INSTITUTO/S: Instituto de Tecnología e Ingeniería

CARRERA/S: Tecnicatura Universitaria en Informática

MATERIA: Estructuras de datos

NOMBRE DEL RESPONSABLE DE LA ASIGNATURA: Sergio Gonzalez

EQUIPO DOCENTE: Sergio Gonzalez

Fernando Puricelli

CUATRIMESTRE: Primero - Segundo

AÑO: 2020

PROGRAMA N°: Aprob. Por Cons.Directivo 30/10/2018

Instituto/s: Instituto de Tecnología e Ingeniería
Carrera/s: Tecnicatura Universitaria en Informática
Nombre de la materia: Estructuras de datos
Responsable de la asignatura y equipo docente: Gonzalez Sergio
Cuatrimestre y año: Primer y segundo cuatrimestre de 2020
Carga horaria semanal: 8 hs
Programa N°: Aprobado por consejo directivo el 30/10/2018
Código de la materia en SIU: 752

Estructuras de datos

1. Fundamentación

El perfil del egresado de la carrera, se apunta a un profesional de la construcción de software con parámetros de calidad, tanto en etapas de diseño como de programación, con una base conceptual sólida que permita adaptarse a proyectos de distinta índole y favorecer el trabajo en equipo y colaborativo. En este contexto, la materia toma un importante rol, ya que, en esta se adquieren conceptos centrales que luego se utilizarán a lo largo de la carrera y en la práctica profesional.

2. Propósitos y/u objetivos

Objetivos

Se desea que el estudiante:

- Comprenda los conceptos de programación estructurada y modular.
- Comprenda del concepto de dato, tipos de datos primitivos y estructuras de datos, y su importancia e interrelación con la resolución algorítmica de un problema.
- Conozca de la interface de una estructura de datos y su utilización para la resolución de problemas.
- Se familiarice con los conceptos de ámbito y de pasaje de parámetros por valor o por referencia.
- Pueda resolver problemas mediante programas recursivos, y entienda la diferencia entre una solución resursiva y una iterativa

- Comprenda el concepto de asignación dinámica de memoria y pueda escribir programas que realicen un manejo dinámico explicito de memoria en forma adecuada.
- Conozca las estructuras de datos básicas y su interfaz: Vectores, matrices, pilas, colas, listas, árboles, diccionarios, hashing, grafos, etc.
- Comprenda el funcionamiento interno de estructuras de datos básicas y maneje principios básicos del diseño de la interfaz.
- Entienda la noción de implementación de una estructura de datos, y de su eficiencia, y sea capaz de implementar las interfaces vistas anteriormente con alternativas variadas en eficiencia.
- Se familiarice con las tareas de compilar y vincular programas para lograr un ejecutable.
- Incorpore la elección de la estructura de datos como una parte de la solución de un problema, en relación con la resolución algorítmica y la programación en computadora.
- Utilize e internalice buenas prácticas de programación, en relación a la práctica profesional, trabajo en equipo y generación de código legible y portable.

3. Programa sintético:

Programación estructurada y modular. Programas recursivos. Representación de datos en memoria. Paso de variables por valor y por referencia. Tipos abstractos de datos. Estructuras de datos. Estructuras contenedoras: Vectores, matrices, pilas colas, lista, diccionarios, árboles y grafos. Implementación de estructuras de datos estáticas. Uso dinámico de memoria. Listas y árboles implementados con punteros. Análisis, eficiencia e implementación de estructuras de datos. Algoritmos de recorrido, búsqueda y ordenamiento. Nociones básicas de algoritmos sobre grafos.

4. Programa analítico

Unidad 1: Introducción

- Incorporar conceptos básicos necesarios para escribir, compilar y ejecutar un programa.

- Utilizar reglas sintácticas. Tipos de datos primitivos y referenciales.
- Definición de funciones y variables. Operadores. Estructuras de control.
- Proporcionar ejemplos y realizar ejercicios.

Unidad 2: Tipos de datos abstractos

- Incorporar el concepto de abstracción de datos y encapsulamiento de datos.
- Incorporar el concepto de tipo abstracto de dato (TDA).
- Diferenciar entre tipo de dato y tipo abstracto de dato.
- Realizar requerimientos y diseño de un TDA.
- Trabajar con TDAs

Unidad 3: Funciones recursivas, arreglos unidimensionales y bidimensionales

- Incorporar principio de recursión. Funcionamiento de recursión. Trabajar problemas y ejercicios con soluciones resursivas.
- Trabajar con arreglos unidimensionales y bidimensionales, representación en memoria, creación, recorrido, inserción y eliminación de elementos.

Unidad 4: Pilas estáticas, definición y terminología

- Incorporar definición y terminología de pilas.
- Trabajar con el TDA pila. Operaciones básicas.
- Realizar implementación estática mediante arreglos.

Unidad 5: Colas estáticas, definición y terminología

- Incorporar definición y terminología de colas.
- Trabajar con el TDA cola. Operaciones básicas.
- Realizar implementación estática mediante arreglos.
- Colas simples y circulares.

Unidad 6: Listas

- Incorporar definición y terminología de listas.
- Trabajar con TDA lista. Operaciones básicas.
- Implementación dinámica utilizando punteros.
- Listas simples enlazadas, dobles enlazadas y circulares.

Unidad 7: Árboles

- Incorporar definición y terminología de árbol.
- Árboles binarios. Árboles binarios de búsqueda. Realizar implementación utilizando punteros. Recorridos e implementaciones recursivas. Inserción, eliminación, modificación y búsqueda de elementos.
- Árboles balanceados. Árboles AVL. Árboles de M-vias.
- Trabajar con árboles y sus aplicaciones.

Unidad 8: Grafos

- Incorporar definición y terminología de grafos.
- Realizar recorridos básicos, aplicaciones y ejemplos.

4.1 Bibliografía y recursos obligatorios:

Alfred Aho & Jeffrey Ullman. (1983) “Data Structures and Algorithms”, Addison – Wesley.

Chris Okasaki. (1998) “Purely Functional Data Structures”, Cambridge University Press.

Mark Allen Weiss. (1997) “Data Structures and Algorithms in C”. Addison – Wesley (2da Edición).

5. Metodología de enseñanza:

Las clases constarán de clases teóricas cortas, en las cuales se verterán los conceptos básicos de cada uno de los temas en cuestión, acompañados con ejemplos prácticos en el pizarrón y favoreciendo la participacion de los/las estudiantes. La mayor cantidad de tiempo estará dedicado a la parte práctica, con la resolución de problemas en el pizarrón de forma constructiva con participación de los/las estudiantes y principalmente con la realización de ejercicios por parte de los/las estudiantes en el laboratorio de computación. Cada unidad perteneciente a los contenidos mínimos tendrá una guía de ejercicios y será de caracter obligatorio la entrega de ejercicios resueltos de cada una de las guías a elección del docente en grupos de hasta dos personas.

6. Plan de trabajo en el campus:

En el aula virtual se propondrá material educativo, apuntes de clase y bibliografía; así como también el programa y el cronograma de la asignatura y las guías de ejercicios y trabajos prácticos. Las entregas programadas de ejercicios obligatorios de los trabajos prácticos, también se realizarán a través del campus.

7. Evaluación y régimen de aprobación

7.1 Aprobación de la cursada

Para aprobar la cursada y obtener la condición de regular, el régimen académico establece que debe obtenerse una nota no inferior a cuatro (4) puntos. Todas las instancias evaluativas deberán tener una instancia de recuperatorio. Podrán acceder a la administración de esta modalidad solo aquellos y aquellas estudiantes que hayan obtenido una nota inferior o igual a 6 (seis) puntos en el examen parcial.

Siempre que se realice una evaluación de carácter recuperatorio, la calificación que los/as estudiantes obtengan reemplazará la calificación obtenida en el examen que se ha recuperado y será la considerada definitiva a los efectos de la aprobación.

El alumno deberá poseer una asistencia no inferior al 75% en las clases presenciales.

7.2 Aprobación de la materia

La materia puede aprobarse por promoción, evaluación integradora, examen final o libre.

Promoción directa: tal como lo establece el art°17 del Régimen Académico, para acceder a esta modalidad, el/la estudiante deberá aprobar la cursada de la materia con una nota no

inferior a siete (7) puntos, no obteniendo en ninguna de las instancias de evaluación parcial menos de seis (6) puntos, sean evaluaciones parciales o recuperatorios. El promedio estricto resultante deberá ser una nota igual o superior a siete(7) sin mediar ningún redondeo.

Evaluación integradora: tal como lo establece el art°18 del Régimen Académico, podrán acceder a esta evaluación aquellos estudiantes que hayan aprobado la cursado con una nota de entre cuatro (4) y seis (6) puntos.
La evaluación integradora tendrá lugar por única vez en el primer llamado a exámenes finales posterior al término de la cursada. Deberá tener lugar en el mismo día y horario de la cursada y será administrado, preferentemente, por el/la docente a cargo de la comisión. Se aprobará tal instancia con una nota igual o superior a cuatro (4) puntos, significando la aprobación de la materia.
La nota obtenida se promediará con la nota de la cursada.

Examen final: Instancia destinada a quienes opten por no rendir la evaluación integradora o hayan regularizado la materia en cuatrimestres anteriores. Se evalúa la totalidad de los contenidos del programa de la materia y se aprueba con una calificación igual o superior a cuatro (4) puntos. Esta nota no se promedia con la cursada.

7.3 Criterios de calificación

La calificación de las/los estudiantes se compone de las notas de los parciales (con sus respectivos recuperatorios), haciendo foco en las resoluciones conceptuales y otorgando menor peso a errores de codificación, intentando no evaluar los conocimientos de un lenguaje de programación en particular. Se tomará en cuenta el trabajo en clase y las entregas obligatorias de ejercicios de las guías de trabajos prácticos.

8. Cronograma

Clase	Temario
1	Unidad 1: Presentación de los contenidos y temas de la materia y estructura de las clases. Estructura básica de computadoras modernas. Lenguajes de programación. Escritura, compilación y ejecución. Tipos de datos primitivos. Primeros ejemplos prácticos y ejercicios en PC.
2	Unidad 1: Lenguajes de programación. Reglas sintácticas. Lenguaje de programación Python. Operadores básicos. Estructuras de control. Comunicación con usuario. Ejemplos prácticos y ejercicios en PC.
3	Unidad 1: Lenguaje de programación Python. Definición de funciones y paso de parámetros. Ejemplos prácticos y ejercicios en PC.

4	Unidad 1: Lenguaje de programación Python. Ejemplos prácticos y ejercicios en PC.
5	Unidad 2: Concepto de tipo de dato abstracto (TDA). Requerimientos y diseños de TDA. Diferentes tipos de TDA. Ejemplos prácticos y ejercicios en PC.
6	Unidad 2: Concepto de tipo de dato abstracto (TDA). Requerimientos y diseños de TDA. Diferentes tipos de TDA. Ejemplos prácticos y ejercicios en PC.
7	Unidad 3: Programación modular, Programación estructurada. Resoluciones iterativas. Introducción a Recursividad. Ejemplos prácticos y ejercicios en PC.
8	Unidad 3: Implementaciones recursivas. Cadenas. Arreglos unidimensionales. Arreglos multidimensionales. Operaciones básicas. Asignación de memoria. Ejemplos prácticos y ejercicios en PC.
9	Unidad 3: Ejemplo de algoritmos recursivos: Algoritmos de ordenamiento y búsqueda.
10	Unidad 3: Algoritmos de ordenamiento y búsqueda. Implementación.
11	Unidad 4: Pilas estáticas. Definición y operaciones básicas. Implementación con arreglos.
12	Unidad 4: Pilas estáticas. Ejercicios practicos y ejemplos en PC
13	Unidad 5: Colas estáticas. Definición y operaciones básicas. Implementación con arreglos.
14	Unidad 5: Colas estáticas. Ejercicios practicos y ejemplos en PC. Repaso. Clase de consulta
15	**Primer Parcial**
16	Unidad 6: Estructuras dinámicas. Listas. Definición y terminología. Ejemplos prácticos y ejercicios en PC.
17	Unidad 6: Listas. Ejemplos prácticos y ejercicios en PC.
18	Unidad 6: Listas. Implementacion de pilas y colas dinámicas.
19	Unidad 6: Listas. Implementacion recursiva.

20	Unidad 7: Árboles. Definición y terminología. Ejemplos de uso de arboles en resolución de problemas. Ejemplos prácticos y ejercicios en PC.
21	Unidad 7: Árboles binarios. Árboles binarios de búsqueda. Definiciones de peso, altura, etc. Inserción y borrado en árboles binarios. Ejemplos prácticos y ejercicios en PC.
22	Unidad 7: Árboles binarios. Árboles binarios de búsqueda. Ejemplos prácticos y ejercicios en PC.
23	Unidad 7: Árboles balanceados. Árboles AVL. Árboles de M vias. Implementación de balanceo en árboles binarios.
24	Unidad 7: Árboles balanceados. Árboles AVL. Árboles de M vias. Implementación de balanceo en árboles binarios.
25	Unidad 8: Grafos. Definicion y terminología. Ejemplos de usos de grafos en casos reales.
26	Unidad 8: Grafos simples y multiple. Grafos dirigidos y no dirigidos. Grafos pesados. Grafos bipartitos.
27	Unidad 8: Grafos. Caminos y ciclos. Grafos ciclicos y aciclicos. Árboles como casos particulares de grafos. Recorridos.
28	Unidad 8: Grafos. Conectividad. Grafos conexos. Componentes gigantes. Clase de consultal
29	**Segundo Parcial**
30	Repaso: Clase de Consulta
31	Repaso: Clase de Consulta
32	**Recuperatorios**