NOMBRE DE LA INSTITUCIÓN	Universidad de Sonora
DIVISIÓN ACADÉMICA	División Ciencias Exactas y Naturales
DEPARTAMENTO QUE IMPARTE LA MATERIA	Departamento de Matemáticas
LICENCIATURAS USUARIAS	Ciencias de la Computación
NOMBRE DE LA MATERIA	**Lenguajes de Programación**
CLAVE	**9452**
EJE FORMATIVO	Profesional
REQUISITOS	Teoría de la Computación, Estructuras de Datos.
CARÁCTER	Obligatorio
VALOR EN CRÉDITOS	8 (3 teoría/ 2 taller)

Introducción

Los Lenguajes de Programación son herramientas de software que permiten al ser humano interactuar con dispositivos de cómputo para procesar información, motivo por el cuál el área que se dedica al estudio de Lenguajes de Programación siempre ha sido una de las más importantes en el desarrollo de las Ciencias de la Computación. Para su creación y diseño, es fundamental comprender los aspectos teóricos que permiten aplicar técnicas eficientes que atiendan a la problemática que se estudia en los distintos paradigmas de programación.

Objetivo General del Curso

El estudiante adquirirá las bases teóricas y prácticas para que diseñe, desarrolle y opere con Lenguajes de Programación en sus distintas modalidades.

Objetivos Específicos del Curso

- El alumno conocerá los paradigmas principales de programación contemporáneos.
- El alumno adquirirá los conocimientos fundamentales de diseño e implementación de los lenguajes de alto nivel, incluyendo los conceptos de ligado, chequeo de tipos y administración de memoria durante ejecución.

Contenido

1. Historia y evaluación de los lenguajes de programación.

1.1. Primeros lenguajes: Algol, Fortan, Cobol.
1.2. La evolución de los lenguajes orientados a procedimientos (las cadenas de desarrollo de Algol PL/I, Pascal, Euclid, Modula y Ada).
1.3. Paradigmas y lenguaje no orientados a procedimientos: funcionales (LISP), lógicos (Prolog), orientados a objetos (Smalltalk) y paralelos (Occam).

2. Organización de la computadora a nivel de ensamblador.

2.1. Organización básica: Von Neumann, diagramas de bloque, rutas para datos, ruta de control, unidades funcionales (i.e. Unidad Aritmético-Lógica, memoria, registros), ciclos de instrucción.
2.2. Conjuntos y tipos de instrucciones.
2.3. Lenguaje ensamblador/de máquina.
2.4. Modos de direccionamiento (i.e. directo, indirecto, de desplazamiento, de registro, indexamiento).
2.5. Unidad de control; carga y ejecución de la instrucción: carga del operando.
2.6. Interrupción de entrada/salida.
2.7. Instrumentación por alambrado.
2.8. Instrumentación por microprogramación: formatos y codificación.
2.9. Máquina virtuales
2.10. Máquinas virtuales para los lenguajes de programación

2.11. Jerarquía de máquinas virtuales, presentadas al usuario a través del programa, el traductor, el sistema operativo, etc.
2.12. Consecuencias para la traducción de los distintos momentos en los que se hace el ligado.

3. Control de secuencia.
3.1. Expresiones, orden en la evaluación, efectos laterales.
3.2. Enunciados: simples y compuestos
3.3. Subprogramas y corrutinas como una abstracción de expresiones y enunciados.

4. Control de los datos, maneras de compartirlos, chequeos de tipo.
4.1. Mecanismos para compartir y restringir el acceso a datos (estructura de bloques, COMMOM, ADTs y aliases).
4.2. Rangos estáticos vs. dinámico, extensión, visibilidad.
4.3. Mecanismo para el paso de parámetros: Por referencia, por valor, por nombre, por resultado, etc.
4.4. Variedad en las disciplinas para el chequeo de tipos y sus mecánicas; estática vs. Dinámica vs. sin tipo, explícita vs. implícita, polimorfismo vs. Sobrecarga.

5. Manejo del espacio de almacenamiento durante ejecución
5.1. Asignación estática.
5.2. Asignación basada en un stack y su relación con la recursividad.
5.3. Asignación basada en una estructura de heap.

6. Paradigma de programación
6.1. Revisión de los paradigmas y lenguajes funcional, lógico y orientado a objetos.
6.2. Diseñar programas con estos paradigmas; ambientes de ejecución, flujo de Control.
6.3. Programas ejemplo y aplicaciones.
6.4. Ventajas y desventajas.

7. Diseños de Lenguajes: Semántica.
7.1. Una máquina sencilla y el modelo denotacional.
7.2. Tipos, vinculación operadores y coerción.
7.3. Asignación de memoria.
7.4. Estructuras de control.
7.5. Procedimientos y parámetros.

8. Diseños de Lenguajes: Pragmática.
8.1. El arte y ciencia del diseño de lenguajes.
8.2. El arte y ciencia de la programación.
8.3. Entorno de programación.
8.4. Comparación y evaluación de lenguaje.
8.5. Conclusiones.

Estrategias Didácticas

- Promover en los estudiantes la investigacion sobre distintos Lenguajes de Programación.
- Promover la participación activa de los estudiantes en el desarrollo de sus propios codigos en diversos Lenguajes de Programación.
- Promover la investigacion y desarrollo de aplicaciones de estrucuras de datos a areas de procesamiento de lenguajes.
- Desarrollar un sistema de software donde se apliquen aspectos teóricos del curso.
- Promover la participación grupal
- Desarrollar los conocimientos basicos para interconectividad de Lenguajes y reutilización de código

Estrategias de Evaluación

Para la evaluación de los estudiantes, el profesor tomará en cuenta:

- Tareas, trabajos de investigación, presentaciones en público.
- Resultados de los exámenes parciales (se sugiere que al menos sean tres).
- Desarrollo de un trabajo final.

Los criterios de aprobación del curso deberán de ser presentados al inicio del semestre.

Bibliografía

- Frieman, D. P.; Wand, M. and Heynes, C.T*. Essentials Of Programming Languages.* The MIT Press, 1992.
- Sethi, R. R. *Programming Languages*, *Concepts and Constructs.* Addison-Wesley Publishing Company, 1989.
- Scragg, G. W. *Computer Organization,* A *Top-Down Approach* McGraw-Hill Publishing Company, Inc., 1992.
- Budd, T. *An Introduction To Object-Oriented Programming.* Addison –Wesley Publishing Company, 1991.
- Field, A.J. and Harrison, P.G*. Functional Programming*. Addison-Wesley Publishing Company, 1991.
- Friedman, L. W. *Comparative Programming Language, Generalizing The Programming Function.* Prentice Hall, Inc., 1991.
- Kogge, P. M. *The Architecture of Symbolic Computers*. Mc.Graw-Hill Incorporated, 1991.
- Tucker, Jr., A. B. *Lenguajes de programación*. McGraw-Hill, 1987.

Perfil Académico Deseable del Maestro

Se recomienda que el profesor tenga las siguientes características:

- Formación sólida en el área de Ciencias de la Computación o área afín de forma tal que sea capaz de dar un panorama del uso de los Lenguajes de Programación dentro de las distintas áreas de las Ciencias de la Computación.
- Experiencia en la programación en varios Lenguajes.