Ministerio de Educación y Deportes
Universidad Tecnológica Nacional
Facultad Regional San Francisco

"2017 - Año de las Energías Renovables"

San Francisco, 21 de diciembre de 2017

VISTO Lo dispuesto por la Ordenanza 1383/12, y

CONSIDERANDO:

Que por medio de esta normativa y mediante el dictado de asignaturas electivas es posible incorporar perfiles propios de la región a efectos de adaptar los diseños curriculares a las necesidades de la misma.-

Que en tal sentido y en cumplimiento de las reglamentaciones vigentes, y a propuesta de los Departamentos respectivos los Consejos Directivos de las Facultades Regionales definirán cuales serán las materias electivas, área del conocimiento, objetivos generales y específicos que justifiquen la inclusión, carga horaria, sus contenidos analíticos, bibliografía, modalidad de dictado, propuesta pedagógica, y sus correspondientes correlatividades debidamente justificadas.-

Que el Consejo Departamental de Ing. En Sistemas de Información elevó al Consejo Directivo de esta Facultad Regional San Francisco la propuesta de implementación de materias electivas.-

Por ello,

EL CONSEJO DIRECTIVO DE LA FACULTAD REGIONAL SAN FRANCISCO

RESUELVE:

ARTICULO 1º.-Aprobar la continuidad del dictado de **Construcción de Software** como materia electiva parte de la currícula de la Carrera Ingeniería en Sistemas de Información del área Programación a dictarse en el tercer nivel, con modalidad cuatrimestral (segundo cuatrimestre), con una carga horaria de 8 horas semanales.

ARTICULO 2º.- Aprobar en **Anexo I**, Objetivo General y objetivos específicos que justifican la inclusión de dicha materia, las correlatividades debidamente justificadas, el programa analítico, la bibliografía y la propuesta pedagógica.

ARTICULO 3º.- Regístrese. Comuníquese. Cumplido, archívese.-

RESOLUCION C.D. Nº : 711 /2017

Ing. ALBERTO R. TOLOZA
Decano

Ing. JUAN CARLOS CALLONI
Secretaria Académica

Anexo Nº I
Construcción de Software

1. Objetivos generales y específicos que justifican la inclusión de la Materia

Objetivo General:
Los objetivos de la asignatura se fundamentan en otorgar a los alumnos las mejores técnicas y métodos para la escritura de código de programación utilizando distintas herramientas de escritura aplicables al desarrollo de software.

Objetivos específicos:

1. Escribir código con buenas costumbres de programación.
2. Saber distinguir entre código de calidad y código amateur.
3. Adquirir y evaluar conceptos tales como la programación defensiva y ocultamiento de información.
4. Asumir la importancia de programar pensando en la calidad del código fuente.
5. Comprender la necesidad de ser disciplinado en la programación de software y en el uso de sus herramientas.

2. Correlatividades debidamente justificadas

Para Cursar

Regularizadas

a. **Gestión de Datos**: Esta materia es necesario tenerla cursada y regular ya que el alumno necesita tener hábitos en la descripción de algoritmos para el tratamiento de la información almacenada, conocer la utilización de tipos de accesos diferentes, expresiones e instrucciones, conocer el diseño de algoritmos, poniendo en práctica las distintas operaciones para consultar, actualizar y mantener la información guardada en los archivos.
b. **Análisis de Sistema**: Esta materia es necesario tenerla cursada y regular ya que el alumno necesita saber modelar las características intrínsecas de los sistemas de información, conocer y aplicar las metodologías, modelos, técnicas y lenguajes de la etapa de análisis, seleccionar adecuadamente los modelos que mejor se adapten para dar soluciones a los problemas de información.

Aprobadas

a. **Algoritmo y estructura de datos**: Esta materia es necesario tenerla aprobada ya que el alumno necesita tener afianzado los contenidos para poder resolver situaciones problemáticas bajo el paradigma imperativo (representación gráfica, lenguaje nemotécnico), organizar los PROGRAMAS: paradigmas, lenguajes, desarrollos y estilos, tener los conocimientos para almacenar datos en las distintas estructuras de datos como una manera conceptual de organizar los datos (pilas, colas, tablas, listas, árboles y grafos).

Para Rendir

Regularizadas

a. **Diseño de Sistemas:** Esta materia es necesario tenerla cursada y regular ya que el alumno necesita tener herramientas necesarias para que el alumno pueda escoger las herramientas más adecuadas para diseñar un sistema de información y construirlo exitosamente, conocer las metodologías, modelos, técnicas y lenguajes del proceso de diseño, diseñar y construir productos de software asociado a los sistemas de información aplicando herramientas de soporte de diseño.

Aprobadas

a. **Gestión de Datos:** Esta materia es necesario tenerla aprobada para rendir la cátedra en cuestión ya que el alumno necesita tener afianzado los contenidos para conocer la utilización de tipos de accesos diferentes, expresiones e instrucciones, conocer el diseño de algoritmos, poniendo en práctica las distintas operaciones para consultar, actualizar y mantener la información guardada en los archivos, saber sobre el diseño de bases de datos DER y Formas Normales .
b. **Análisis de Sistema:** Esta materia es necesario tenerla aprobada para rendir la cátedra en cuestión ya que el alumno necesita tener afianzado los contenidos para las características intrínsecas de los sistemas de información, conocer y aplicar las metodologías, modelos, técnicas y lenguajes de la etapa de análisis, seleccionar adecuadamente los modelos que mejor se adapten para dar soluciones a los problemas de información.

3. Programa analítico

Eje Temático Nº 1: Prerrequisitos de la construcción

Unidad Nº 1:

Introducción. Prerrequisito de la definición del problema. Prerrequisito de los requerimientos. Prerrequisito de la arquitectura. La elección del lenguaje de programación. Convenciones de programación.

Eje Temático Nº 2: Diseño

Unidad Nº 2: Rutinas

Pasos en la construcción de una rutina. Proceso de programación con PseudoCódigo. Diseño de la rutina. Codificación de la rutina. Características de rutinas de alta calidad. Razones para crear una rutina. Buenos nombres de rutinas. Cohesión y acoplamiento de rutinas. Longitud de una rutina. Programación defensiva. Utilización de parámetros.

Unidad Nº 3: Módulos - Clases

Cohesión y acoplamiento. Ocultamiento de información. Razones para crear módulos y Clases.

Eje Temático Nº 3: Datos

Unidad Nº 4:

Creación de tipos de datos. Tipos de datos enumerados. Guías para la inicialización de datos. Consideraciones en la elección de nombres de variables. Temas generales sobre el uso de variables.

Eje Temático Nº 4: Control

Unidad Nº 5:

Organización de código lineal. Uso de condicionales. Sentencias If. Sentencias Switch. Selección del tipo de bucle. Control del bucle. Estructuras de control inusuales. Expresiones booleanas. Bloques de sentencias. Domesticar los anidamientos profundos. Estructuras de control y complejidad.

Eje Temático Nº 5: Consideraciones constantes

Unidad Nº 6: Documentación del código

Código autodocumentado. El estilo de programación como documentación. Claves para comentarios efectivos. Técnicas para comentarios.

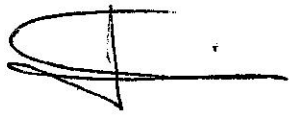
Unidad Nº 7: Tamaño del programa

Como el tamaño del programa afecta a la construcción. Efecto del tamaño del proyecto sobre las actividades de desarrollo. Efecto del tamaño del proyecto sobre los errores.

Eje Temático Nº 6: Mejorar la calidad

Unidad Nº 8:

Características de la calidad del software. Técnicas para mejorar la calidad del software. Efectividad relativa de las técnicas. Revisiones. El papel de las revisiones en la calidad del software. Inspecciones. Otros tipos de revisiones.



4. Bibliografia

Steve McConnell
Code Complete 2
Microsoft Press

Steve McConnell
Code Complete
Microsoft Press

Estándares
Obtenidos de la documentación referida de distíntos lenguajes de programación actuales

5. Propuesta pedagógica

Teoría
Duración: 4 horas semanales a cargo del profesor.

Tendrán el carácter de teórico-prácticas.
En las mismas se impartirán los conocimientos claves del contenido de la materia, presentando y evaluando buenas costumbres de programación destinadas a lograr código de buena calidad.

Se ejemplificaran los conceptos en diferentes lenguajes de programación. La teoría será reafirmada mediante la realización de cuestionarios que los alumnos deberán responder en grupo en clase, como así también la revisión de código propio y ajeno, utilizando los conocimientos recién adquiridos y su propia experiencia.

Práctica y laboratorio
Duración: 4 horas semanales a cargo del profesor.

En las mismas se impartirán guías de ejercicios grupales. Se codificaran rutinas y programas en el lenguaje de programación que el alumno considere apropiado (teniendo en cuenta la disponibilidad de lenguajes en el laboratorio).

Trabajos prácticos
Se realizarán aplicaciones simples en las cuales el alumno deberá ir aplicando los conocimientos adquiridos en la teoría y práctica.