Universidad Nacional de San Agustín
VICE RECTORADO ACADÉMICO
SILABO

CODIGO DEL CURSO: CS101F

1 Datos Generales

FACULTAD : Ingeniería de Producción y Servicios						
DEPARTAMENTO : Ingeniería de Sistemas e Informática			**ESCUELA :** Ciencia de la Computación			
PROFESOR : **TÍTULO :**						
ASIGNATURA : Introducción a la Programación						
PREREQUISITO: Ninguno	**CREDITOS:** 4		**Año**: 2010-1 **Sem:** 1^{er} Semestre.		**Total Horas:** 2 HT; 2 HT	2 HP 2 HL
Horario	Lun	Mar	Mie	Jue	Vie	Sáb
Total Semanal						
Aula						

2 Exposición de Motivos

Este es el primer curso en la secuencia de los cursos introductorios a la informática. En este curs
los conceptos señalados por la *Computing Curricula* IEEE-CS/ACM 2008, bajo el enfoque *functio*
La programación es uno de los pilares de la informática; cualquier profesional del área, necesit
concretizar sus modelos y propuestas.
Este curso introducirá a los participantes en los conceptos fundamentales de este arte. Lo tópic
datos, estructuras de control, funciones, listas, recursividad y la mecánica de la ejecución, prueba
El curso también ofrecerá una introducción al contexto histórico y social de la informática y una
de esta disciplina.

2 Objetivo

- Introducir los conceptos fundamentales de programación y estructuras de datos utilizando un lenguaje funci
- Desarrollar su capacidad de abstracción, utilizar un lenguaje de programación funcional.

3 Contenido Temático

Objetivos Específicos	**Conte**
	3 SP/Historia de la Computación.(4 horas)
▪ Listar las contribuciones de varios pioneros en el campo de la computación.	▪ H 1
▪ Comparar la vida diaria antes y después del advenimiento de las computadoras personales e Internet.	▪ H t
▪ Identificar las tendencias continuamente significativas en la historia del campo de la computación.	▪ H [1], [3],

3 PL/Visión General de los Lenguajes de Programación.(1 horas)

Objetivos Específicos	Conten
▪ Listar la evolución de los lenguajes de programación identificando como es que su historia nos ha conducido a los paradigmas actuales. ▪ Identificar al menos una característica distintiva para cada uno de los paradigmas de programación cubiertos en esta unidad.	▪ Hi ma ▪ Pa [3], [2], [

3 PL/Declaración y Tipos.(1 horas)

Objetivos Específicos	Contenidos
▪ Explicar el valor de los modelos de declaración, especialmente con respecto a la programación en mayor escala. ▪ Identificar y describir las propiedades de una variable, tales como su: dirección asociada, valor, ámbito, persistencia y tamaño. ▪ Discutir la incompatibilidad de tipos. ▪ Demostrar las diferentes formas de enlace, visibilidad, ámbito y manejo del tiempo de vida. ▪ Defender la importancia de los tipos y el chequeo de tipos para brindar abstracción y seguridad. ▪ Evaluar las ventajas y desventajas en el manejo del tiempo de vida (conteo por referencia vs. recolección de basura).	▪ La concepción de tipos como un conjunto de valores unidos a un conjunto de operaciones. ▪ Vista general del chequeo de tipos. [3], [2], [4]

3 PF/Construcciones fundamentales.(2 horas)

Objetivos Específicos	Contenidos
▪ Analizar y explicar el comportamiento de programas simples involucrando las estructuras de programación fundamental cubiertas por esta unidad. ▪ Modificar y extender programas cortos que usan condicionales estándar, estructuras de control iterativas y funciones. ▪ Diseñar, implementar, probar y depurar un programa que use cada una de las siguientes estructuras fundamentales de programación: cálculos básicos, entrada y salida simple, estructuras estándar condicionales e iterativas y definición de funciones. ▪ Escoger la estructura apropiada condicional e iterativa para una estructura de programación dada. ▪ Aplicar técnicas de descomposición estructurada o funcional para dividir un programa en pequeñas partes. ▪ Describir los mecanismos de paso de parámetros.	▪ Sintaxis básica y semá lenguaje de más alto ni ▪ Variables, tipos, expresi naciones. ▪ Entrada y salida simple ▪ Estructuras de control c e iterativas. ▪ Funciones y paso de par ▪ Descomposición estruct [3], [2], [4]

3 PL/Programación Funcional.(1 horas)

Objetivos Específicos	Contenidos
▪ Delinear las fortalezas y debilidades del paradigma de programación funcional. ▪ Diseñar, codificar, probar y depurar programas usando el paradigma funcional. ▪ Explicar el uso de funciones como datos, incluyendo el concepto de cerraduras.	▪ Panorama general y motivació los lenguajes funcionales. ▪ Recursión sobre listas, número turales, árboles y otros datos d dos recursivamente. ▪ Pragmáticas (depuración en d y vencerás, persistencia de la tructuras de datos). [3], [2], [4]

	Objetivos Específicos	Contenidos	Hora
3 PF/Recursividad.(6 horas)	▪ Describir el concepto de recursividad y dar ejemplos de su uso. ▪ Identificar el caso base y el caso general de un problema definido recursivamente. ▪ Comparar soluciones iterativas y recursivas para problemas elementales tal como factorial. ▪ Describir la técnica dividir y conquistar. ▪ Implementar, probar y depurar funciones y procedimientos recursivos simples. ▪ Describir como la recursividad puede ser implementada usando una pila. ▪ Determinar cuando una solución recursiva es apropiada para un problema.	▪ El concepto de recursividad. ▪ Funciones matemáticas recursivas. ▪ Funciones recursivas simples. ▪ Estrategias de dividir y conquistar. [3], [2], [4]	

3 AL/Algoritmos Fundamentales.(4 horas)

Objetivos Específicos	Contenidos
▪ Implementar los algoritmos cuadráticos más comunes y los algoritmos de ordenamiento $O(NlogN)$. ▪ Diseñar e implementar una función de (*hash*) apropiada para una aplicación. ▪ Diseñar e implementar un algoritmo de resolución de colisiones para tablas de *hash*. ▪ Discutir la eficiencia computacional de los principales algoritmos de ordenamiento, búsqueda y (*hashing*). ▪ Discutir otros factores, además de la eficiencia computacional, que influyen en la elección de los algoritmos, tales como tiempo de programación, mantenimiento y el uso de patrones específicos de aplicación en los datos de entrada. ▪ Resolver problemas usando los algoritmos de grafos fundamentales, incluyendo búsqueda por amplitud y profundidad; caminos más cortos con uno y múltiples orígenes, cerradura transitiva, ordenamiento topológico y al menos un algoritmo de árbol de expansión mínima. ▪ Demostrar las siguientes capacidades: evaluar algoritmos, seleccionar una opción de un rango posible, proveer una justificación para tal elección e implementar el algoritmo..	▪ Algoritmos numéricos simpl ▪ Búsqueda secuencial y bina ▪ Algoritmos cuadráticos de miento (selección, inserción ▪ Árboles de búsqueda binari ▪ Recorridos por amplitud y didad. [3], [2], [4]

3 PL/Mecanismos de Abstracción.(4 horas)

Objetivos Específicos	Contenidos
▪ Explicar como los mecanismos de abstracción soportan la creación de componentes de software reusables. ▪ Demostrar la diferencia entre paso de parámetros por valor y por referencia. ▪ Defender la importancia de la abstracción especialmente con respecto a la programación en mayor escala.	▪ Procedimientos, funciones e res como mecanismos de ción. ▪ Mecanismos de parametriza ferencia vs. valor). ▪ Tipos de parámetros y tip metrizados. ▪ Módulos en lenguajes de p ción. [3], [2], [4]

3 PF/Algoritmos y Resolución de Problemas.(10 horas)

Objetivos Específicos	Contenidos
▪ Discutir la importancia de los algoritmos en el proceso de solución de problemas. ▪ Identificar las propiedades necesarias de un buen algoritmo. ▪ Crear algoritmos para resolver problemas simples. ▪ Usar pseudocódigo o un lenguaje de programación para implementar, probar y depurar algoritmos para resolver problemas simples. ▪ Describir estrategias útiles para depuración.	▪ Estrategias p blemas. ▪ El rol de los a de solución d ▪ Estrategias d algoritmos. ▪ Estrategias d ▪ El Concepto ritmos. [3], [2], [4]

3 PL/Máquinas Virtuales.(1 horas)

Objetivos Específicos	Contenidos
▪ Describir la importancia y poder de la abstracción en el contexto de máquinas virtuales.	▪ El concepto de máquina virtual. [3], [2], [4]

3 PL/Programación Orientada a Objetos.(4 horas)

Objetivos Específicos	Contenidos
▪ Diseñar, implementar y probar la implementacion de la relación es-un *IsKindOf* entre objetos usando jerarquía de clases y herencia. ▪ Comparar y contrastar las nociones de sobrecarga y sobreescritura de métodos en un lenguaje de programación.	▪ Clases y subclases. ▪ Polimorfismo (poli tipo vs. herencia). ▪ Jerarquías de clase [3], [2], [4]

3 SE/Usando APIs.(2 horas)

Objetivos Específicos	Contenidos	Hora
▪ Explicar el valor de las interfaces para programación de aplicaciones (APIs) en el desarrollo de software.	▪ Programación usando API. [3], [2], [4]	

4 Actividades

- Asignaciones
- Controles de Lectura
- Exposiciones

5 Recursos Materiales

- Apuntes del curso
- Libro(s) de la bibliografía

6 Metodología

- Clase Magistral.

- Taller didáctico.
- Social Constructivismo.
- Prácticas personales y en grupo.

7 Evaluación

La nota final (NF) se obtiene de la siguiente manera:

NE Nota de Exámenes 60 %, esta nota se divide en
- Exámen Parcial 40 %
- Examen Final 60 %

NT Nota de Trabajos e Intervención en clase 40 %

$$NF = 0{,}6 * NE + 0{,}4 * NT$$

Referencias

[1] J. G. Brookshear. *Computer Science: An Overview*. Addison-Wesley, 10th edition, January 2008. 0321524039.

[2] John V. Guttag. *Introduction To Computation And Programming Using Python*. Mit Press, 2013 edition, 2013.

[3] Simon Thompson. *The Craft of Functional Programming, 3E*. Addison Wesley, 2011.

[4] Jhon Zelle. *Python Programming: An Introduction to Computer Science*. Franklin, Beedle Associates Inc, 2nd edition, 2010.

Docente del curso