FACULTAD DE INGENIERIA
Universidad de Buenos Aires

Planificaciones

9501 - Computación

Docente responsable: JIMENEZ REY MYRIAN ELIZABETH

OBJETIVOS

ENSEÑANZA

Objetivos Generales

● Que el alumno adquiera una visión global de la Computación para comprender el aspecto científico de la actual sociedad informatizada. ● Que el alumno comprenda conceptos y técnicas de la disciplina que en el ejercicio profesional le posibiliten la interacción con profesionales en Informática sin problemas de comunicación. ● Que el alumno logre compenetrarse con las tecnologías y herramientas fundamentales de la Computación para usar la computadora como instrumento de trabajo, conociendo su precisión, capacidad y limitaciones. ● Que el alumno se familiarice con el modo de pensar en Ingeniería.

Objetivos Específicos

● Que el alumno tome conciencia de la importancia de la Algoritmia como paradigma de resolución de problemas y de la Programación como práctica en la resolución de problemas con la computadora. ● Que el alumno desarrolle la capacidad de relacionar esquemas de solución de problemas con la resolución de problemas algorítmicos, con hincapié en el método científico. ● Que el alumno desarrolle la capacidad de Análisis, Sistematización, Programación y Procesamiento de distintos problemas de tipo técnico-científicos para utilizar dichos conocimientos en su formación académica actual y en su ejercicio profesional futuro.

APRENDIZAJE

Competencias Generales de Ingeniería

● Conocimiento de tecnologías y métodos básicos para adquirir capacidad para el aprendizaje de nuevos métodos y tecnologías, y versatilidad para la adaptación a nuevas situaciones. ● Capacidad de resolver problemas con iniciativa, toma de decisiones, creatividad y sentido crítico. ● Capacidad de comunicar y transmitir conocimientos, habilidades y destrezas, comprendiendo la responsabilidad profesional de la actividad del Ingeniero.

Competencias Específicas de Computación

Cognitivas (Saber conocer)
● Conocimiento general sobre Algoritmia y conocimientos básicos sobre el uso y Programación Imperativa de computadoras, y de la sintaxis de un Lenguaje de Programación imperativo. ● Conocimiento de la organización, capacidades y funcionamiento de las computadoras y los fundamentos de su programación, y su aplicación para la resolución de problemas propios de la Ingeniería. ● Conocimiento y aplicación de los procedimientos algorítmicos básicos de las tecnologías informáticas para diseñar soluciones a problemas, analizando la idoneidad de los algoritmos propuestos, y conocimiento de los tipos y estructuras de datos fundamentales y su utilización apropiada para la resolución de un problema (programa eficiente).

Procedimentales (Saber hacer)
● Capacidad de resolución de problemas mediante algoritmos y programas que permitan ejecutarlos en una computadora (programa eficaz). ● Capacidad para descomponer un problema real en subproblemas para su posterior codificación en un programa. ● Capacidad para documentar programas con claridad y sencillez (programa inteligible) y, comprender documentación técnica y reutilizar código desarrollado por terceras partes.

Actitudinales (Saber ser)
● Motivación por la claridad, sencillez, eficacia y eficiencia algorítmica de problemas y su traducción a programas. ● Capacidad para debatir y concluir las distintas soluciones algorítmicas a un problema traducidas a programas.

Competencias Transversales o Genéricas

● Capacidad para la autoorganización y planificación del trabajo individual y del proceso de aprendizaje. ● Capacidad para el trabajo en grupo. ● Capacidad de análisis y síntesis. ● Motivación por la calidad del resultado. ● Disposición para el compromiso, la responsabilidad, la colaboración, el empeño, la dedicación, la solidaridad, la honestidad y el respeto en el trabajo individual y grupal durante el proceso de construcción del conocimiento.

CONTENIDOS MÍNIMOS
-

PROGRAMA SINTÉTICO
Alcance de las Ciencias de la Computación. Técnicas para representar y almacenar información y forma en que las máquinas digitales manipulan los datos. Software de sistema, de aplicación y de traducción. Lenguajes de programación. Algoritmia y programación básicas.

PROGRAMA ANALÍTICO
Unidades de Programa

INTRODUCCIÓN A LA COMPUTACIÓN
Algoritmos. Alcance de las ciencias de la computación. Arquitectura de computadoras. Sistemas de numeración binario y hexadecimal.
REPRESENTACIÓN Y ALMACENAMIENTO DE DATOS
Unidad central de almacenamiento. Memoria secundaria. Dispositivos periféricos. Códigos: para representar y almacenar símbolos (ASCII y EBCDIC), números enteros (en complemento a dos y en exceso) y números reales (punto flotante). Confiabilidad: métodos de detección y corrección de errores.
MANIPULACIÓN DE DATOS
Unidad central de procesamiento. Codificación y almacenamiento de programas. Lenguaje de máquina. Ejecución de programas.
NOCIONES DE SOFTWARE
Software de sistema, de aplicación y de desarrollo. El sistema operativo: funciones, interfaz basada en caracteres e interfaz gráfica. Redes de computadoras y software de comunicación.
INTRODUCCIÓN A LA ALGORITMIA Y A LA PROGRAMACIÓN
Desarrollo de algoritmos: teoría de resolución de problemas de Polya aplicada a la algoritmia. Enfoques descendente (top down) y ascendente (bottom up). Pensamiento computacional. Tipos estándar y definición de variables globales. Primitivas de especificación de algoritmos: asignación, entrada y salida estándar de datos, expresiones, estructuras de control selectivas (simples y múltiples), repetitivas (indefinidas y definidas) y de invocación de subalgoritmos (transferencia/retorno). Procesamiento de secuencias. Eficiencia, generalidad y corrección de algoritmos. Lenguajes de programación: historia, la programación imperativa o por procedimientos como paradigma de comunicación de algoritmos a computadoras; traducción e interpretación de programas; paradigmas de programación abstracta (programación orientada a objetos, programación lógica, programación funcional) y lenguajes que las sustentan.
LENGUAJE PYTHON
Estructura de un programa Python y entorno integrado de desarrollo y aprendizaje IDLE. Modelo de Programa Tipo. Tipos de datos y variables: declaraciones. Funciones de librería. Enunciados de documentación interna y de entradas y salidas. Definición de tipos estructurados: secuencias. Mutabilidad, inmutabilidad y representación de la información en memoria en programas. Archivos de texto (memoria persistente): estructura, caracteres de control, funciones predefinidas y aplicaciones para captura de datos y comunicación de resultados de programas.
UNIDADES DE PROGRAMACIÓN
Funciones como estructuras de control de transferencia-retorno y como recursos de programación. Parámetros: declaración y pasaje. Definición de variables locales; reglas de alcance. Reusabilidad del software. Principios de modularización: cohesión y acoplamiento.
AGRUPAMIENTO DE DATOS EN ARREGLOS
Definición y manipulación de arreglos de una y dos dimensiones. Arreglos como parámetros. Búsqueda y ordenamiento de elementos en arreglos. Aplicaciones: aritmética de alta precisión, álgebra de polinomios y matrices, resolución algebraica de sistemas de ecuaciones.

BIBLIOGRAFÍA
1 INTRODUCCIÓN A LA COMPUTACIÓN. J. Glenn Brookshear. Pearson Educación, S. A. Madrid. Undécima edición, 2012. ISBN: 97884782911397.

2 ALGORITMOS Y PROGRAMACIÓN I. CON LENGUAJE PYTHON. Rosita Wachenchauzer, Margarita Manterola, Maximiliano Curia, Marcos Medrano y Nicolás Paez. 9 de marzo de 2011. En https://librosweb.es/libro/algoritmos_python/

3 INTRODUCCIÓN A LA PROGRAMACIÓN CON PYTHON 3. Andrés Marzal Varó, Isabel Gracia Luengo y Pedro García Sevilla. Departamento de Lenguajes y Sistemas Informáticos. Códigos de Asignaturas: EI1003 y MT1003. Publicaciones Universitat Jaume I. Primera edición, 2014. ISBN: 978-84-697-1178-1. En http://repositori.uji.es/xmlui/handle/10234/102653

4 GUÍAS TEÓRICAS Y PRÁCTICAS DEL CURSO DE COMPUTACIÓN (Elaboración Propia).

5 PYTHON SOFTWARE FOUNDATION. En https://www.python.org/about/

6 EL TUTORIAL DE PYTHON. Guido van Rossum.
En http://docs.python.org.ar/tutorial/pdfs/TutorialPython3.pdf

7 GUÍA DE ESTILO DEL CÓDIGO PYTHON. Guido van Rossum y Barry Warsaw. Traducción al castellano por Raúl
González Duque. 10 de Agosto de 2007. En
http://mmc.geofisica.unam.mx/edp/Herramientas/Lenguajes/Python/Convenciones.pdf

RÉGIMEN DE CURSADA

Metodología de enseñanza

La estrategia pedagógica utilizada para el desarrollo de contenidos de Computación se sintetiza en la experiencia de pensar para crear juntos. Integra enseñar a pensar y enseñar contenidos. El desarrollo de los contenidos teóricos y prácticos se aborda desde tres núcleos centrales (algoritmo, programa y computadora) en forma interrelacionada, iterativa e incremental y se organiza en espiral. El descubrimiento de algoritmos y el desarrollo de programas se complejiza y el conocimiento de la computadora se profundiza durante las 16 semanas de clase.

Para solucionar problemas con la computadora los estudiantes utilizan el Modelo de Solución de Problemas de Polya aplicado al ámbito de la construcción de programas que consta de cuatro fases: 1 Análisis, deben comprender en qué consiste el problema a resolver con la construcción del programa. 2 Diseño, deben diseñar una estrategia para definir recursos y descubrir el algoritmo. El Análisis del problema y el Diseño de la solución, son las fases algorítmicas y constituyen el desafío creativo de aprendizaje porque requiere que los estudiantes desplieguen el pensamiento computacional, competencia clave de aprendizaje en la formación de los estudiantes de ingeniería. 3 Codificación, deben implementar la estrategia para construir el programa. 4 Evaluación, deben ejecutar el programa construido para comprobar que la solución es correcta. La Codificación y la Evaluación son las fases de programación.

Cada una de las fases del Modelo de Solución de Problemas con la Computadora se expande en una Guía para la Creación de Programas en forma de Mapa Conceptual como dispositivo didáctico para ayudar a los estudiantes a descubrir un algoritmo que solucione el problema propuesto y a codificarlo como programa.

Con una visión sistémica, se diseñó un Modelo Evolutivo de Enseñanza, Aprendizaje y Evaluación que denominamos CRAC, siglas de Comprensión, Reflexión, Acción y Comunicación, procesos componentes de una trama educativa compleja que intenta efectivizar una enseñanza y aprendizaje activos para generar en el estudiante un pensamiento vinculante, que no aísle y separe los procesos sino que los distinga y los una.

La práctica educativa activa y reflexiva se focaliza en hacer visible el pensamiento de los estudiantes de una manera sistematizada y formalizada mediante el diseño y la implementación de tablas para valorar el impacto de la metacognición en el aprendizaje de la algoritmia y la programación. Se utilizan rutinas de pensamiento en aula real y aula virtual durante el proceso de descubrimiento de algoritmos (como herramientas, como estructuras y como patrones de comportamiento) porque promueven movimientos específicos de pensamiento e implican un proceso que al ponerse en práctica permite que el pensamiento de los estudiantes se haga visible a medida que expresan sus ideas, debaten y reflexionan en torno a ellas.

El Aula de Computación está constituida por espacios de clases presenciales, Aula real, Lab E, y espacios de clases virtuales, Aula virtual, Google Drive Institucional, donde los estudiantes aprenden a través de la experiencia de pensar para crear con otros (pensamiento computacional). En ambos espacios, utilizan las tablas para ingeniar algoritmos: en Aula real, todos los participantes piensan con sus pares y el profesor para la construcción del conocimiento nuevo, y en Aula virtual, con sus compañeros de taller (grupos de tres integrantes) para la apropiación del nuevo conocimiento, acompañados por el profesor quien interviene para modelar el pensamiento.

“Un Lugar para Algoritmiar”, tanto en aula real como virtual, está constituido por el espacio que proporcionan

las tablas: Tabla 1 (¿Qué problema se debe resolver? ¿Cuáles son los recursos necesarios para resolver el problema?); Tabla 2 Análisis del Problema (columnas Caso Representativo, Estado Inicial, Estado Final, Estados Intermedios de Transformación) y Tabla 3 Diseño de la Solución (columnas Estado Genérico de Adaptación, Descomposición del Problema en Subproblemas, Naturaleza de los Subproblemas, Primitivas de Programación). Cada tabla conecta con la anterior y cada columna (en cada tabla) conecta con la columna anterior. “Un Lugar para Programar” está constituido por el IDLE Python donde los estudiantes codifican los algoritmos utilizando otra tabla, Tabla 4 Modelo de Programa Tipo en Python, un texto organizativo que implica las fases del proceso de construcción de programas a través de las secciones declarativa, algorítmica y evaluativa.
La estructura de las tablas ofrece un andamiaje al pensamiento visible para que los estudiantes puedan enfrentar la mayor dificultad de aprendizaje: ingeniar algoritmos con calidad de diseño para solucionar problemas con la computadora.

Modalidad de Evaluación Parcial

En las Evaluaciones Parciales Obligatorias (Parte Práctica y Parte Teórica) se valora la adquisición de las competencias cognitivas (saber conocer) y procedimentales (saber hacer) de manera individual.
La Parte Práctica consiste en una evaluación integradora escrita de conocimientos procedimentales mediante desarrollo en papel de un programa en lenguaje Python. En la corrección se tendrá en cuenta la aplicación de la metodología de desarrollo (programa inteligible), el funcionamiento del programa (programa eficaz) y la calidad del diseño (programa eficiente).
La Parte Teórica consiste en una prueba semiestructurada escrita de respuesta restringida en la cual se evalúa el dominio de conocimientos sobre arquitectura de una computadora. La comprensión y la reflexión serán necesarias para lograr una respuesta precisa (responder solamente aquello que se pregunta), clara (utilizar la terminología propia del campo disciplinar) y correcta (la respuesta debe ser pertinente).
Las Evaluaciones Parciales Obligatorias constituyen evaluaciones formativas, autoevaluativas y calificativas (acreditación numérica).

CALENDARIO DE CLASES

Semana	Temas de teoría	Resolución de problemas	Laboratorio	Otro tipo	Fecha entrega Informe TP	Bibliografía básica
<1> 09/03 al 14/03	Algoritmo y Programa. Alcance de las Ciencias de Computación. Desarrollo de algoritmos: teoría de resolución de problemas de Polya aplicada a la algoritmia. Desarrollo de Programas: (a) Análisis y Documentación del Problema a Resolver; (b) Definición de Recursos y Esquematización, Refinamientos y Prueba del Algoritmo; (c) Formalización del Programa, y (d) Depuración y Prueba del Programa. Modelo de un Programa Tipo (Prólogo-Resolución-Epílogo). Lenguajes de programación: paradigmas de comunicación de algoritmos a las computadoras ; traducción e interpretación de programas. Arquitectura Básica de una Computadora.	Estructura de un programa Python y ambiente integrado de desarrollo y aprendizaje. Recursos Básicos de un Programa en Lenguaje Python: Variables Enteras, Reales y de Caracteres. Enunciados de Documentación Interna. Instrucciones de Asignación, de Lectura desde Teclado, de Escritura en Pantalla. Estructura de un programa en lenguaje Python. Conformación Grupos de Trabajo. Especificación Actividad Grupal Formativa Nº 1 AGF1 (Introductoria al Módulo 1).			15 de marzo	1 a 7
<2> 16/03 al 21/03	Sistema de Numeración Binario. Cambios de Base de Numeración. Numeración Hexadecimal como Abstracción de la Binaria. Resolución	Estructuras de Selección Simple (IF…, IF…ELSE…). Anidamiento de Selecciones Simples y Selección Múltiple (IF… ELIF… ELSE…). Comparación y Equivalencia. Resolución Problemas Propuestos Prácticos. Especificación Actividad Grupal Formativa Nº 2 AGF2. Descubrimiento de algoritmos en Taller “Un Lugar para Algoritmiar”. Codificación de algoritmos en IDLE Python “Un Lugar para Programar”.			22 de marzo	1 a 7

Semana	Temas de teoría	Resolución de problemas	Laboratorio	Otro tipo	Fecha entrega Informe TP	Bibliografía básica
	Problemas Propuestos Teóricos.					
<3> 23/03 al 28/03	Organización de la Unidad Central de Almacenamiento. Unidades de Información en una Computadora: Órdenes, Valores Enteros, Valores Fraccionarios y Símbolos. Representación de Símbolos: ASCII/UNICODE. Representación de Valores Enteros: Complemento a Dos. Resolución Problemas Propuestos Teóricos.	Foro de Discusión AGF2 Estructura de Repetición con Control Inicial (WHILE). Condiciones Compuestas de Control de Repetición. Lógica Binaria Resolución Problemas Propuestos Prácticos. Especificación Actividad Grupal Formativa Nº 3 AGF3. Descubrimiento de algoritmos en Taller “Un Lugar para Algoritmiar”. Codificación de algoritmos en IDLE Python “Un Lugar para Programar”. ESPECIFICACIÓN PRIMER TRABAJO GRUPAL FORMATIVO TGF1.			29 de marzo / 19 de abril	1 a 7
<4> 30/03 al 04/04	Representación de Valores Fraccionarios: Coma Flotante. Resolución Problemas Propuestos Teóricos.	Foro de Discusión AGF3. PRESENTACIÓN PRIMER TRABAJO GRUPAL FORMATIVO TGF1. DESARROLLO PRIMER TRABAJO GRUPAL FORMATIVO TGF1. Descubrimiento de algoritmos en Taller “Un Lugar para Algoritmiar”. Codificación de algoritmos en IDLE Python “Un Lugar para Programar”.			19 de abril	1 a 7
<5> 06/04 al 11/04	Organización y Funcionamiento de la Unidad Central de Procesamiento. Órdenes en Lenguaje de Máquina y Almacenamiento de Programas. El Ciclo de Máquina. Resolución Problemas Propuestos Teóricos.	DESARROLLO PRIMER TRABAJO GRUPAL FORMATIVO TGF1. Descubrimiento de algoritmos en Taller “Un Lugar para Algoritmiar”. Codificación de algoritmos en IDLE Python “Un Lugar para Programar”.			19 de abril	1 a 7
<6> 13/04 al 18/04	Taxonomía del Software: de sistema, de aplicación y de desarrollo. El sistema operativo: funciones, interfaz	DESARROLLO PRIMER TRABAJO GRUPAL FORMATIVO TGF1. Descubrimiento de algoritmos en Taller “Un Lugar para Algoritmiar”. Codificación de algoritmos en IDLE Python “Un Lugar para Programar”. ENTREGA PRIMER TRABAJO GRUPAL FORMATIVO TGF1 vía			19 de abril	1 a 7

Semana	Temas de teoría	Resolución de problemas	Laboratorio	Otro tipo	Fecha entrega Informe TP	Bibliografía básica
	basada en caracteres e interfaz gráfica. Taxonomía del Hardware. Redes y software de comunicación. Resolución Problemas Propuestos Teóricos.	Campus FIUBA Domingo 19 de abril 23:59. CLASE EXTRAORDINARIA OPTATIVA DE CONSULTA (PREVIA ENTREGA PRIMER TRABAJO GRUPAL FORMATIVO TGF1). DÍA Y HORARIO A ESTABLECER.				
<7> 20/04 al 25/04	EVALUACIÓN PRIMER TRABAJO GRUPAL FORMATIVO TGF1 en LAB E (inscripción previa). CLASE EXTRAORDINARIA OPTATIVA DE CONSULTA (PREVIA PRIMER PARCIAL TEÓRICO). DÍA Y HORARIO A ESTABLECER.	EVALUACIÓN PRIMER TRABAJO GRUPAL FORMATIVO TGF1 en LAB E (inscripción previa). CLASE EXTRAORDINARIA OPTATIVA DE CONSULTA (PREVIA PRIMER PARCIAL PRÁCTICO). DÍA Y HORARIO A ESTABLECER.				1 a 7
<8> 27/04 al 02/05	PRIMER PARCIAL TEÓRICO (en Laboratorio E) Evaluación de conocimientos contenidos teóricos (mediante prueba semiestructurada escrita de respuesta restringida).	PRIMER PARCIAL PRÁCTICO (en Laboratorio E) Evaluación de habilidades contenidos prácticos (sobre desarrollo en papel de un programa integrador en lenguaje Python). Especificación Actividad Grupal Formativa Nº 4 AGF4 (Introductoria al Módulo 2)				1 a 7
<9> 04/05 al 09/05	Comunicación con Periféricos: Puertos y Controladores. Acceso Directo a Memoria (DMA) y Buffers. Interrupciones y Entorno de Programación.	Subprogramas como Recurso de Programación y como Estructuras de Control de Transferencia-Retorno. Funciones en Python. Parámetros Formales y Reales. Variables Globales) y Locales. Mecanismo de Invocación a Funciones. Reglas de Alcance. Ejemplo de programa desarrollado con funciones. Entrega AGF4 Propuesta Resolución Problemas de Parte Práctica Guía 7 Especificación Actividad Grupal Formativa N°5 AGF5			10 de mayo	1 a 7
<10>	Tipos de	Foro de Discusión Actividad Grupal			17 de mayo	1 a 7

Semana	Temas de teoría	Resolución de problemas	Laboratorio	Otro tipo	Fecha entrega Informe TP	Bibliografía básica
11/05 al 16/05	datos secuenciales: Listas. Agrupamiento de Datos en Arreglos (Vectores). Estructura de Control Repetitiva: for-in (ciclo definido). El método append().	Formativa N°5 AGF5 Carga de Arreglos. Almacenamiento de Arreglos. Búsqueda de Elementos en Arreglos. Arreglos como Parámetros. Propuesta Resolución Problemas de Parte Práctica Guía 8 Especificación Actividad Grupal Formativa Nº 6 AGF6 I Descubrimiento de algoritmos en Taller “Un Lugar para Algoritmiar”. Codificación de algoritmos en IDLE Python “Un Lugar para Programar”.				
<11> 18/05 al 23/05	Tipos de datos secuenciales: Listas. Agrupamiento de Datos en Arreglos (Matrices). Las funciones predefinidas range(), list(), split().	Foro de Discusión Actividad Grupal Formativa Nº6 AGF6 I Carga de Arreglos. Almacenamiento de Arreglos. Búsqueda de Elementos en Arreglos. Arreglos como Parámetros. Ordenamiento de Arreglos. Búsqueda Óptima de Elementos en Arreglos Ordenados. Operaciones Binarias con Arreglos Ordenados: Unión, Intersección y Diferencia. Propuesta Resolución Problemas de Parte Práctica Guía 8 Especificación Actividad Grupal Formativa Nº6 AGF6 II Descubrimiento de algoritmos en Taller “Un Lugar para Algoritmiar”. Codificación de algoritmos en IDLE Python “Un Lugar para Programar”. ENUNCIADO SEGUNDO TRABAJO GRUPAL FORMATIVO TGF2 Primera Versión			24 de mayo / 14 de junio	1 a 7
<12> 25/05 al 30/05	FERIADO NACIONAL	FERIADO NACIONAL			14 de junio	1 a 7
<13> 01/06 al 06/06	Organización de Dispositivos de Almacenamiento Masivo. Discos Magnéticos, Ópticos y Magnetoópticos. Confiabilidad de la Información Almacenada: Detección y Corrección de Errores. Propuesta Resolución Ejercicios de Parte Teórica Guía N° 9	Foro de Discusión Actividad Grupal Formativa Nº6 AGF6 II Archivos de Texto. Organización de Líneas y Caracteres de Control. Lectura de Secuencias de Datos desde un Archivo de Texto. Creación de Archivos de Texto y Escritura de Datos. Agregado de Datos a un Archivo de Texto Existente. ENUNCIADO SEGUNDO TRABAJO GRUPAL FORMATIVO TGF2 - Segunda Versión DESARROLLO SEGUNDO TRABAJO GRUPAL FORMATIVO TGF2. Descubrimiento de algoritmos en Taller “Un Lugar para Algoritmiar”. Codificación de algoritmos en IDLE Python “Un Lugar para Programar”.			14 de junio	1 a 7
<14> 08/06 al 13/06	REVISIÓN CONTENIDOS PRÁCTICOS DESARROLLO SEGUNDO	REVISIÓN CONTENIDOS PRÁCTICOS DESARROLLO SEGUNDO TRABAJO GRUPAL FORMATIVO TGF2 Descubrimiento de algoritmos en			14 de junio	1 a 7

Semana	Temas de teoría	Resolución de problemas	Laboratorio	Otro tipo	Fecha entrega Informe TP	Bibliografía básica
	TRABAJO GRUPAL FORMATIVO TGF2 Descubrimiento de algoritmos en Taller “Un Lugar para Algoritmiar”. Codificación de algoritmos en IDLE Python “Un Lugar para Programar”.	Taller “Un Lugar para Algoritmiar”. Codificación de algoritmos en IDLE Python “Un Lugar para Programar”. CLASE EXTRAORDINARIA OPTATIVA DE CONSULTA (PREVIA ENTREGA SEGUNDO TRABAJO GRUPAL FORMATIVO TGF2). DÍA Y HORARIO A ESTABLECER.				
<15> 15/06 al 20/06	FERIADO NACIONAL EVALUACIÓN SEGUNDO TRABAJO GRUPAL FORMATIVO TGF2 en LAB E (inscripción previa). CLASE EXTRAORDINARIA OPTATIVA DE CONSULTA (PREVIA SEGUNDO PARCIAL TEÓRICO). DÍA Y HORARIO A ESTABLECER.	FERIADO NACIONAL EVALUACIÓN SEGUNDO TRABAJO GRUPAL FORMATIVO TGF2 en LAB E (inscripción previa) CLASE EXTRAORDINARIA OPTATIVA DE CONSULTA (PREVIA SEGUNDO PARCIAL PRÁCTICO). DÍA Y HORARIO A ESTABLECER.				1 a 7
<16> 22/06 al 27/06	SEGUNDO PARCIAL TEÓRICO (en Laboratorio E) Evaluación de conocimientos contenidos teóricos (mediante prueba semiestructurada escrita de respuesta restringida).	SEGUNDO PARCIAL PRÁCTICO (en Laboratorio E) Evaluación de habilidades contenidos prácticos (sobre desarrollo en papel de un programa integrador en lenguaje Python).				1 a 7

CALENDARIO DE EVALUACIONES

Evaluación Parcial

Oportunidad	Semana	Fecha	Hora	Aula
1º	8	27/04	9:00	Lab E
2º	16	22/06	9:00	Lab E
3º	16	22/06	9:00	Lab E
4º				

Observaciones sobre el Temario de la Evaluación Parcial
En Esquema Temático Primer y Segundo Parcial en sitio del curso en campus FIUBA.

Otras observaciones
Recuperatorio Segundo Parcial Práctico: Semana 17 (29/06). 9:00. Lab E.