Universidad Católica "Nuestra Señora de Asunción"
Sede Regional Asunción
Facultad de Ciencias y Tecnología

Departamento de Ingeniería Electrónica e Informática
Carrera de Ing. Electrónica, Ing. Informática

LENGUAJES DE PROGRAMACIÓN 3

CÓDIGO:	CYT646
CARRERA:	Ing. Informática
SEMESTRE:	6º
CORRELATIVAS:	Lenguajes de Programación 2
CARGA HORARIA SEMANAL:	5 horas
HORAS TOTALES:	90 horas
HORAS TEÓRICAS:	60 horas
HORAS PRÁCTICAS:	30 horas

DESCRIPCIÓN DEL CURSO:

Presentar los aspectos generales relacionados al diseño e implementación de un lenguaje de Programación. Estudio de cualidades de diseño y valoración de los lenguajes de programación.

OBJETIVOS:

Proporcionar al alumno una base teórica y práctica para sopesar y valorar las características de cualquier lenguaje de programación. Reforzar los conceptos de programación estructurada, funcional y de orientación a objetos. Ayudar al alumno al entendimiento y aprendizaje rápido de los lenguajes de Programación.

SÍNTESIS DEL PROGRAMA:

Evolución de los lenguajes de programación. El diseño de un lenguaje de programación. Definición de la sintaxis. Variables, expresiones y sentencias. Tipos de datos. Ámbito y alcance (scope y extent). Procedimientos. Programación orientada a objetos. Manejo de excepciones. Concurrencia. Estudio detallado y comparativo de lenguajes de programación

PROGRAMA ANALÍTICO

PARTE I: TEORÍA DE LOS LENGUAJES DE PROGRAMACIÓN

1. Evolución de los lenguajes de programación.

Historia de los lenguajes de programación. Primeros algoritmos. Historia antigua, historia reciente. Los años 60. Fortran y algol. Lenguajes de programación algol-like. El advenimiento de pascal y ada. Clasificación de los lenguajes de programación.

2. El diseño de un lenguaje de programación.

Criterios para el diseño de lenguajes. Definición semántica y sintáctica. Confiabilidad. Traducción eficiente. Código objeto eficiente. Ortogonalidad. Independencia del hardware. Probabilidad. Generalidad. Consistencia y notación común. Subconjuntos. Uniformidad. Extensibilidad.

3. Definición de la sintaxis.

Juego de caracteres. Bnf y grafos de sintaxis. Relación entre la definición sintáctica y la confiabilidad.

4. Variables, expresiones y sentencias.

Variables y la sentencia de asignación. Semántica axiomática. Definición de variable. Binding de variables. Conceptos de manejo de memoria. Stack y heap. Constantes e inicialización. Declaración de constantes. Manejo de las constantes en los diferentes lenguajes. Decisiones de diseño al implementar constantes. Expresiones. Diferentes notaciones para las expresiones aritméticas. Sentencias de control: if, elif, case. Sentencias iterativas: for, while, do while, loop, etc. Decisiones de diseño al implementar las diferentes sentencias iterativas.

5. Tipos de datos.

Definición de tipos de datos. Tipos de datos enumerados. Tipos de datos simples: números, datos booleanos, caracteres, cadenas. Apuntadores. Tipos de datos estructurados: arreglos y estructuras. Conversión de tipos de datos. Equivalencia de tipos.

6. Ámbito y alcance (scope y extent).

Definición de scope y extent. Tipos de scope. Scope estático, scope dinámico, relación entre los traductores y el scope. Definición de extent y su relación con su extent. Implementación en tiempo de ejecución.

7. Procedimientos.

Definición y características generales. Paso de parámetros y evaluación de parámetros. Paso por valor, por referencia, por valor resultado. Call-by-name. Call-by-text. Otros métodos de compartir variables. Paquetes y clases. Aliasing. Overloading. Funciones genéricas. Co-rutinas.

8. Programación orientada a objetos.

Conceptos básicos. Historia. Lenguajes orientados a objetos. Clases y objetos. Principios de ocultamiento de información y encapsulamiento de datos. Miembros de una clase. Visibilidad. Herencia. Interfaces. Funciones virtuales. Apuntadores y objetos.

9. Manejo de excepciones.

Aspectos de diseño. Concepto de excepción. Modelos de manejo de excepciones. Decisiones de diseño de excepciones. Implementación de excepciones en diferentes lenguajes.

10. Concurrencia.

Semáforos. Monitores. Aspectos de diseño. Paso de mensajes.

PARTE II: ESTUDIO DETALLADO Y COMPARATIVO DE LENGUAJES DE PROGRAMACION

Durante esta parte, se toman lenguajes particulares y se los analiza y aplica a la luz de los conceptos anteriormente presentados. Normalmente el alumno debe tomar algún lenguaje de programación. Presentar un tutorial breve sobre el mismo y realizar una práctica de laboratorio con dicho lenguaje.
Dependiendo de la cantidad de grupos formados, el número de lenguajes que se deben presentar varía entre 4 y 9. Los alumnos están obligados a estudiar y valorar los diferentes lenguajes. Ejemplos de lenguajes utilizados durante las clases: SmallTalk, Eiffel, Visual Basic, Java, PHP, Perl, Ruby, Python, C#, Awk, Lisp, Assembler, etc.

METODOLOGÍA:

Desarrollo de las características de los lenguajes de programación. Utilizando exposiciones a cargo de los docentes o invitados.
Durante las horas prácticas, los alumnos requieren la presentación de un lenguaje de Programación cuya metodología puede variar entre un lenguaje y otro. Las alternativas son: presentación, ejemplificación, pruebas guiadas o ejercicios de verificación.

BIBLIOGRAFÍA PRINCIPAL:

1. "Programming Language Design Concepts". David Watt.

BIBLIOGRAFÍA ALTERNATIVA:

2. "Programming Language Pragmatics". Michael Scott
3. "Programming Languages: Principles and Paradigms". Maurizio Gabbrielli, Simone Martini.

REDACCIÓN ORIGINAL:

Ing. Mauricio Kreitmayr

ÚLTIMA REVISIÓN:

Ing. Mauricio Kreitmayr, Agosto/2016

APROBADO POR CONSEJO DE DEPARTAMENTO EN FECHA:
25 de octubre del 2004, mediante nota Nro. 120/04

APROBADO POR CONSEJO DE FACULTAD EN FECHA:
16 de diciembre del 2004, mediante acta Nro. 12/04