MODALIDAD ACADÉMICA

Asignatura	**Algoritmos y Estructuras de Datos**	
Carrera	**INGENIERÍA EN SISTEMAS DE INFORMACIÓN**	
Ciclo Lectivo	*2018*	
Vigencia del programa	*Desde el ciclo lectivo 2015*	
Plan	*2008*	
Nivel	■ 1er. Nivel ☐ 2do. Nivel ☐ 3er. Nivel ☐ 4to. Nivel ☐ 5to. Nivel	
Coordinador de la Cátedra	Ing. Valerio Frittelli	
Área de Conocimiento	■ Programación ☐ Computación ☐ Sistemas de Información ☐ Gestión Ingenieril ☐ Modelos ☐ Complementaria	
Carga horaria semanal	5 horas	
Anual/ cuatrimestral	Anual	
Contenidos Mínimos, según Diseño Curricular-Ordenanza 1150 (sólo para asignaturas curriculares, no electivas)	**Contenidos Mínimos:** La Ordenanza 1150 de Rectorado que reglamenta el Diseño Curricular de la Carrera, establece un cierto conjunto de *contenidos mínimos* para cada asignatura del Plan. Específicamente, para la asignatura AED esos contenidos mínimos son: [1] Concepto de Dato. [2] Tipos de Datos Simples. [3] Estructuras de Control Básicas: Secuencial, Condicional, Cíclica. [4] Tipos Abstractos de Datos. [5] Abstracciones con Procedimientos y Funciones. [6] Pasaje de Parámetros. [7] Estructuras de Datos: Registros, Arreglos y Archivos. [8] Estructuras de Datos Lineales (Pilas – Colas). [9] Recursividad. [10] Estrategias de Resolución *[de Problemas]*. [11] Algoritmos de Búsqueda, Recorrido y Ordenamiento. [12] Archivos de Acceso Secuencial y Aleatorio: Organizaciones y Accesos. [13] Procesamiento Básico *[de Archivos]*. [14] Noción de Orden de Complejidad. [15] Noción de Complejidad Computacional.	
Correlativas para cursar (según Diseño Curricular-Ordenanza 1150)	Regulares • Ninguna (salvo curso de ingreso)	Regulares • Ninguna (salvo curso de ingreso)
Correlativas para rendir (según Diseño Curricular-Ordenanza 1150)	Regulares • Ninguna (salvo curso de ingreso)	Regulares • Ninguna (salvo curso de ingreso)

Objetivos generales de la Asignatura	**a.) Objetivos Generales para la Asignatura:** Tal cual los prescribe la *Ordenanza 1150* de Rectorado, los *objetivos generales* para AED son en realidad *expectativas de logro* para los alumnos (es decir, se plantean como metas que deberían ser capaces de alcanzar los propios alumnos al terminar el cursado). Literalmente, son los siguientes: ● Identificar problemas algorítmicos. ● Conocer el proceso de diseño e implementación de software. ● Aplicar las herramientas fundamentales representativas de los procesos, integrando la sintaxis elemental de un lenguaje de programación en el laboratorio asociado.

Programa Analítico

Propuesta de Programa Analítico: Se propone el siguiente esquema de unidades para el nuevo programa analítico (vjgente desde el ciclo 2015). Los números que figuran entre corchetes hacen referencia a los contenidos mínimos previstos por la Ordenanza 1150, que están siendo efectivamente cubiertos por el ítem nombrado y resaltado en *cursiva*:

Unidad Nro. 1: Fundamentos de programación – Estructuras secuenciales.

Resultados de Aprendizaje:

- Manifestar capacidad de razonamiento para resolver problemas simples de modo de llegar a identificar datos, procesos posibles y resultados a obtener, en el contexto de un avance sólido en el resto de los contenidos de esta misma asignatura.
- Demostrar conocimiento de las instrucciones simples más elementales de un lenguaje de programación de propósito general, para poder plantear programas completos en base a los algoritmos propuestos, teniendo en cuenta que esta habilidad facilite la puesta en práctica de los conceptos y técnicas aprendidas.

Contenidos:

- Concepto de algoritmo, computadora (o máquina algorítmica) y programa. Memoria, celdas de memoria y representación de información en base al sistema binario.
- Lenguajes de programación. Descarga, instalación y configuración del kit de herramientas (SDK) de Python y de un entorno de desarrollo (o IDE).
- *Concepto de dato* [1] y tipo de datos. *Tipos de datos simples* [2] en Python. Variables.
- Instrucciones básicas: asignación, carga por teclado y visualización por consola estándar. Operadores aritméticos. Expresiones aritméticas (o numéricas). Tipos estructurados elementales en Python: secuencias de datos y cadenas de caracteres.
- *Estructuras de control de flujo básicas: la estructura secuencial de instrucciones* [3]. Concepto de script en Python. Estructura de un script, edición, depuración y ejecución a través de la consola de comandos de Python y a través de un IDE. Reglas y convenciones de escritura de código fuente del Lenguaje Python. Uso de comentarios en un script Python.
- Pasos en la resolución de problemas y estructura de un algoritmo. Técnicas de representación de algoritmos: pseudocódigo y diagrama de flujo.
- Definición de problema simple. Planteo y resolución de problemas simples en base a *secuencias de instrucciones simples* [3].
- Problemas típicos: aplicaciones matemáticas (conversión de unidades de tiempo, cálculo de promedios y porcentajes directos, aplicaciones del operador resto de una división) o de gestión de cadenas de

caracteres (a partir de invocación a métodos nativos).

Bibliografía Obligatoria:

a. Frittelli, V., et al. (2016). Publicaciones Cátedra de AED. *Fichas de Estudio para AED (disponible en Aula Virtual AED)*. Cordoba: UTN – Facultad Regional Córdoba.

b. Python Software Foundation. (2015). *Python Documentation*. Retrieved February 24, 2016, from https://docs.python.org/3/.

Bibliografía Complementaria:

a. Pilgrim, M. (2004). *Dive Into Python - Python from novice to pro*. Retrieved March 6, 2016, from http://www.diveintopython.net/toc/index.html.

Evaluación:

Contenidos y prácticas evaluados a través de los Cuestionarios Teóricos 1, 2 y 3 y el Desafío de Programación 1 disponibles en el Aula Virtual.

Unidad Nro. 2: Estructuras condicionales - Subproblemas.

Resultados de Aprendizaje:

- Mostrar capacidad de razonamiento para resolver problemas de estructura condicional, de modo de llegar a plantear las distintas alternativas lógicas en que puede dividirse un algoritmo, en el contexto de un avance sólido en el resto de los contenidos de esta misma asignatura.
- Usar en forma correcta las instrucciones condicionales provistas por un lenguaje de programación de propósito general, para poder plantear programas cuya lógica abarque diversas alternativas y situaciones, teniendo en cuenta que esta habilidad facilite la puesta en práctica de los conceptos y técnicas aprendidas.

Contenidos:

- Estructuras de control de flujo básicas: la instrucción condicional [3]. Instrucción condicional simple, doble y múltiple en Python. Condiciones anidadas.
- Operadores relacionales y lógicos. Expresiones y proposiciones lógicas (o booleanas). Tablas de verdad.
- Planteo de problemas simples en base a instrucciones condicionales [3]. Concepto de Subproblema. Problemas compuestos (divisibles en subproblemas). Descomposición de un problema compuesto en subproblemas simples. Planteo y resolución de problemas compuestos en base a instrucciones condicionales [3].
- Problemas típicos: determinación del menor o el mayor de una secuencia fija de números de entrada (dos, tres o cuatro números), filtrado de valores de entrada para evitar operaciones matemáticas no definidas, cálculo de raíces de una ecuación lineal o cuadrática, cálculo del valor de un polinomio.

Bibliografía Obligatoria:

a. Frittelli, V., et al. (2016). Publicaciones Cátedra de AED. *Fichas de Estudio para AED (disponible en Aula Virtual AED)*. Cordoba: UTN – Facultad Regional Córdoba.

b. Python Software Foundation. (2015). *Python Documentation*. Retrieved February 24, 2016, from https://docs.python.org/3/.

Bibliografía Complementaria:

a. Pilgrim, M. (2004). *Dive Into Python - Python from novice to pro*. Retrieved March 6, 2016, from http://www.diveintopython.net/toc/index.html.

Evaluación:

Contenidos y prácticas evaluados a través de los Cuestionarios Teóricos 3, 4 y 5 y el Trabajo Práctico 1 disponibles en el Aula Virtual.

Unidad Nro. 3: Estructuras repetitivas.

Resultados de Aprendizaje:

- Mostrar capacidad de razonamiento para resolver problemas de naturaleza repetitiva, de modo de llegar a plantear una solución en base a repetir en forma sistemática la ejecución de un proceso, en el contexto de un avance sólido en el resto de los contenidos de esta misma asignatura.
- Aplicar en forma correcta los distintos tipos de instrucciones repetitivas provistas por un lenguaje de programación de propósito general, para poder plantear programas cuya lógica se base en repetición de procesos, teniendo en cuenta que esta habilidad facilite la puesta en práctica de los conceptos y técnicas aprendidas.

Contenidos:

- Concepto de proceso repetitivo. Estructuras de control de flujo básicas: la estructura cíclica o repetitiva [3]. Tipos generales de ciclos: el ciclo 0-N y el ciclo 1-N. Los ciclos while y for en Python.
- Ciclos infinitos. Interrupción de un ciclo. Ciclos anidados.
- Variables de conteo y de acumulación. Variables centinela (o banderas). Uso de banderas en la condición de un ciclo.
- Noción de conteo de operaciones críticas y comparación de tiempos esperados de ejecución.
- Planteo y resolución de problemas de naturaleza repetitiva básica usando instrucciones cíclicas [3].
- Problemas típicos: carga y procesamiento de sucesiones de datos desde la consola estándar, conociendo la cantidad de datos y/o mediante un proceso de carga por doble lectura, determinación de promedios y porcentajes, determinación del mayor y/o menor de una sucesión de valores, problemas generales de búsqueda de un valor único o de múltiples valores, conteo y acumulación de cantidades, generación y procesamiento de sucesiones de números a partir de un valor inicial dado, generación de números aleatorios y su aplicación en juegos y simulaciones simples, procesamiento básico de caracteres y cadenas (aplicación de métodos nativos para cadenas de caracteres de Python).

Bibliografía Obligatoria:

a. Frittelli, V., et al. (2016). Publicaciones Cátedra de AED. *Fichas de Estudio para AED (disponible en Aula Virtual AED)*. Cordoba: UTN – Facultad Regional Córdoba.

b. Python Software Foundation. (2015). *Python Documentation*. Retrieved February 24, 2016, from https://docs.python.org/3/.

Bibliografía Complementaria:

a. Pilgrim, M. (2004). *Dive Into Python - Python from novice to pro*. Retrieved March 6, 2016, from http://www.diveintopython.net/toc/index.html.

Evaluación:

Contenidos y prácticas evaluados a través de los Cuestionarios Teóricos 7, 8 y 9 y el Trabajo Práctico 2 disponibles en el Aula Virtual.

Unidad Nro. 4: Funciones – Programación Estructurada.

Resultados de Aprendizaje:

- Manifestar habilidad para identificar procesos que podrían detallarse por separado en la solución de un problema, de modo de poder reusar esos procesos en las soluciones para otros problemas, en el contexto de un avance sólido en el resto de los contenidos de esta misma asignatura, como en asignaturas

similares de la carrera y en el futuro desarrollo profesional.

- Mostrar capacidad de razonamiento para resolver problemas de naturaleza recursiva, de forma de llegar a plantear soluciones en las que un proceso recurra a si mismo para arribar al resultado, considerando que esta habilidad facilitará el dominio de otras técnicas especiales de planteo de algoritmos.
- Desarrollar procesos en forma de funciones propias en un lenguaje de programación de propósito general, para poder plantear programas que reutilicen soluciones ya aplicadas, teniendo en cuenta que esta habilidad facilite la puesta en práctica de los conceptos y técnicas aprendidas.

Contenidos:

- Concepto elemental de subrutina para plantear un subproblema: introducción al uso de funciones.
- Funciones en Python: cabecera y estructura general. Funciones con retorno de valor y funciones sin retorno de valor. Pasaje de parámetros a una función [6]. Parámetros actuales y parámetros formales.
- Formas especiales de parametrización en Python. Concepto general/teórico de parametrización por valor y por referencia y el modelo en Python: parametrizacción por valor.
- Ámbito de una variable: variables locales y variables globales a un bloque de acciones.
- Concepto de Programación Estructurada: planteo de un programa en forma modular usando funciones.
- Generalización de código fuente: planteo de funciones parametrizadas de uso general. Reutilización de código.
- Módulos y paquetes de módulos en Python. Módulos de la librería estándar más comunes en Python. Importación de módulos. Creación de módulos y paquetes del programador.
- Recursividad [9]. Funciones recursivas bien planteadas. Determinación de la condición de corte de la recursividad. Seguimiento de la recursividad. Recursividad mutua o indirecta. Análisis básico de la eficiencia y la aplicabilidad de la recursividad en situaciones prácticas comunes.

Bibliografía Obligatoria:

a. Frittelli, V., et al. (2016). Publicaciones Cátedra de AED. *Fichas de Estudio para AED (disponible en Aula Virtual AED)*. Cordoba: UTN – Facultad Regional Córdoba.

b. Python Software Foundation. (2015). *Python Documentation*. Retrieved February 24, 2016, from https://docs.python.org/3/.

Bibliografía Complementaria:

a. Pilgrim, M. (2004). *Dive Into Python - Python from novice to pro*. Retrieved March 6, 2016, from http://www.diveintopython.net/toc/index.html.

Evaluación:

Contenidos y prácticas evaluados a través de los Cuestionarios Teóricos 10, 11, 12 y 13 y el Desafío de Programación 2 disponibles en el Aula Virtual.

Unidad Nro. 5: Arreglos.

Resultados de Aprendizaje:

- Mostrar capacidad para detectar situaciones de procesamiento de grandes volúmenes de datos con acceso directo, de modo de usar en forma correcta estructuras de datos con acceso mediante índices, en el contexto de un avance sólido en asignaturas similares de la carrera y en el futuro desarrollo profesional.
- Diseñar algoritmos basados en arreglos para problemas que impliquen procesamiento de grandes volúmenes de datos, para poder plantear soluciones a problemas básicos de ingeniería, considerando que

esa habilidad de diseño consolide en cada alumno la capacidad de razonamiento, acreciente la confianza en sí mismo, y fomente su independencia de criterio.

Contenidos:

- Concepto de tipo compuesto o estructurado (estructuras de datos vs. variables de tipo simple). Estructuras de datos [7]: definición general y necesidades de uso.
- Arreglos [7] (variables subindicadas): concepto de acceso directo y acceso secuencial. Dimensión de un arreglo. Índices. Implementación del concepto de arreglo de tamaño dinámico mediante listas nativas en Python.
- Arreglos unidimensionales (o "vectores"): creación y asignación de valores. Recorrido secuencial de un vector [11]. Carga y visualización.
- Ordenamiento: clasificación tradicional en métodos directos (o simples) y compuestos (o mejorados). Algoritmos directos (Bubble sort, Selection sort e Insertion sort) [11].
- Algoritmos de Búsqueda Secuencial y Búsqueda Binaria [11].
- Fusión (o mezcla) de vectores ordenados.
- Elementos básicos de análisis de eficiencia mediante conteo de comparaciones.
- Vectores de conteo y vectores de acumulación.
- Vectores paralelos.
- Arreglos bidimensionales (o "matrices") y multidimensionales. Conceptos y aplicaciones. Creación y asignación de valores. Recorrido de matrices en orden de filas y en orden de columnas [11]. Sumarización por filas y columnas.

Bibliografía Obligatoria:

a. Frittelli, V., et al. (2016). Publicaciones Cátedra de AED. *Fichas de Estudio para AED (disponible en Aula Virtual AED)*. Cordoba: UTN – Facultad Regional Córdoba.

b. Python Software Foundation. (2015). *Python Documentation*. Retrieved February 24, 2016, from https://docs.python.org/3/.

Bibliografía Complementaria:

a. Pilgrim, M. (2004). *Dive Into Python - Python from novice to pro*. Retrieved March 6, 2016, from http://www.diveintopython.net/toc/index.html.

Evaluación:

Contenidos y prácticas evaluados a través de los Cuestionarios Teóricos 15, 16, 17 y 18 más el Desafío de Programación 3 y el Trabajo Práctico 3 disponibles en el Aula Virtual.

Unidad Nro. 6: Registros.

Resultados de Aprendizaje:

- Mostrar capacidad para detectar situaciones de procesamiento de grandes volúmenes de datos con datos etiquetados por nombres, de modo de usar en forma correcta estructuras de datos con acceso mediante campos, en el contexto de un avance sólido en asignaturas similares de la carrera y en el futuro desarrollo profesional.
- Diseñar algoritmos basados en registros para problemas que impliquen procesamiento de grandes volúmenes datos identificados por nombres, para poder plantear soluciones a problemas básicos de ingeniería, considerando que esa habilidad de diseño consolide en cada alumno la capacidad de razonamiento, acreciente la confianza en sí mismo, y fomente su independencia de criterio.

- Demostrar capacidad de combinar soluciones basadas en arreglos con soluciones basadas en registros, para ampliar el campo de posibles problemas de ingeniería a resolver, considerando que esa habilidad de diseño consolide en cada alumno la capacidad de razonamiento, acreciente la confianza en sí mismo, y fomente su independencia de criterio.

Contenidos:

- Registros [7]: agrupamientos de variables de tipos diferentes. Definición y forma de creación de registros en Python. Campos de un registro. Acceso a campos.
- Concepto de abstracción de datos y empleo de registros para modelar esa abstracción. Tipos nativos de datos y tipos abstractos de datos [4]. Concepto de abstracción funcional y empleo de funciones en Python para modelar esa abstracción [5].
- Uso combinado de arreglos y registros: vectores y matrices de registros.
- Problemas típicos: implementación de tipos abstractos que representen números complejos, puntos en un plano, segmentos de recta, fechas, entidades del dominio para sistemas de gestión administrativa (como clientes, cuentas, artículos, libros y otras entidades).

Bibliografía Obligatoria:

a. Frittelli, V., et al. (2016). Publicaciones Cátedra de AED. *Fichas de Estudio para AED (disponible en Aula Virtual AED)*. Cordoba: UTN – Facultad Regional Córdoba.

b. Python Software Foundation. (2015). *Python Documentation*. Retrieved February 24, 2016, from https://docs.python.org/3/.

Bibliografía Complementaria:

a. Pilgrim, M. (2004). *Dive Into Python - Python from novice to pro*. Retrieved March 6, 2016, from http://www.diveintopython.net/toc/index.html.

Evaluación:

Contenidos y prácticas evaluados a través del Cuestionario Teórico 14 más el Desafío de Programación 3 y el Trabajo Práctico 3 disponibles en el Aula Virtual.

Unidad Nro. 7: Estructuras lineales.

Resultados de Aprendizaje:

- Mostrar habilidad para detectar situaciones de procesamiento de datos en diferentes modalidades, de modo de diseñar estructuras de datos que se adecuen en forma eficiente a cada modalidad, en el contexto de un avance sólido en asignaturas similares de la carrera y en el futuro desarrollo profesional.
- Demostrar capacidad para implementar estructuras de datos abstractas, para ampliar el dominio del lenguaje de programación empleado, considerando que esa capacidad consolide en cada alumno su habilidad de resolución de problemas.

Contenidos:

- Estructuras de datos nativas y estructuras de datos abstractas [4].
- Estructuras de datos lineales [8] y no lineales.
- Estructuras lineales básicas: concepto general de pilas, cola [8] y lista.
- Análisis de la implementación general de listas nativas ya provista por Python, en base a un arreglo de crecimiento dinámico como soporte.
- Implementación de pilas y colas usando registros, arreglos/listas y funciones de Python como soporte para modelar la abstracción de datos y la abstracción funcional [5].

- Aplicación de Pilas en problemas de control de simetría, control de retornos, e inversión de secuencias.
- Aplicación de colas en problemas de espera para acceso a puestos de servicio.
- Descripción de las estructuras predefinidas de Python para manejo de estructuras o colecciones abstractas (listas usadas como pilas o colas, uso general de listas nativas).

Bibliografía Obligatoria:

a. Frittelli, V., et al. (2016). Publicaciones Cátedra de AED. *Fichas de Estudio para AED (disponible en Aula Virtual AED)*. Cordoba: UTN – Facultad Regional Córdoba.

b. Python Software Foundation. (2015). *Python Documentation*. Retrieved February 24, 2016, from https://docs.python.org/3/.

Bibliografía Complementaria:

a. Pilgrim, M. (2004). *Dive Into Python - Python from novice to pro*. Retrieved March 6, 2016, from http://www.diveintopython.net/toc/index.html.

Evaluación:

Contenidos y prácticas evaluados a través de los Cuestionarios Teóricos 19 y 20 disponibles en el Aula Virtual.

Unidad Nro. 8: Elementos de análisis de algoritmos.

Resultados de Aprendizaje:

- Analizar en forma esencial la eficiencia de un algoritmo, ya sea en términos de tiempo de ejecución, consumo de memoria o complejidad aparente de código fuente, para poder plantear soluciones profesionales y eficientes a diversos problemas del campo de la ingeniería, teniendo en cuenta que esa capacidad de análisis les permita diferenciarse como programadores y desarrolladores, además de convertirse en referentes sólidos y confiables.

Contenidos:

- Motivaciones para el análisis formal de algoritmos.
- Elementos de análisis de eficiencia de un algoritmo: tiempo de ejecución, espacio de memoria empleado y complejidad del código fuente.
- Concepto de comportamiento asintótico. Formalización del concepto de orden de complejidad [14]. Notación O mayúscula (o Big O) para expresar el orden de un algoritmo.
- Clasificación básica de las principales funciones asociadas al concepto de orden de complejidad.
- Propiedades de la relación de orden y aplicaciones prácticas.
- Aplicaciones: Análisis de tiempo de ejecución de los algoritmos de ordenamiento simples [11][14]. El ordenamiento de Shell como algoritmo mejorado de la inserción simple [11][14]. Comparación del comportamiento asintótico de los tiempos de ejecución. Análisis comparativo entre la búsqueda secuencial y la búsqueda binaria [11][14].

Bibliografía Obligatoria:

a. Frittelli, V., et al. (2016). Publicaciones Cátedra de AED. *Fichas de Estudio para AED (disponible en Aula Virtual AED)*. Cordoba: UTN – Facultad Regional Córdoba.

b. Python Software Foundation. (2015). *Python Documentation*. Retrieved February 24, 2016, from https://docs.python.org/3/.

Bibliografía Complementaria:

a. Pilgrim, M. (2004). *Dive Into Python - Python from novice to pro*. Retrieved March 6, 2016, from http://www.diveintopython.net/toc/index.html.

Evaluación:
Contenidos y prácticas evaluados a través de los Cuestionarios Teóricos 20 y siguientes, los Desafíos de Programación 3 y 4 y el Trabajo Práctico 4 disponibles en el Aula Virtual.

Unidad Nro. 9: Archivos.

Resultados de Aprendizaje:

- Mostrar capacidad para detectar situaciones que requieran almacenamiento externo de datos y resultados, de modo de usar en forma correcta estructuras de datos externas, en el contexto de un avance sólido en asignaturas similares de la carrera y en el futuro desarrollo profesional.
- Diseñar programas basados en archivos para problemas que impliquen almacenamiento externo, para poder plantear soluciones a problemas típicos de ingeniería, considerando que esa habilidad de diseño consolide en cada alumno su futura capacidad profesional.

Contenidos:

- El problema de la volatilidad de los datos en memoria principal. Concepto y necesidad de medios de almacenamiento externo.
- Archivos [7]. Archivos de texto y binarios. Archivos de registros.
- Apertura. Modos de apertura. Lectura. Escritura. Verificación de final de archivo. Reposicionamiento.
- Formas de organización y accesos: archivos de acceso secuencial y archivos de acceso aleatorio [12].
- Elementos de procesamiento básico de archivos de registros (y archivos binarios en general) [13]: altas, bajas (lógicas y físicas) y modificaciones (procesamiento ABM), listados de contenido (completos o filtrados), búsqueda secuencial, consultas.
- Procesamiento básico de archivos de texto [13]: archivos de texto como soporte de lotes de datos secuenciales para su uso en problemas de procesamiento masivo de datos. Creación, carga, recorrido, conversión de contenidos (de tipo texto a tipo numérico) y eventual creación de arreglos a partir de esos contenidos.

Bibliografía Obligatoria:

a. Frittelli, V., et al. (2016). Publicaciones Cátedra de AED. *Fichas de Estudio para AED (disponible en Aula Virtual AED)*. Cordoba: UTN – Facultad Regional Córdoba.
b. Python Software Foundation. (2015). *Python Documentation*. Retrieved February 24, 2016, from https://docs.python.org/3/.

Bibliografía Complementaria:

a. Pilgrim, M. (2004). *Dive Into Python - Python from novice to pro*. Retrieved March 6, 2016, from http://www.diveintopython.net/toc/index.html.

Evaluación:
Contenidos y prácticas evaluados a través de los Cuestionarios Teóricos 21 al 24, el Desafíos de Programación 4 y el Trabajo Práctico 4 disponibles en el Aula Virtual.

Unidad Nro. 10: Estrategias de resolución de problemas.

Resultados de Aprendizaje:

- Diseñar algoritmos basados en diversas estragegias de resolución para los problemas estudiados, de manera de poder plantear soluciones a problemas básicos de ingeniería, considerando que esa habilidad de diseño consolide en cada alumno la capacidad de razonamiento, acreciente la confianza en sí mismo, y

fomente su independencia de criterio.

Contenidos:

- Motivación y antecedentes para el estudio de estrategias de resolución de problemas (o de planteo de algoritmos) [10].
- Clasificación y concepto básico de las estrategias existentes más comunes: fuerza bruta, recursividad, backtraking, divide y vencerás, randomización, algoritmos ávidos y programación dinámica.
- Revisión de planteos algorítmicos conocidos basados en Fuerza Bruta.
- Recursividad [9] y Backtraking: planteo de algoritmos basados en aprovechar el proceso de vuelta atrás de la recursión para analizar soluciones parciales a un problema. Casos típicos: problema de las Ocho Reinas, problema de las Torres de Hanoi.
- Recursividad [9] y Divide y Vencerás: estructura general de la estrategia. Aplicaciones: los algoritmos de ordenamiento Merge sort y Quick sort [11]. Análisis general de eficiencia en tiempo de ejecución, ocupación de memoria y complejidad de código fuente [14].

Bibliografía Obligatoria:

a. Frittelli, V., et al. (2016). Publicaciones Cátedra de AED. *Fichas de Estudio para AED (disponible en Aula Virtual AED)*. Cordoba: UTN – Facultad Regional Córdoba.

b. Python Software Foundation. (2015). *Python Documentation*. Retrieved February 24, 2016, from https://docs.python.org/3/.

Bibliografía Complementaria:

a. Pilgrim, M. (2004). *Dive Into Python - Python from novice to pro*. Retrieved March 6, 2016, from http://www.diveintopython.net/toc/index.html.

Evaluación:

Contenidos y prácticas evaluados a través de los Cuestionarios Teóricos 25 y 26 y el Trabajo Práctico 4 disponibles en el Aula Virtual.

Unidad Nro. 11: Introducción a la Teoría de la Complejidad Computacional.

Resultados de Aprendizaje:

- Demostrar conocimientos básicos en el campo de la Teoría de la Complejidad, para poder distinguir en forma esencial entre problemas tratables e intratables, considerando que esta habilidad oriente a los estudiantes en el aprendizaje futuro de nuevas técnicas para enfrentar problemas en el límite de la computabilidad.

Contenidos:

- Motivación y antecedentes para el estudio de la Teoría de la Complejidad Computacional [15].
- Algoritmos de tiempo de ejecución de orden polinomial. Algoritmos intratables (tiempo de ejecución de orden exponencial o factorial).
- Reducción de un problema a otro. Reducción polinómica.
- Nociones de máquina algorítmica determinista y de máquina algorítmica no determinista.
- Problemas de decisión y problemas de optimización.
- Clases de Complejidad. Las clases P, NP y NP Completo. Propiedades de un problema NP Completo.
- Aproximación conceptual al Teorema de Cook-Levin.
- Los problemas del milenio y la cuestión P vs. NP en la actualidad.

Bibliografía Obligatoria:

a. Frittelli, V., et al. (2016). Publicaciones Cátedra de AED. *Fichas de Estudio para AED (disponible en Aula Virtual AED)*. Cordoba: UTN – Facultad Regional Córdoba.

b. Python Software Foundation. (2015). *Python Documentation*. Retrieved February 24, 2016, from https://docs.python.org/3/.

Bibliografía Complementaria:

a. Pilgrim, M. (2004). *Dive Into Python - Python from novice to pro*. Retrieved March 6, 2016, from http://www.diveintopython.net/toc/index.html.

Evaluación:

Contenidos y prácticas evaluados obligatoriamente en el Examen Final de la asignatura.

Metodología de enseñanza y aprendizaje
(Planificar estrategias centradas en el aprendizaje activo del estudiante)

- ✓ En cada clase se desarrolla un tema central, y el mismo se ejemplifica y analiza con modelos de programas presentados por los profesores.
- ✓ Los alumnos realizan modificaciones y variantes sobre los modelos, y realizan ejercicios nuevos en base a los temas tratados.
- ✓ Los trabajos prácticos y las tareas semanales (cuestionarios y desafíos) integran los conocimientos de las distintas unidades, y parte del tiempo de clase se usa para analizar dudas y elementos relevantes referidos a esos trabajos.
- ✓ Todos los materiales del curso (materiales del estudio y evaluaciones de tipos diversos) están siempre disponibles en el aula virtual de la asignatura. El esquema de trabajo al que se orienta toda la Cátedra (desde 2015 en adelante) es el de *Clase Invertida*, con variaciones que son propias del hecho de contar con un muy elevado número de estudiantes.

Sistema de evaluación
(Nombrar y describir cada una de las diferentes instancias de evaluación, pensando en la Evaluación como proceso continuo de recolección de evidencias)

1. Se aplica un esquema de regularidad ***común a todos los cursos***, y como resultado del mismo un estudiante puede quedar en alguna de las siguientes condiciones al finalizar el cursado:

Condición	Efecto
Aprobación Directa	El estudiante aprueba la asignatura sin tener que rendir el examen final. El estudiante sólo debe inscribirse para cualquier turno de examen (como si fuese a rendir) y simplemente pasar ese día para que se le firme la libreta y se registre su nota final en el acta.
Promocionado	El estudiante rinde el examen final, pero sólo la parte teórica (no rinde parte práctica).
Regular	El estudiante rinde examen final completo: debe rendir primero la parte práctica, y en caso de aprobarla continúa rindiendo la parte teórica.
Libre	El estudiante no llegó a cumplir los requisitos mínimos para regularizar, y debe recursar la materia. No puede rendir examen final. Se considera que un estudiante quedó *Libre* si desarrolló todas las *evaluaciones principales*

	previstas por la Cátedra (esto es: los cuatro parciales y los cuatro trabajos prácticos), y aún así no llegó a las notas y promedios mínimos requeridos para al menos regularizar.
Abandonó	Equivale a la condición de *Libre*, pero se registra en este estado a los estudiantes que quedaron libres y no desarrollaron todas las *evaluaciones principales* previstas por la Cátedra.

2. En todos los cursos y sin excepción, para determinar la *condición o situación final* de un estudiante al terminar el cursado, se tienen en cuenta las siguientes evaluaciones:

   - ✓ Cuatro *Parciales de Regularidad* (*P1*, *P2*, *P3* y *P4*) a desarrollar en el laboratorio. Los dos primeros (*P1* y *P2*) se evalúan en el primer semestre de cursado, y los dos últimos (*P3* y *P4*) se evalúan en el segundo semestre. A partir del ciclo 2019, está previsto además ***UN*** *Parcial Recuperatorio* (que será evaluado en la última semana de clases) si al finalizar el ciclo lectivo algún estudiante necesitase hacerlo. Si un estudiante hace el *PR*, la nota que obtenga en el mismo *reemplazará* a la nota (o al ausente) que originalmente tuviese en ***UNO (Y SÓLO UNO)*** de los cuatro *Parciales de Regularidad* originales. Por esta razón, el *PR* sirve no solamente para cubrir un aplazo o un ausente en ***UN*** Parcial, sino también para levantar ***UNA*** nota de parcial si el estudiante necesita hacerlo para aspirar a la condición de Aprobación Directa (los detalles se explican más adelante).
   - ✓ Un *Coloquio Teórico Final* (que designamos como *Parcial 5* o *P5*) que sólo realizan los estudiantes que estén en condiciones de lograr la condición de Aprobación Directa, según se explica más adelante en este mismo documento). El *P5* no tiene recuperatorio.
   - ✓ Cuatro *Trabajos Prácticos* (*TP1*, *TP2*, *TP3* y *TP4*) a desarrollar en grupos. A partir del ciclo 2019, está previsto que se pueda desarrollar y entregar ***UN*** *Trabajo Práctico Recuperatorio* (*TPR*) (que será evaluado en la última semana de clases) si al finalizar el ciclo lectivo algún estudiante necesitase hacerlo. Si un estudiante hace y entrega el *TPR*, la nota que obtenga en el mismo *reemplazará* a la nota (o al ausente) que originalmente tuviese en ***UNO (Y SÓLO UNO)*** de los cuatro *Trabajos Prácticos* originales. Por esta razón, el *TPR* sirve no solamente para cubrir un aplazo o un ausente en ***UN*** *Trabajo Práctico*, sino también para levantar ***UNA*** nota de *Trabajo Práctico* si el estudiante necesita hacerlo para aspirar a la condición de Aprobación Directa.

- ✓ Un conjunto de veintinueve *Cuestionarios Teóricos* tipo preguntas de opciones múltiples, a desarrollar en el aula virtual (a modo de refuerzo de lectura y estudio organizado y continuo de materiales). A partir del ciclo 2019, los estudiantes tendrán ***DOS*** oportunidades de hacer y completar cada *Cuestionario* (en lugar de sólo una, como era el caso hasta el ciclo 2018). El segundo intento cuenta como una instancia de recuperación de cada *Cuestionario*. La nota final que quedará registrada para cada *Cuestionario*, será el *promedio* entre los dos intentos que cada estudiante haya hecho (o bien la nota del único intento que haya desarrollado, si hizo uno sólo).
- ✓ Un conjunto de cuatro *Desafíos de Programación* consistentes en resolver algún problema y sólo informar los resultados. Como sólo se informan los resultados obtenidos, y estos deben coincidir con los resultados previstos por la Cátedra, cada estudiante tiene ***CINCO*** oportunidades de informar sus resultados hasta llegar eventualmente a la respuesta correcta (y cada uno de los intentos extra cuenta como instancia de recuperación). La nota final que quedará registrada para un estudiante en un *Desafío*, será la más alta que haya logrado entre los intentos que haya usado.
- ✓ Los cuatro *Trabajos Prácticos* (con el eventual *Trabajo Práctico Recuperatorio* ya aplicado para reemplazar alguna nota original) se promediarán con las notas finales de los veintinueve *Cuestionarios* y con las notas finales de los cuatro *Desafíos*, para formar una única *Nota Final de Trabajos Prácticos* (abreviado como *NFTP*) que irá a la libreta junto con las notas de los cuatro *Parciales*.
- ✓ El promedio *NFTP* se calculará en *forma ponderada*: las notas que se obtengan en cada actividad tendrán pesos diferentes en el cálculo final. Cada uno de los cuatro *Trabajos Prácticos* tendrá un peso del *17.5%* del total (con lo cual, entre los cuatro suman el *70%* del total en el cálculo de la *NFTP*). El promedio entre los veintinueve *Cuestionarios* tendrá un peso del *10%* sobre el total de la *NFTP*. Y el promedio entre los cuatro *Desafíos* tendrá un peso del *20%* sobre el total.
- ✓ Con lo anterior, y designando al promedio de todos los *Cuestionarios* como *PC* y al promedio de todos los *Desafíos* como *PD*, entonces la fórmula de cálculo de la *NFTP* es la siguiente:

NFTP = 0.175*TP1 + 0.175*TP2 + 0.175*TP3 + 0.175*TP4 + 0.1*PC + 0.2*PD

- ✓ El valor final así obtenido, será redondeado al entero más próximo, y esa finalmente será la nota *NFTP*.

Criterios de evaluación (los cuales serán tenidos en cuenta en las correcciones)	• Los *trabajos prácticos* se entregan a través del aula virtual, y son revisados y calificados por los docentes de la Cátedra. • Los *cuestionarios* y *desafíos* son de corrección automática. • En casos de *parciales* y *trabajos prácticos*, los alumnos deben presentar a través del aula virtual proyectos desarrollados en Python, que incluyan los archivos fuente implementados. • Cuando se trate de *cuestionarios* y *desafíos de programación*, los alumnos simplemente deben subir sus respuestas mediante los recursos disponibles en el aula virtual. • El enunciado de cada *trabajo práctico* incluye consignas a cumplir y recomendaciones adicionales, además de criterios de evaluación tales como: compilación, completitud de consignas cubiertas, diseño de clases, lógica aplicada en la solución de cada ítem, diseño de interfaz de usuario y elementos de control de carga e integridad. Cada criterio se evalúa y aporta un peso a la calificación final. De acuerdo a la suma de esos pesos, se aplica la escala de notas que ya fue indicada en secciones anteriores de este mismo documento (y que forma parte de la consigna mostrada a los estudiantes). • Todo lo expresado en el ítem anterior para los trabajos prácticos, vale también para los parciales. • Esencialmente, se aplica un esquema de evaluación continua, con reglas de trabajo firmes pero claras. • En todo momento los estudiantes saben cuál será el esquema de evaluaciones, cómo serán evaluados, y cuándo les serán entregadas sus devoluciones y calificaciones. • El soporte del aula virtual única y centralizada para todos los cursos, con exactamente el mismo y único modelo de intervención para todos los cursos, facilita el proceso de avance y de control.
Regularidad: condiciones (Describir las condiciones necesarias para regularizar. Se sugiere incluir la aclaración que el estudiante en condición de regular puede rendir en el plazo de un ciclo lectivo sin control de correlativas aprobadas)	• Para ***regularizar*** la materia, el alumno tendrá cinco notas al final del año, que irán a la libreta: los parciales P1, P2, P3 y P4 más la NFTP. Para obtener la condición de ***Regular*** el alumno deberá: **a.** *Aprobar con nota mayor o igual a 4(cuatro) al menos tres de los cuatro parciales (combinados en la forma que sea) O BIEN los dos del segundo semestre (parciales P3 y P4). Como queda dicho, se puede hacer el Parcial Recuperatorio (PR) para <u>UN y SÓLO UN</u> Parcial para cumplir con esta regla, y la nota del PR reemplaza a la del Parcial original. Quede claro: aprobando los dos parciales del segundo semestre (o uno de ellos y el PR), el alumno podrá regularizar incluso si tiene desaprobados o no realizados los dos del primero. Y de cualquier forma, aprobando tres parciales cualesquiera (o dos cualesquiera y el PR) también se podrá lograr la regularidad.* **b.** *Obtener un promedio final de 4(cuatro) o más en la NFTP (<u>no es obligatorio</u> que los cuatro trabajos prácticos se entreguen ni que se*

hagan todos los cuestionarios y desafíos, siempre que al final del cursado el promedio ponderado que conforma la NFTP sea mayor o igual a 4). Recordemos que en el cálculo de la NFTP ya están aplicados los recuperatorios de Trabajos Prácticos, Cuestionarios y Desafíos.

- Por reglamento aprobado desde 2017 en la UTN, los alumnos que alcancen la condición de regular podrán rendir cualquier asignatura dentro del mismo ciclo lectivo sin que se controlen las correlativas previas (en caso de haberlas). A partir del siguiente ciclo lectivo, sólo se podrá rendir una asignatura si previamente se rindieron y aprobaron todas las previas. La materia AED no tiene materias previas, pero la norma se cita aquí en carácter informativo (para ser tenida en cuenta por los estudiantes en asignaturas posteriores).
- Escala de notas para el proceso de regularidad(*)

NOTAS	PORCENTAJE	CALIFICACIÓN
1		No Aprobado
2		No Aprobado
3		No Aprobado
4	55% a 57%	Aprobado
5	58% a 59%	Aprobado
6	60% a 68%	Aprobado
7	69% a 77%	Aprobado
8	78% a 86%	Aprobado
9	87% a 95%	Aprobado
10	96% a 100%	Aprobado

(*) Escala acordada en reunión de Docentes Coordinadores

- Todo alumno que no llegue a la condición mínima de regular quedará en condición de ***Libre*** y deberá recursar la materia. Como ya se indicó más arriba, y sólo a los efectos estadísticos, estos alumnos serán clasificados como alumnos ***libres*** o como alumnos que ***abandonaron*** el cursado, en base a los siguientes criterios:

  - ✓ *Se considerará que un alumno **Abandonó** el cursado de la materia (y será registrado como tal) si quedó libre y no llegó a hacer todas las evaluaciones principales previstas para el proceso de regularidad. Aquí consideramos como evaluaciones principales a los cuatro parciales y los cuatro trabajos prácticos (o sea, ocho instancias en total).*
  - ✓ *Se considerará que un alumno quedó en estado de **Libre** (y será registrado como tal) si desarrolló todas las ocho instancias de evaluación principales, pero no alcanzó las calificaciones y*

	promedios requeridos para regularizar.
Aprobación Directa: condiciones. (la calificación será la nota registrada como Nota Final en Autogestión) (Se sugiere incluir la aclaración que el estudiante, en esta condición, puede registrar su nota en examen en el plazo de un ciclo lectivo, sin control de correlativas aprobadas, y después de ello se le exigirán correlativas aprobadas)	• A partir del ciclo 2017 y siguientes se implementa la situación final de *Aprobación Directa*, la cual implica que todo alumno que la logre quedará eximido por completo del examen final y sólo deberá inscribirse en cualquier turno de exámenes y presentarse para registrar su nota final en actas y en la libreta. Las condiciones para obtener la *Aprobación Directa* son las siguientes: a. *Rendir los cuatro parciales, aprobarlos a todos con notas no menores a 7(siete) y promedio entre los cuatro mayor o igual a 8(ocho). UNO Y SÓLO UNO de estos parciales puede ser recuperado (para levantar un aplazo, cubrir un ausente o simplemente levantar una nota) y el recuperatorio reemplaza a la nota original a los efectos de la aplicación de esta regla.* b. *Obtener un promedio en la NFTP mayor o igual a 8(ocho) pero con todos los Trabajos Prácticos entregados y aprobados con nota de 7 o más. Recordemos que en el cálculo de la NFTP ya están aplicados los recuperatorios de Trabajos Prácticos, Cuestionarios y Desafíos.* c. *Rendir y aprobar con nota de 6 o más el Coloquio Final Teórico (que hemos designado como Parcial 5 o P5).* • La condición de ***Aprobación Directa*** no se pierde con el transcurso del tiempo: los estudiantes en esta condición pueden inscribirse en cualquier turno de examen posterior al cursado. Además, estos estudiantes podrán inscribirse a exámenes dentro del plazo de un ciclo lectivo completo sin que se controlen las correlatividades previas. Pero luego de ese ciclo completo, sólo podrán inscribirse a examen para firmar su aprobación directa si antes aprobaron las materias correlativas previas. • Para los alumnos que alcancen la condición de ***Aprobación Directa***, la nota final de la materia, que será registrada en el acta de examen y en la libreta, *será el promedio entre los cinco parciales (los cuatro normales de regularidad y el quinto parcial) y la nota final del práctico (con el esquema de recuperatorios ya incluido).* Esta nota será también registrada en la planilla final de regularidad en el Sistema de Autogestión.
Promoción: condiciones (Aclarar si hubiera promoción de alguna parte de la asignatura, las condiciones y si tiene duración, con el mayor detalle posible)	• Un estudiante que cumpla con todos los requisitos para llegar a la situación de *Aprobación Directa*, pero al rendir el *P5* obtenga nota menor a 6, quedará inmediatamente en condición de ***Promocionado*** (y es esta la única forma en que un estudiante puede quedar en esa condición). Un estudiante en condición de ***Promocionado***, deberá rendir el examen final, *pero sólo la parte teórica*. • La condición de ***Promocionado*** no se pierde con el transcurso del tiempo, pero sí se perderá si el alumno resultare aplazado al

<table>
<tr><td></td><td>presentarse a rendir el examen final. Vale decir: después del aplazo en un final, la promoción ya no valdrá para los turnos siguientes y el alumno deberá rendir como regular.</td></tr>
<tr><td>Modalidad de examen final (Describir las características metodológicas del examen final para los distintos estados del estudiante)</td><td>

Alumnos regulares: desarrollo de un programa en el laboratorio sobre todos los temas de la asignatura. Si aprueba esta instancia, un coloquio breve con un profesor de la cátedra, que incluye también los temas de las unidades números 10 y 11.
Alumnos promocionados: un coloquio sobre todos los temas de la asignatura, que incluye los temas de las unidades 10 y 11.
Alumnos con aprobacion directa: sólo deben inscribirse a rendir y pasar al aula del examen el día en que sustancie, para registrar su nota final en el acta del examen y en su libreta.
Escala de Notas para Examen Final (*):

<table>
<tr><th>NOTA</th><th>PORCENTAJE</th><th>CALIFICACIÓN</th></tr>
<tr><td>1</td><td></td><td>Insuficiente</td></tr>
<tr><td>2</td><td></td><td>Insuficiente</td></tr>
<tr><td>3</td><td></td><td>Insuficiente</td></tr>
<tr><td>4</td><td></td><td>Insuficiente</td></tr>
<tr><td>5</td><td></td><td>Insuficiente</td></tr>
<tr><td>6</td><td>60% a 68%</td><td>Aprobado</td></tr>
<tr><td>7</td><td>69% a 77%</td><td>Bueno</td></tr>
<tr><td>8</td><td>78% a 86%</td><td>Muy Bueno</td></tr>
<tr><td>9</td><td>87% a 95%</td><td>Distinguido</td></tr>
<tr><td>10</td><td>96% a 100%</td><td>Sobresaliente</td></tr>
</table>
(*) Escala acordada en reunión de Docentes Coordinadores
</td></tr>
<tr><td>Actividades en laboratorio</td><td>

✓ En los primeros dos meses las clases teóricas se llevarán a cabo en las aulas comunes, y las prácticas en los laboratorios.
✓ Luego de esos dos primeros meses, todas las clases se desarrollarán en los laboratorios de informática.
✓ Cada alumno debe instalar en su computadora personal las herramientas de software requeridas por la materia, y realizar en forma personal prácticas y pruebas sobre ellas para lograr dominio pleno de su uso.
✓ Es bienvenida la utlización por parte de los alumnos de sus propias notebooks, en el trancurso de cada clase.
✓ En varias clases se prevé la realización de trabajos y tareas pautadas para ser teminadas (en la medida de lo posible) en el transcurso de la misma clase. Esos trabajos se suben al Aula Virtual para su revisión por parte de los jefes de trabajos prácticos y auxiliares del curso, o son evaluados en forma automática mediante mecanismos del Aula Virtual.

</td></tr>
<tr><td>Cantidad de horas prácticas totales (en el aula)</td><td>160 horas (32 semanas de clase, 5 horas cátedra por semana).</td></tr>
<tr><td>Cantidad de horas teóricas totales (en el aula)</td><td>80 horas.</td></tr>
<tr><td>Cantidad de horas estimadas totales de trabajo</td><td>80 horas.</td></tr>
</table>

(extra áulicas).	
Horas/año totales de la asignatura (en el aula).	☐ Formación experimental ■ Resolución de problemas de ingeniería ☐ Actividades de proyecto y diseño ☐ Prácticas supervisadas en los sectores productivos y /o de servicios
Tipo de formación práctica (sólo si es asignatura curricular -no electiva-)	34 horas.
Cantidad de horas cátedras afectadas a la formación práctica indicada en el punto anterior (sólo si es asignatura curricular -no electiva-)	160 horas (32 semanas de clase, 5 horas cátedra por semana).
Descripción de los prácticos	• *Cuestionarios Teóricos*: Cuestionarios de opciones múltiples, uno por semana a modo de control de lectura y refuerzo de temas. • *Desafíos de Programación*: Enunciados sobre temas diversos, que deben ser resueltos por los alumnos informando los resultados obtenidos (sin subir los propios programas). Se complementan con preguntas teóricas. • *Trabajos Prácticos*: Enunciados situacionales generales. Los alumnos resuelven en grupos y entregan el programa completo y funcionando.
Cronograma de actividades de la asignatura (contemplando las fechas del calendario 2019 y para cada unidad)	Ver achivo *Crono AED [2019].xls* anexo a esta presentación.
Propuesta para la atención de consultas y mail de contacto.	Todos los profesores, jefes de trabajos prácticos y auxiliares de la Cátedra están registrados y disponibles en el Aula Virtual de la asignatura, en la que todos los alumnos deben a su vez registrarse. Mediante el servicio de mensajería y los foros del Aula Vritual, todos los alumnos pueden realizar todas las consultas que deseen a cualquiera de los docentes. Además, la Cátedra (en conjunto con el Departamento de Ingeniería en Sistemas de Información) tiene implementado un esquema de *tutorías* para atención de alumnos, fuera de los horarios de clase. El esquema de *horarios de tutorías* es gestionado por Departamento de Sistemas y es atendido por docentes de la Cátedra.
Plan de integración con otras asignaturas	Esta asignatura es la correlativa previa natural de Paradigmas de Programación, y aporta elementos fundamentales para todas las asignaturas de programación de la carrera y para las asignaturas que directa o indirectamente apliquen sobre el diseño y desarrollo de software. Por lo tanto, la Cátedra en su conjunto ha realizado un gran esfuerzo por actualizar sus contenidos y prácticas, de forma de abarcar tan profundamente como sea posible los contenidos mínimos exigidos por la Ordenanza 1150 de Rectorado que regula la creación y funcionamiento de la Carrera. Dichos contenidos son requisitos previos indispensables en numerosas asignaturas posteriores, y de esta forma se ha tratado de garantizar que los alumnos que regularizan AED tengan la formación básica esperable.
Bibliografía Obligatoria	[1] Frittelli, V., et al. (2018). Publicaciones Cátedra de AED. *Fichas de Estudio para AED (disponible en Aula Virtual AED)*. Cordoba: UTN – Facultad Regional Córdoba. [2] Python Software Foundation. (2017). *Python Documentation*. Retrieved

	February 24, 2016, from https://docs.python.org/3/.
Bibliografía Complementaria	[1] Pilgrim, M. (2004). *Dive Into Python - Python from novice to pro*. Retrieved March 6, 2016, from http://www.diveintopython.net/toc/index.html.
Distribución de docentes	Ver achivo *AED Planta Docente [2019].pdf* anexo a esta presentación.

Firma: …….

Aclaración: …….....