

	<b>PROGRAMACION I</b>		<b>3.4.071</b>
<b>Departamento al que pertenece</b>		<b>Director</b>	
<b>Tecnología Informática</b>		<b>Ing. Anibal Freijo</b>	
<b>Carga horaria</b>		<b>Fecha de aprobación en el Consejo de Facultad y N° de Acta</b>	
68 horas		16/06/2020 CF FAIN N° 416	
<b>Carrera(s) en la que se dicta</b>		<b>Código(s) Carrera(s)</b>	
Tecnicatura Universitaria En Desarrollo De Software Licenciatura En Gestión De Tecnología De La Información Ingeniería en Informática Licenciatura en Bioinformática		1111 13116 1608 5910	
<b>Código(s) Correlativa(s) Precedente(s)</b>	<b>Código(s) Correlativa(s) Subsiguiente(s)</b>	<b>Código(s) Carrera(s)</b>	
3.4.069	3.4.074 / 3.4.X18	1111 13116 1608 5910	
<b>Firmas</b>			
<b>Aprobación del Director de Departamento emisor.</b>		<b>Aprobación Decano(s)</b>	
 Ing. Anibal Freijo Director (a/o) Dpto. de Tecnología Informática Fundación UADE		 Ing. Juan R. Ingoin Decano Facultad de Ingeniería y Ciencias Exactas Fundación UADE	

## I. Fundamentos de la materia

Esta materia es una continuación de su correlativa precedente, Fundamentos de Informática. Completa los conceptos básicos de programación estructurada, impartidos en la materia anterior e incorpora los conceptos fundamentales de manejo dinámico de memoria, estructuras dinámicas de datos, y recursión. Estos conceptos son fundamentales, no sólo para un informático, sino también para comprender las siguientes materias del área de Programación dentro de la carrera. Se introduce al alumno en el uso de herramientas de desarrollo de software, que son imprescindibles en la profesión. Se incorpora también abundante práctica en el desarrollo de algoritmos.

## II. Objetivos

Al finalizar el curso el alumno será capaz de:

- Extender el concepto de arreglo a más de una dimensión.
- Resolver algoritmos con matrices.
- Conocer y aprender a utilizar los registros como estructura de datos y comprender la diferencia entre registros y arreglos.
- Comprender las características y diferencias entre memoria primaria y secundaria y la necesidad de almacenar datos en un medio no volátil.
- Aprender a manejar archivos de datos y sus distintas formas de acceso: secuencial y directa.
- Comprender las limitaciones en el uso de tipos de datos estáticos y la necesidad de trabajar con tipos de datos dinámicos.
- Conocer estructura de datos dinámicas y aprender a programar los algoritmos que las manipulan: búsqueda, inserción, eliminación.
- Aprender a codificar soluciones recursivas, con procedimientos y funciones, manejando adecuadamente los pasajes de parámetros.
- Saber identificar problemas recursivos.
- Conocer las estructuras dinámicas de datos más utilizadas, pudiendo identificar el tipo de dato más adecuado al problema.
- Comprender la importancia de la adecuada documentación y calidad del código
- Aprender a utilizar herramientas de desarrollo y de depuración en ambientes integrados.
- Incorporar la formación de una visión global de un programa de mediana complejidad.

## III. Contenidos

### **Contenidos mínimos**

Programación estructurada. Diseño Top-Down. Compilación: vinculación, ejecución y depuración de programas. Entornos de desarrollo. Tipos de datos y operadores. Precedencia. Constantes. Funciones. Declaraciones locales y globales. Sentencias de control de flujo. Entradas de datos. Pasaje de parámetros por valor y por referencia. Funciones de biblioteca. Resolución de algoritmos. Arreglos y punteros.

### **Contenidos conceptuales**

Unidad 1: Introducción a la materia.

Paradigmas de programación. Ambientes de desarrollo. Programa fuente, objeto y ejecutable. Bibliotecas de funciones. Compilación y vinculación. Familiarización con un Lenguaje de Programación. Tipos de datos. Operadores. Estructuras de control. Programas de ejemplo.

Unidad 2: Matrices

Matrices como arreglos de varias dimensiones. Índices y componentes. Recorridos. Relación entre el concepto informático y el matemático. Operaciones con matrices. Pasaje de arreglos como parámetro de funciones.

#### Unidad 3: Cadenas de Caracteres.

Implementación de cadenas de caracteres como arreglos delimitados. Operaciones: Copia y concatenación de cadenas. Determinación de su longitud. Uso de funciones de biblioteca para el tratamiento de caracteres y cadenas.

#### Unidad 4: Registros.

Tipos de datos estructurados estáticos. Registros como tipos de datos estructurados de componentes heterogéneos; su utilización. El acceso a los campos de datos. Registros vs. Arreglos.

#### Unidad 5: Archivos.

Características y diferencias de memoria primaria y secundaria. Archivos como almacenamiento de datos. Declaración, apertura, procesamiento y cierre. Acceso secuencial y directo. Algoritmos de búsqueda, agregado, inserción y eliminación en archivos. Consideraciones sobre la eficiencia en el uso de archivos. Similitudes y diferencias entre los tipos de archivos.

#### Unidad 6: Introducción al uso dinámico de memoria.

Limitaciones en el uso de memoria estática. Motivos para el uso dinámico de memoria. Apropiación y liberación de memoria. Estructuras de datos dinámicas. Algoritmos de búsqueda, inserción y eliminación sobre las estructuras.

#### Unidad 7: Recursión.

Definición de recursión. Características de los problemas recursivos. Estructura de una solución recursiva: rama explícita y rama recursiva. Iteraciones vs recursividad. Ventajas y desventajas. Seguimiento y depuración. Pasaje de parámetros en soluciones recursivas.

#### Unidad 8: Estructuras dinámicas de datos.

Aplicaciones de las estructuras dinámicas de datos. Representación gráfica.

### **Contenidos procedimentales**

- Familiarización con un Lenguaje de Programación.
- Familiarización con herramientas de desarrollo/depuración.
- Individualización de tipos de datos y operaciones asociadas.
- Identificación de la necesidad de usar archivos.
- Evaluación de distintas alternativas de solución (iterativa o recursiva) de un problema.
- Diseño de un programa estructurado incluyendo la definición de tipos de datos y manejo de almacenamiento persistente (archivos).
- 

### **Contenidos actitudinales**

- Concientización sobre la importancia de adoptar una estrategia de resolución adecuada en el caso de programas de mediana complejidad.
- Adquisición de una posición crítica para evaluar y elegir entre distintas soluciones a un mismo problema.
- Valoración de la calidad de un programa, medida en claridad, modularidad y reusabilidad.
- Proactividad para solucionar problemas de programación.

#### IV. Estrategias de enseñanza

##### Clases expositivas

Resolución de ejercicios en el pizarrón por parte del docente, fomentando:

- la participación de la clase
- la evaluación de soluciones alternativas, sopesando ventajas y desventajas de cada una.

Resolución individual y grupal de ejercicios con la supervisión del docente en los laboratorios.

Resolución de ejercicios fuera del aula, para su posterior discusión en clase.

#### V. Recursos

Pizarrón y/o PC con cañón en aula normal.

Aulas de informática con cañón y PCs para los alumnos.

Software: Entorno de desarrollo para el lenguaje usado

Se hará uso intensivo de computadoras para la implementación de lo aprendido. A los alumnos que dispongan de un compilador diferente al utilizado en las Aulas de Informática se les guiará en forma genérica para poder utilizarlos. Estos compiladores no serán instalados ni utilizados en las clases. Los alumnos que posean computadoras portátiles las podrán utilizar en las clases prácticas de laboratorio, pero no en las evaluaciones.

#### VI. Modalidad de Evaluación y condiciones de aprobación

Para aprobar la asignatura el alumno deberá cumplir con la aprobación de dos instancias: la cursada y el examen final. Requisitos académicos obligatorios para aprobar la cursada:

- a) Cumplir con un mínimo del 75% de asistencia a clase.
- b) Aprobar cada una de las evaluaciones parciales y/o recuperatorios previstos con una calificación mínima de 4 (cuatro) puntos.
- c) Aprobar los trabajos prácticos obligatorios (TPO) si los hubiere.

Quienes cumplan con todos los requisitos indicados, quedarán habilitados para rendir el examen final de la asignatura en los 11 (once) turnos de exámenes finales consecutivos posteriores a la aprobación de la cursada.

Se consignará como nota final el promedio simple entre la nota de aprobación de la cursada (promedio de las calificaciones en las evaluaciones parciales/TPs aprobados) y la calificación obtenida en el examen final regular.

En el caso que el alumno haya aprobado las instancias de evaluación y no requiera recuperar, podrá optar por rendir el examen final regular en la fecha prevista para el examen recuperatorio o bien en la fecha prevista para el examen final regular (una de las dos).

Los alumnos que rindan el examen final en la etapa de previos, la nota final a consignarse será exclusivamente la obtenida en dicha instancia de evaluación.

**NORMAS DE SEGURIDAD:** El trabajo en laboratorios y talleres debe llevarse a cabo respetando las normas de seguridad obligatorias. La aprobación de la cursada/materia estará sujeta al cumplimiento de las mismas, ya que son el principal factor de riesgo en las actividades de los alumnos, docentes, investigadores o técnicos. Utilizar siempre los Equipos de Protección Individual que se requiera (consultar procedimientos o protocolos de trabajo), por ejemplo protección ocular (anteojos gafas/pantallas faciales), guantes de vinilo y guardapolvo.

## VII. Bibliografía

### Básica

JOYANES Aguilar, Luis: Fundamentos de Programación. 4ta. Edición. Editorial Mc.Graw-Hill. ISBN 978-84-481-6111-8. EAN 9788448161118. Fecha de publicación: 24-03-2008

### Complementaria

PILGRIN, Mark. Inmersión en Python 3 [consultas: 19/4/2018] Traducido por José Miguel González Aguilera. Disponible gratuitamente bajo licencia Creative Commons 3.0 en <http://www.jmgaguilera.com/libro/python/traducci%C3%B3n/latex/2016/08/19/inmersion-python.html> y en <https://openlibra.com/es/book/inmersion-en-python-3>

MARZAL VARÓ, Andres; GRACIA LUENGO, Isabel; GARCÍA SEVILLA, Pedro. Introducción a la programación con Python 3 [En línea]. Disponible gratuitamente bajo licencia Creative Commons en <https://openlibra.com/es/book/introduccion-a-la-programacion-con-python-3>. [consultas: 19/4/2018]

Van Rossum, Guido. El tutorial de Python. Traducido y empaquetado por la comunidad de Python Argentina. [consultas: 19/4/2018] Español: [docs.python.org.ar/tutorial/pdfs/TutorialPython3.pdf](https://docs.python.org.ar/tutorial/pdfs/TutorialPython3.pdf).

[En línea]. The Python Language Reference. [consulta: 19/4/2018] Español: <https://docs.python.org/3/reference/>

## VIII. Cronograma

Clase	Actividad/contenido
1	Introducción al lenguaje Python. Variables, operadores, estructuras de control. Instrucciones break y continue. Uso del entorno de desarrollo. Ejercitación.
2	Funciones. Estructura. Parámetros. Parámetros mutables e inmutables. Valores por omisión. Ámbito de las variables. Funciones Lambda. Módulos y paquetes.
3	Listas. Acceso por subíndice. Desempaquetado. Operaciones. Métodos. Números al azar.
4	Listas. Rebanadas. Concepto de iterables. Instrucción for. Listas por comprensión.
5	Matrices. Implementación en Python.
6	Cadenas de caracteres. Operaciones. Métodos. Conversión de números a cadenas.
7	Simulacro de parcial.
8	1er Parcial
9	Cadenas de caracteres. Métodos. Formateo de cadenas.
10	Excepciones. Concepto. Cláusulas try, except y finally. Instrucción raise.

11	Archivos. Concepto y clasificación. Archivos de texto. Archivos separados por comas (CSV).
12	Recursividad. Iteraciones vs. Recursividad. Ventajas y desventajas. Simulacro de parcial
13	Tuplas, conjuntos y diccionarios. Concepto. Aplicaciones. Operaciones.
14	Simulacro de parcial.
15	2doParcial
16	Repaso
17	Recuperatorio y Final Regular. Ejercitación
	Final Regular