UNIVERSIDAD NACIONAL DE CÓRDOBA
Facultad de Ciencias Exactas, Físicas y Naturales
República Argentina

Programa de

Informática

Código:

Carrera: *Todas las Ingenierías*
Escuela: *Todas las Escuelas de Ingeniería*
Departamento: *Computación*.

Plan: *2007*
Semestre: *Primero y Segundo*
Carácter: *Obligatoria*
Bloque: *Ciencias Básicas*

Puntos: *3,5*
Carga Horaria: *84*
Horas Semanales: *5,25*
Año: *Primero*

Objetivos: Al terminar el curso el alumno:

- Comprenderá los principios necesarios para generalizar las soluciones específicas de los problemas científicos y de ingeniería a modelos de simulación mediante herramientas informáticas basadas en los algoritmos matemáticos.
- Será capaz de analizar, representar y resolver los problemas científicos y de ingeniería en un lenguaje formal de programación por procedimientos.
- Adquirirá la habilidad para utilizar un lenguaje informático que le facilite la formulación, resolución e implementación de programas compilados directamente sobre la arquitectura del computador.

Programa Sintético:

1. Introducción a la Informática.
2. Introducción a la especificación de programas.
3. Estructuras de control.
4. Funciones definidas por el usuario.
5. Tipos de datos cadenas, listas y arreglos.
6. Estructuras de datos compuestos.
7. Entrada/salida de información

Programa Analítico: de foja 2 a foja 10.

Programa Combinado de Examen (si corresponde): de foja a foja .

Bibliografía: de foja *10* a foja *10*.

Correlativas Obligatorias: Matemática del Curso de Nivelación.
Correlativas Aconsejadas: Introducción a la Matemática.

Rige: *2007*

Aprobado HCD, Res.:
Fecha:

Modificado / Anulado / Sust. HCD Res.:
Fecha:

El Secretario Académico de la Facultad de Ciencias Exactas, Físicas y Naturales (UNC) certifica que el programa está aprobado por el (los) número(s) y fecha(s) que anteceden. Córdoba, / / .

Carece de validez sin la certificación de la Secretaría Académica:

PROGRAMA ANALÍTICO

LINEAMIENTOS GENERALES

Para darle un sentido histórico a los particulares recortes pedagógicos de la asignatura se debe comenzar por la primera asignatura que comenzó a dictarse a partir de 1976 bajo la denominación de Computación y Cálculo Numérico, con un Programa basado en los conceptos de los Métodos Numéricos (McCracken y Dorn, 1976) y la implementación de los correspondientes algoritmos (Rice y Rice, 1973) en el lenguaje de programación científica denominado FORTRAN IV (Mc Cracken, 1967). Este programa se mantuvo, salvo por la forma de dictado que desde 1981, se dividió en teóricos de Cálculo Numérico y prácticas de Laboratorio de Programación en el Centro de Cálculo de la U.N.C., hasta la modificación integral de los planes de estudio ocurrida en 1988, a partir de lo cual se divide en dos materias. Computación ubicada en el primer año se ocupa de los conceptos básicos de la Informática y en particular de Programación de Computadoras y Métodos Numéricos en tercer año de los respectivos temas de aplicación en las diferentes ingenierías. Este cambio además estuvo acompañado de un cambio en el lenguaje de programación ya que se adoptó el lenguaje PASCAL (Sanchis, 1980) (Joyanes, 1990) y tanto la formulación de programas así como su diseño se vieron influenciadas por las ideas de la Programación Estructurada (Braunstein y Giogia, 1986) y los Algoritmos y Estructuras de Datos (Wirth, 1986).

Simultáneamente a los cambios originados por las corrientes europeas, si bien influenciados por éstas, se produce en Estados Unidos una difusión masiva del Sistema Operativo Unix en los ámbitos universitarios y del lenguaje con el que se desarrolló que es el C y sus antecesores (Kernigham y Ritchie, 1985) que también sirve al desarrollo de los Sistemas Operativos DOS y Windows de Microsoft. Este lenguaje fue incorporado al dictado durante 1991, y posteriormente fue adecuado durante 1994 a las ideas de Programación Orientada a Objetos en lenguaje C++ (Barclay y Gordon, 1994). La tendencia a la orientación a objetos se ha profundizado con el lenguaje JAVA, que posee una base sintáctica en los lenguajes del tipo C.

Durante la década de los 90 se mantuvo el equilibrio entre los conceptos muy básicos sobre algoritmos y un lenguaje de programación claramente imperativo, pero nunca pudo implementarse la asignatura basada en el paradigma de objetos en forma completa por falta de tiempo a raíz de estas consideraciones sus contenidos han sido transferidos a la asignatura Informática Avanzada (Barnes y Kölling, 3ed 2007).

Durante éste mismo período la asignatura de Métodos Numéricos acompañó los cambios en los lenguajes, incluyendo en parte la enseñanza de una Planilla de Cálculo como modo de presentar una herramienta orientada a determinados problemas de ingeniería. Se redactó un manual sobre soluciones a los métodos numéricos con Excel (Gil Montero, 1999), pero como en el caso de la orientación a objetos tampoco se dispuso del tiempo suficiente para su dictado completo, no obstante marcó la necesidad de contar con herramientas de software interactivas y de uso inmediato con capacidades de graficación.

Desde 2001 se produce una nueva bifurcación con dos orientaciones muy pronunciadas, por un lado la profundización de los conceptos de la Ciencia de la Computación en términos de Algoritmos y Estructuras de Datos, que permitió darle a la asignatura una formalización, similar a las de otras asignaturas de la matemática, en lo que hace al Diseño de Algoritmos (Galvez y otros, 1993) de los que carecía (Braunstein y Giogia, 1986), y además ya es un hecho indiscutible que los conocimientos sobre informática se han hecho imprescindibles como herramienta de Simulación de Sistemas en todas las ciencias y las ingenierías, en particular en las de Electrónica y Computación donde ya no se conciben como temas o disciplinas separadas.

Se destaca que hubiera correspondido estudiar previamente una asignatura de Matemática Discreta (Comellas, 2002), pero hasta el momento las Escuelas de Ingeniería en general y las de Electrónica y Computación en particular no han aceptado la propuesta del Departamento de Computación de su incorporación a los planes de estudio, más que nada por la gran sobrecarga de asignaturas, especialmente del campo de la matemática que ya

soportan las carreras. Esta incorporación de la Matemática Discreta será obligatoria al momento de la Acreditación de la carrera de Ingeniería en Computación ya que ha sido incorporada en las propuestas de CONFEDI y de RedUNCI (2005)

La otra dirección importante es la elección, no ya de un lenguaje de programación sino de una herramienta de gran difusión en el desarrollo de aplicaciones interactivas de métodos numéricos y de graficación en ingeniería, denominado MATLAB (Hanselman, 1996). Éste intérprete fue desarrollado en el lenguaje C y es en su funcionalidad de programación, enteramente similar en su sintaxis a la de aquel .

Ésta elección se ha basado también en la dificultad de la enseñanza, a estudiantes de primer año, de herramientas informáticas y lenguajes de programación para aplicarlas a Problemas de Ingeniería que desconocen casi completamente, inclusive carecen de conceptos de física básica, lo que impide el uso de material didáctico orientado a problemas (Biran y Breiner, 1999) y en relación a los Métodos Numéricos (Nakamura, 1997). Se hizo necesario adaptar los textos de MATLAB a la enseñanza de la programación y los algoritmos para lo cual se desarrolló un Apunte Operacional, y que es un texto ejecutable en el intérprete como Literatura Computacional (van Dyke, 1987) lo que permite unir en un mismo texto las explicaciones teóricas significativas con el código correspondiente. Éste material se desarrolló en 2000 con la finalidad de ser utilizado como transparencias de ejemplos de problemas y ejemplos resueltos de cierta longitud y complejidad, lo que no puede lograrse en un tiempo razonable, con uso exclusivo del pizarrón.

Durante 2001, 2002 y 2003 se ha incrementado el uso del Software Libre en ingeniería, en particular la existencia del émulo de MATLAB denominado GNU-OCTAVE (Eaton, 1997) lo que permitió utilizar dicho software sin restricciones de licencias de ningún tipo. Actualmente se está estudiando en la cátedra otro lenguaje de emulación denominado SCILAB (Gómez, 1999), que si bien es de propiedad del INRIA su uso es gratuito pero no es libre el uso del código fuente y sus prestaciones son superiores en algunos casos al del mismo MATLAB actual.

En el año 2007, con el objetivo de unificar criterios de dictado y evaluación, se propuso utilizar como único lenguaje de programación el lenguaje C++ ya que de acuerdo a las experiencias previas del dictado de dos lenguajes (Matlab y C++) ha resultado altamente inconveniente y definitivamente imposible de cumplir en el término de un cuatrimestre con un nivel de adquisición de conocimientos que pueda considerarse como una competencia adquirida.

A pesar de que la dedicación a un solo lenguaje ha mejorado la profundización de algunos conceptos se perdió una de las cualidades de los lenguajes interpretados como MATLAB/OCTAVE, cual es su facilidad para la construcción de prototipos sin consideración a criterios estrictos de especificación y por otra parte la capacidad de representar conocimientos mediante técnicas multimediales que acercan el pensamiento abstracto con concreciones visuales. Pero dicho lenguaje carecía de la generalidad necesaria para representar los distintos constructos de la Ciencia de la Computación por lo cual ha resultado conveniente seleccionar el lenguaje Python para implementar los conceptos fundamentales de la materia.

Uno de los aspectos centrales del Método Científico y en particular el de la Física, que es una de las ciencias básicas que sustentan los distintos diseños y obras de las ingenierías, es la experimentación, es decir, la posibilidad de repetir experiencias de una manera controlada, de forma tal que se pueda poner en evidencia la corrección de las Teorías Formales en las que se sustentan.

Esta interpretación de la experimentación es el que debe animar al estudiante ante los problemas de la ingeniería ya que las herramientas informáticas adecuadas le permitirán poner a prueba su propia comprensión de las teorías, mediante la observación de los resultados controlados por la lógica de la especificación de programas de simulación y su puesta en funcionamiento mediante su implementación computacional.

La ubicación de la asignatura de Informática en el primer año de los diferentes planes de estudio de las ingenierías, pone a prueba a quienes ingresan al sistema universitario, frente a un sistema de aprendizaje basado en el autocontrol del propio conocimiento, lo que requiere ante todo dejar de creer en los textos sin un pensamiento crítico propio y por ende le exige poner a prueba y experimentar con la realidad acotada o simulación que es posible construir mediante la matemática, la física y la informática.

Otro aspecto, que a lo largo de varios años de enseñanza de la informática se ha puesto en evidencia, es la gran similitud entre con el aprendizaje de idiomas extranjeros, y también con el entrenamiento deportivo. En todos los casos la construcción de estructuras de pensamiento que permiten hablar o jugar en forma automática, requieren de un considerable tiempo dedicado a la práctica. Basta recordar que no es posible hablar ningún idioma extranjero mientras no se puede pensar en él y esto se logra luego de varios años de estudio o convivencia.

En síntesis, para entender, y por ende aprender, no basta con aceptar como buenas las relaciones lógicas, por correctas que éstas sean, que momentáneamente parecen evidentes y casi de sentido común cuando son enunciadas por el profesor en el dictado de una clase o en una lectura circunstancial, es imprescindible dudar, poner a prueba, experimentar, para lo cual nada más gratificante y creativo que construir una solución algorítmica a un problema y comprobar que funciona.

METODOLOGÍA DE ENSEÑANZA

Las etapas de construcción y elaboración de conocimientos son sustentadas mediante la exposición dialogada como estrategia didáctica y el empleo de proyección de diapositivas, transparencias, pizarrón y proyector multimedia como materiales didácticos. Todos los materiales de estudio, incluyendo sistema de consultas, preguntas frecuentes, e-mail, evaluaciones, etc., se disponen en el sistema informático de aprendizaje del Departamento de Computación (Laboratorio de Enseñanza Virtual – LEV. http://lev.efn.uncor.edu)

Las fases de ejercitación y aplicación de los contenidos de la asignatura, se fundamentan tanto en el desarrollo teórico como en el práctico del presente curso. Se realizan dos tipos diferenciados de actividades en coordinación con el desarrollo de la autonomía de aprendizaje, consistentes en la solución de problemas acotados y en la elaboración de un proyecto informático integrador realizado en equipo. En estas instancias el trabajo individual y grupal, permite la conformación de ideas y el establecimiento de relaciones entre el conocimiento adquirido y situaciones nuevas planteadas desde otras problemáticas de la misma disciplina.

El dictado se realizará en 16 clases de 4hs 25min (reloj) consistentes en la presentación teórica de los temas por parte del docente, las que no podrán superar 2hs:00min en cada sesión.

La presentación de temas prácticos se realizará preferentemente, en el caso de disponer de equipos de computación, en el marco de tareas de laboratorio, previamente asignadas por el docente coincidentes con el tema teórico previo, asumiendo el docente el rol de tutor y mediante evaluaciones formativas en cada clase.

En el caso de no disponer del laboratorio se realizarán ejercitaciones y simulaciones de escritorio que permitan poner de manifiesto los objetivos de la asignatura.

El proceso de elaboración del proyecto integrador será seguido mediante entregas parciales pautadas en el LEV, así como la devolución de las evaluaciones parciales.

EVALUACIÓN

La evaluación consta de diferentes instancias que se describen a continuación:

- **Evaluación conceptual (EC)**: comprende 6 (seis) Evaluaciones Conceptuales, cada una compuesta por un conjunto breve de ejercicios que permitan demostrar al alumno su comprensión del tema desarrollado en la o las clases anteriores. En estos ejercicios se evalúa la capacidad de interpretar código escrito en Python/C++ que utilice las herramientas vistas hasta la clase anterior. La evaluación conceptual consta de ejercicios que serán evaluados en línea automáticamente a través del Laboratorio de Educación Virtual (LEV-Moodle). El valor asignado a cada evaluación es de 0 a 10 puntos, haciendo un total de hasta 60 puntos el conjunto de todas las Evaluaciones Conceptuales. Se prevén 2 (dos) recuperatorios de 20

puntos adicionales cada uno, el primero corresponde a las EC1, EC2 y EC3, y el segundo, a las EC4, EC5 y EC6. Los puntos adicionales se deben sumar a los ya obtenidos de las EC que se recuperan, no pudiendo pasar la suma de 30 puntos para las tres primeras EC (EC1, EC2 y EC3) y su recuperatorio REC123, y de 30 puntos para las tres restantes (EC4, EC5 y EC6) y su recuperatorio REC456. Para aprobar esta instancia de evaluación, se debe alcanzar un rendimiento igual o superior a ⅔ del total, es decir, al menos 40 puntos de los 60 en total. Tiene carácter acreditativo.

- **Trabajos Prácticos (TP)**: comprende 3 (tres) Trabajos Prácticos a lo largo del curso, consistentes en el desarrollo de la solución algorítmica a problemas propuestos por la Cátedra, y su especificación en lenguaje Python/C++, que por su extensión resulta imposible evaluar en una situación áulica. Las soluciones deben estar libres de errores sintácticos que impidan su ejecución por parte del intérprete/compilador, como requisito para su evaluación. Las soluciones propuestas son evaluadas objetivamente verificando en cumplimiento de los requerimientos. El valor asignado a cada trabajo práctico es de 0 a 2 puntos, siendo 0 desaprobado, 1 aprobado con errores, y 2 aprobado sin errores. Para aprobar esta instancia de evaluación se debe alcanzar un rendimiento igual o superior a los ⅔ del total, es decir, al menos 4 puntos de los 6 totale. Estas actividades son de carácter extra-áulico, para las cuales los alumnos disponen de una semana para su resolución y entrega.

- **Especificación de algoritmos (EA)**: se realiza una ejercitación consistente en el desarrollo en máquina de un programa en Python/C++ correspondiente al enunciado de un algoritmo con un nivel de especificación comparable a un pseudocódigo. La implementación debe estar libre de errores sintácticos que impidan su ejecución por parte del intérprete/compilador, como requisito para su evaluación. La misma propuesta es evaluada objetivamente, verificando el cumplimiento de los requerimientos automáticamente a través del LEV. El valor asignado a la EA es de 0 a 100 puntos. Para ser aprobada, se debe alcanzar un rendimiento igual o superior al ⅗ del total, es decir al menos 60 puntos de los 100 en total. Esta actividad prevé un recuperatorio. Tiene carácter acreditativo.

Condiciones de regularización

Para alcanzar la condición de regular, el alumno debe cumplir las siguientes condiciones:

1) Asistir al menos al 80% de las clases. La asistencia es registrada en el LEV mediante la presentación a las EC y/o recuperatorios, siendo necesario cumplir con al menos 6 presentaciones de las 8 disponibles (6 EC + 2 recuperatorios)

2) Alcanzar un rendimiento global no inferior a ⅔ (40 de 60 puntos) en las Evaluaciones Conceptuales (EC).

3) Alcanzar un rendimiento global no inferior a ⅔ % (4 de 6 puntos) en los Trabajos Prácticos (TP).

Condiciones de promoción

Para alcanzar la promoción, el alumno debe cumplir las siguientes condiciones:

1) Alcanzar la condición de alumno regular para lo cual se deben cumplir las condiciones indicadas al respecto.

2) Alcanzar un rendimiento no inferior a ⅗ del total (60 de 100 puntos) en la Especificación de Algoritmos (EA).

Cumplidas estas condiciones, la nota final de promoción se calculará según la siguiente fórmula:

$$\text{Nota Final} = \text{redondear}\left(\left(\tfrac{2}{3} * EC + \tfrac{3}{5} * EA\right)/10\right)$$

Donde EC y EA están expresados en puntos de [0,60] y de [0,100], respectivamente.

Examen final

A. Los alumnos **regulares** deben rendir un examen equivalente a la actividad Especificación de Algoritmos (EA), el cual será calificado del mismo modo que para los alumnos que alcanzaron la promoción, considerando para la variable EC el rendimiento alcanzado durante la cursada, y para la variable EA el rendimiento alcanzado en el examen final.
B. Los alumnos **libres** deben rendir un examen que consta de dos partes:
    a. Una prueba de competencias con la misma metodología y objetivos que la Evaluación Conceptual (EC), acotada a 6 ejercicios que serán evaluados automáticamente en el LEV. La aprobación de esta primera parte es requisito excluyente para la prosecución del examen, y se deberá obtener un rendimiento no inferior ⅔ .
    b. Un examen de Especificación de Algoritmos (EA), con la misma modalidad y objetivos que el requerido a los alumnos regulares.

La Nota Final final de examen para los casos A) y B) se obtendrá por la siguiente expresión:

$$\text{Nota Final} = \text{redondear}((⅔ * EC + ⅗ * EA)/10)$$

Donde EC y EA están expresados en puntos de [0,60] y de [0,100], respectivamente.

CONTENIDOS TEMÁTICOS

Unidad 1. Introducción a la informática

Introducción a la programación. Solución de problemas y desarrollo de software. Algoritmos. Errores en programación. Hardware y conceptos de almacenamiento.

Unidad 2. Introducción a la especificación de programas imperativos

Herramienta de programación. Estilo de programación. Constantes y operaciones aritméticas. Variables y declaraciones. Tipos de datos. Procedimiento para el desarrollo de software. Operaciones de asignación. Formato de salida. Funciones de biblioteca. Entrada y salida estándar de información. Aplicaciones.

Unidad 3. Estructuras de control

Operadores lógicos y relacionales. Estructuras de decisión. La estructura de decisión simple. La estructura de decisión doble. Estructuras de decisión anidadas. La estructura de decisión múltiple. Estructuras de repetición. Las estructuras de repetición indefinidas. La estructura de repetición definida. Estructuras de repetición anidadas. Técnicas de programación estructurada. Aplicaciones.

Unidad 4. Funciones definidas por el usuario

Declaración de funciones y parámetros. Prototipos. Argumentos. Alcance de variables. Clases de almacenamiento de variables. Recursividad. Aplicaciones.

Unidad 5. Tipos de datos arreglos y punteros

Arreglos unidimensionales. Inicialización de arreglos. Arreglos bidimensionales. Arreglos como argumentos. Algoritmos de búsqueda y ordenamiento. Aplicaciones. Direcciones y punteros. Nombres de arreglos como punteros. Transmisión de direcciones.

Unidad 6. Estructuras de datos compuestos

Estructuras sencillas. Arreglo de estructuras. Estructuras como argumentos de función. Listas enlazadas. Asignación dinámica de estructuras de datos. Uniones.

Unidad 7. Entrada/salida de información

Lectura y escritura de archivos. Acceso aleatorio de archivos. Flujo de archivos como argumento de función. Excepciones y comprobación de archivos. Bibliotecas de entrada/salida. Aplicaciones.

LISTADO DE ACTIVIDADES PRÁCTICAS Y/O DE LABORATORIO

Actividades de Formación Experimental y Resolución de Problemas

El alumno realizará actividades de programación en el Laboratorio de Computación que se corresponden con los ejercicios propuestos como trabajos a realizar durante el dictado de las clases teórico-prácticas. Son ejercicios breves vinculados directamente al tema que se está considerando en la clase pero tiene todas las características básicas de la especificación de programas.

Actividades de Proyecto y Diseño

Los Trabajos Prácticos serán realizados por el alumno en tiempo diferido respecto de las clases en las cuales se dictaron los temas específicos y tendrá 48hs para su realización debiendo entregar los trabajos en formato digital por medio del LEV para su corrección y calificación.
La corrección y calificación se realizará en horarios de Laboratorio donde los docentes asignados evaluarán y harán entrega de las calificaciones de los mismos.

La siguiente es un listado de los temas a desarrollar en los Trabajos Prácticos y sus definiciones estarán disponibles en el LEV en el Aula de Evaluaciones:

1.- Codificación de algoritmos en computadoras. Tipos de datos básicos y sus expresiones.
2.- Especificación de programas basados en control imperativo. Especificación de programas con tipos de datos en secuencia, arreglos/listas.
3.- Especificaciones modular de programas, funciones, procedimientos, módulos. Especificaciones de programas con tipos de datos compuestos y persistentes.

Tienen por objeto acreditar que el alumno ha adquirido las siguientes habilidades y técnicas, relacionadas preferentemente a la totalidad de los contenidos de la asignatura:

- Aplicar la informática a un problema de ciencias básicas, de ingeniería o de sistemas de información, desde su formulación simbólico-matemática o de información hasta su implementación en un lenguaje informático.
- Adquirir la habilidad para la depuración de algoritmos y programas mediante una técnica basada en principios lógicos.
- Experimentar con diferentes criterios de diseño.

DISTRIBUCIÓN DE LA CARGA HORARIA

ACTIVIDAD		HORAS
TEÓRICA		32
FORMACIÓN PRACTICA:		
	○ FORMACIÓN EXPERIMENTAL	20
	○ RESOLUCIÓN DE PROBLEMAS	20
	○ ACTIVIDADES DE PROYECTO Y DISEÑO	12
	○ PPS	
		84

DEDICADAS POR EL ALUMNO FUERA DE CLASE

ACTIVIDAD		HORAS
PREPARACION TEÓRICA		24
PREPARACION PRACTICA		
	○ EXPERIMENTAL DE LABORATORIO	0
	○ EXPERIMENTAL DE CAMPO	0
	○ RESOLUCIÓN DE PROBLEMAS	36
	o PROYECTO Y DISEÑO	24
	TOTAL DE LA CARGA HORARIA	84

CRONOGRAMA DE ACTIVIDADES

Las actividades se distribuyen en 16 clases semanales en base al siguiente cronograma, el cual puede alterarse en función de los feriados.

Clase	Tema	Capítulo	EC	TP
1	Unidad 1	(1) - [1]		
2	Unidad 2	(2) - [2]		
3	Unidad 2	(3) - [3]		
4	Unidad 3	(4) - [4]	EC1	
5	Unidad 3	(5) - [4]	EC2	TP1
6	Unidad 3	(5) - [4]	EC3	
7	Unidad 5	(11) - [5]		
8	Unidad 5	(11) - [5]	REC123	
9	Unidad 4	(6) - [6]		TP2
10	Unidad 4	(6) - [6]	EC4	
11	Unidad 6	(13) - [7]	EC5	
12	Unidad 7	(13) - [7]	EC6	
13	Unidad 7	(8) - [8]		TP3
14	Todas las unidades		REC456	
15	Parcial EA			
16	Recuperatorio EA			

Referencias de los capítulos

[cap.] Marzal, Andrés y García, Isabel (2003). Introducción a la programación com Python.

(cap.) Bronson, G. (2007). C++ para Ingeniería y Ciencias.(2da. edición)

BIBLIOGRAFÍA

Básica

- Marzal, Andrés y García, Isabel, *Introducción a la programación con Python*, Edición 2003 de Andrés Marzal Varó e Isabel Gracia Luengo. Reservados todos los derechos. Esta (Edición Internet) se puede reproducir con fines autodidactas o para su uso en centros públicos de enseñanza, exclusivamente.
- Bronson, Gary, *C++ para Ingeniería y Ciencias* (2da. edición), International Thomson Editores, México, 2007

Recomendada

- van Rossun, Guido: *El Tutorial de Python*. Editor Fred L. Drake, Jr.
- Deitel, H. M., Deitel, P. J., *Cómo programar en C++,* Pearson Educación, 2003
- Galve, J. et al., *Algorítmica*: *Diseño y análisis de algoritmos funcionales e imperativos*. Addison-Wesley Iberoamericana / Ra-Ma, 1993