podrían** aparecer en un problema, incluso en los más difíciles. Por lo tanto:
 - **No se espera** que los participantes necesariamente conozcan y dominen **todos** los temas del temario.
 - Se espera que los problemas más sencillos requieran únicamente de los conocimientos más básicos.

Nivel 1

Teoría: Matemática	Teoría: Ciencias de la computación	Algunos problemas de ejemplo
<p>Aritmética y Geometría:</p> <ul style="list-style-type: none"> • Enteros, operaciones aritméticas, comparación • Sistemas de numeración, conversión entre ellos • Propiedades básicas de los enteros (signo, paridad, divisibilidad) • Aritmética modular básica: suma, resta, multiplicación • Fracciones, porcentajes • Números primos <p>Lógica básica:</p> <ul style="list-style-type: none"> • Entender enunciados matemáticos con “existe” y “para todo”, y su diferencia. • Conectivos lógicos (y, o, no, entonces). • Uso de razonamientos y deducciones lógicas 	<p>Programación:</p> <ul style="list-style-type: none"> • Sintaxis y semántica básicas de algún lenguaje permitido en OIA • Variables, tipos, expresiones y asignación • Entrada y salida sencilla (desde y hacia archivos y stdin / stdout) • Estructuras de control selectivas (if) • Estructuras de control repetitivas (while / for / repeat) • Funciones (subrutinas) y pasaje de parámetros en ambos sentidos • Descomposición de problemas (pensamiento top-down) <p>Estructuras de datos fundamentales:</p> <ul style="list-style-type: none"> • Tipos primitivos (booleanos, enteros con y sin signo, caracteres) • Arreglos unidimensionales • Cadenas y procesamiento básico de cadenas • Variables globales y locales <p>Estrategias algorítmicas:</p> <ul style="list-style-type: none"> • Iteración <p>Algoritmos:</p> <ul style="list-style-type: none"> • Algoritmos sencillos con enteros: <ul style="list-style-type: none"> • Cambio de base • Test de primalidad (buscar divisores hasta la raíz) • Factorización (buscando divisores) • Manipulación de arreglos: <ul style="list-style-type: none"> • Llenar • Desplazar • Rotar • Invertir • Mínimo/Máximo • Sumas parciales • Algoritmos sencillos con cadenas de texto <ul style="list-style-type: none"> • Búsqueda directa de una subcadena dada. • Invertir una cadena. • Convertir entre mayúsculas y minúsculas. • Procesamiento y búsqueda secuencial sobre listas de valores (arreglos). 	<ul style="list-style-type: none"> • Construir un arreglo con ciertos elementos especiales de un conjunto (Ej: El máximo, los números primos, los divisibles por 7 y por 11, etc) • Convertir números entre sistemas de numeración • Realizar operaciones con números y escribir sus resultados • Procesar un texto identificando cadenas que contienen ciertas subcadenas definidas • Procesar un texto identificando o corrigiendo palabras de acuerdo a un criterio definido (identificar palíndromos, mayúsculas/minúsculas, letras prohibidas, etc)

Nivel 2

Teoría: Matemática	Teoría: Ciencias de la computación	Algunos problemas de ejemplo
<p>Aritmética y Geometría:</p> <ul style="list-style-type: none"> • Recta, segmento, ángulo, triángulo, rectángulo, cuadrado, circunferencia • Teorema de pitágoras <p>Funciones, relaciones y conjuntos:</p> <ul style="list-style-type: none"> • Funciones (inyectivas, sobreyectivas, inversas, composición) • Relaciones (Propiedad de ser simétricas, transitivas) • Orden lexicográfico • Conjuntos (inclusión, complementos, disjuntos) <p>Lógica básica:</p> <ul style="list-style-type: none"> • Tablas de verdad <p>Técnicas de demostración:</p> <ul style="list-style-type: none"> • Nociones de implicación, recíproco, contrarrecíproco, negación, contradicción • Demostraciones directas, por contraejemplo, por absurdo • Inducción matemática • Definiciones matemáticas recursivas <p>Combinatoria básica:</p> <ul style="list-style-type: none"> • Argumentos de conteo (principio de suma y multiplicación, progresiones aritméticas y geométricas, números de Fibonacci) • Factorial, números combinatorios • Principio del palomar <p>Grafos y árboles:</p> <ul style="list-style-type: none"> • Árboles y sus propiedades básicas, árboles con raíz • Grafos no dirigidos (grados, camino, ciclo, conectividad) • Grafos dirigidos (grado de entrada, grado de salida, camino y ciclo dirigido) • Árboles generadores • Grafos con pesos, etiquetas, colores • Multigrafos, grafos con bucles 	<p>Estructuras de datos fundamentales:</p> <ul style="list-style-type: none"> • Arreglos multidimensionales • Registros / Tipos de Datos definidos por el usuario • Estructuras enlazadas (listas, árboles binarios, árboles con raíz, y similares estructuras naturalmente recursivas) • Estrategias de implementación de grafos y árboles <p>Recursión:</p> <ul style="list-style-type: none"> • Implementación de funciones matemáticas recursivas <p>Análisis de algoritmos:</p> <ul style="list-style-type: none"> • Complejidades estándar (constante, logarítmica, lineal, $N \lg N$, cuadrática, cúbica, exponencial) • Medición empírica de la eficiencia <p>Estrategias algorítmicas:</p> <ul style="list-style-type: none"> • Algoritmos de fuerza bruta / búsqueda exhaustiva • Algoritmos golosos <p>Algoritmos:</p> <ul style="list-style-type: none"> • Algoritmos sencillos con enteros: <ul style="list-style-type: none"> • Algoritmo de Euclides • Criba de Eratóstenes • Factorización (con criba) • Operaciones sencillas con enteros de precisión arbitraria (suma, resta, multiplicación cuadrática) • Manipulación de arreglos: <ul style="list-style-type: none"> • Redimensionar • Histogramas • Bucket / counting sort • Búsqueda binaria • Quicksort y Quickselect (k-ésimo elemento) • Heapsort y Mergesort • DFS y BFS • Recorrido Inorder, Preorder, Postorder • Aplicaciones de DFS (Orden topológico) • Componentes conexas • Algoritmo de Dijkstra • Árbol generador mínimo (Prim / Kruskal) <p>Estructuras de datos:</p> <ul style="list-style-type: none"> • Pilas y colas • Representación de grafos (listas de adyacencia, matriz de adyacencia) • Heap binario • Estructura de Union-Find (Listas de componentes + Arreglo de representantes) 	<ul style="list-style-type: none"> • Ordenar un conjunto de datos de acuerdo a un criterio • Encontrar el camino más corto en un mapa • Dado un mapa, encontrar el máximo número de ciudades que están conectadas entre sí por caminos • En una tabla de $N \times M$ letras, encontrar el conjunto máximo de letras iguales unidas entre sí • Dada una descripción de un sistema de archivos, reportar los directorios que contienen más cantidad de archivos, contando y sin contar subdirectorios, o cuántos son dichos archivos

Nivel 3

<p>Teoría: Matemática</p> <p>Aritmética y Geometría:</p> <ul style="list-style-type: none"> • Punto, vector, coordenadas en el plano • Polígono (vértices, lados, simple, convexo, interior, área) • Distancia euclídea <p>Funciones, relaciones y conjuntos:</p> <ul style="list-style-type: none"> • Relaciones (de equivalencia, de orden, de orden total) • Conjuntos (producto cartesiano, conjunto de partes) <p>Combinatoria básica:</p> <ul style="list-style-type: none"> • Principio de inclusión/exclusión <p>Grafos y árboles:</p> <ul style="list-style-type: none"> • Grafos no dirigidos (ciclo y camino euleriano, ciclo y camino hamiltoniano) • Grafos dirigidos (ciclo y camino euleriano dirigido, ciclo y camino hamiltoniano dirigido) • Grafos bipartitos • Grafos planares (propiedades básicas) 	<p>Teoría: Ciencias de la computación</p> <p>Estructuras de datos fundamentales:</p> <ul style="list-style-type: none"> • Uso elemental de números de punto flotante en cómputos estables • Representación de datos en memoria • Memoria Heap vs Stack • Implementación de fracciones para cálculos exactos <p>Análisis de algoritmos:</p> <ul style="list-style-type: none"> • Especificación, precondition, poscondición, correctitud, invariantes • Análisis asintótico de complejidad, y notación de O grande. <p>Estrategias algorítmicas:</p> <ul style="list-style-type: none"> • Divide y vencerás • Backtracking (recursivo y no recursivo), Branch-and-bound • Programación dinámica <p>Algoritmos:</p> <ul style="list-style-type: none"> • Exponenciación eficiente (especialmente con aritmética modular) • Aplicaciones de DFS (encontrar ciclo/camino euleriano) • Algoritmo de Bellman Ford, Floyd-Warshall • Matching máximo bipartito $O(VE)$ • Biconectividad en grafos no dirigidos (puentes, puntos de articulación) • Conectividad en grafos dirigidos (componentes fuertemente conexas) <p>Estructuras de datos:</p> <ul style="list-style-type: none"> • Árboles binarios balanceados estáticos (Fenwick Tree y Segment Tree) • Árbol binario de búsqueda balanceado (Treap - Splay Tree - AVL u otro) • Árbol binario de búsqueda aumentado • Consulta de LCA en tiempo logarítmico • Estructuras de datos persistentes • Composición de estructuras de datos (Ej: Fenwick Tree en 2 dimensiones) • Tries <p>Algoritmos avanzados:</p> <ul style="list-style-type: none"> • Nociones básicas de teoría de juegos combinatoria: posiciones perdedoras y ganadoras, minimax para juego óptimo • Algoritmos Online vs Offline. Algoritmos probabilísticos. <p>Algoritmos geométricos:</p> <ul style="list-style-type: none"> • Representación de puntos, vectores, rectas, segmentos • Detección de puntos alineados, vectores paralelos / perpendiculares, giros horarios / antihorarios • Intersección de dos rectas/segmentos • Área de polígono y verificación de “punto dentro o fuera de polígono” • Compresión de coordenadas • Algoritmos $O(N \lg N)$ para cápsula convexa • Método de Sweep Line <p>Teoría de lenguajes:</p> <ul style="list-style-type: none"> • Gramáticas sencillas en BNF 	<p>Algunos problemas de ejemplo</p> <ul style="list-style-type: none"> • Problema del viajante de comercio • Problema de la mochila • Problemas de planificación de tareas de manera óptima • Optimizar triangulación de polígonos convexos con algún criterio (ejemplo: mínima suma de perímetros al cuadrado) • Problema de la subsecuencia creciente más larga • Problema de la subsecuencia común más larga • Problema de las ocho reinas y similares en tableros • Problema del par de puntos más cercano entre N en el plano • Dada una posición de Ajedrez o similar juego de tablero, decidir si existe “mate en 2” o similares desafíos
--	--	--