Taller de Lenguajes I

Carrera: ***Ingeniería en Computación***

Año: ***2°***
Duración: ***Semestral***
Profesor: ***Lic. Laura Lanzarini***
Hs. semanales: **6 hs.**

OBJETIVOS GENERALES:

Profundizar los conocimientos obtenidos por el alumno en los primeros cursos vinculados con Algoritmos y Programación, permitiéndole desarrollar un estudio teórico-práctico de algún lenguaje de programación procedural (el lenguaje puede variar con los cambios tecnológicos), poniendo énfasis en el análisis formal de las características del lenguaje y su comparación con los que el alumno conociera a ese momento (típicamente Pascal).

CONTENIDOS MINIMOS:

Estudio de un lenguaje de programación (como por ejemplo C, que resulta adecuado para la carrera) en el que se desarrollen aplicaciones concretas.

Programa

1. Introducción

La historia de C. ANSI C. Características de C. La estructura de un programa en C. Identificadores. Alcance de un identificador. Cómo trabaja el compilador C. Análisis de un par de programas sencillos en C. Comentarios. Caracteres especiales. La biblioteca estándar de C. Sentencias printf y scanf.

2. Tipos de Datos

Tipos de datos simples. Tipos enteros con una cantidad exacta de bits. Tipos de punto flotante. Conversión de tipos. Conversión de tipos aritméticos. Conversión aritmética usual. Otras conversiones implícitas. Conversión explícita de tipos.

3. Expresiones, Operadores y Sentencias

Evaluación de expresiones. Operadores de asignación. Operadores incrementales y decrementales.

Sentencia. Bloque de sentencias. Iteraciones. Sentencias while, for y do-while. Loops anidados. Sentencias de selección. Sentencias *if*, *if-else*, operador condicional y *switch*. Saltos incondicionales. Sentencias *break, continue, return*

4. Funciones e identificadores

Definición, declaración e invocación de funciones. Prototipo de una función. Coerción de tipos. Retorno de una función. El tipo *void*; su uso en declaración de funciones. Funciones recursivas.

Clases de almacenamiento de un identificador. Persistencia automática: *auto* y *register*. Persistencia estática: *static* y *extern*.

5. Arreglos

Arreglos de longitud fija y arreglos de longitud variable. Definición. Acceso a los elementos del arreglo. Inicialización del arreglo. Vector de caracteres. Funciones para cadenas de caracteres. Arreglos y el especificador *static*. Arreglos multidimensionales. Arreglos como argumento de una función.

6. Punteros

Declaración de punteros. Inicialización de punteros. Identificador *null*. Operadores de punteros. Visualización del valor de un puntero. Pasaje de parámetros por referencia.
El calificador *const* aplicado a arreglos y a punteros.
Operaciones aritméticas con punteros. Asignación de punteros. Comparación entre punteros.
Arreglos de punteros y punteros a arreglos. Punteros a funciones.

7. Cadenas de caracteres

Definición. Declaración e inicialización. Funciones de manejo de caracteres (librería *ctype.h*). Funciones de conversión (librería *stdlib.h*). Funciones de entrada/salida (*stdio.h*). Funciones de comparación, búsqueda y manipulación de cadenas (*string.h*).

8. Estructuras, uniones y enumeraciones

Estructuras. Definición. Declaración de variables. Inicialización. Operaciones. Acceso a los campos de una estructura. Punteros como miembros de una estructura.

Union. Definición de tipos Union. Operaciones sobre uniones.

Operadores a nivel de bits. Operadores de asignación a nivel de bits. Campos de bits.

Constantes de enumeración. Ejemplos.

9. Archivos

Streams. Dispositivos estándar: *stdin*, *stdout*, *stderr*. Archivos. Definición. Modos de apertura de un archivo. Posición dentro del archivo. Operaciones de alta, baja y modificación de archivos. Acceso secuencial y acceso directo. Manejo de errores de acceso. Archivos binarios.

10. Manejo de memoria dinámica

Alocación de memoria de manera dinámica. Redimensionado y liberación de memoria. Arreglos dinámicos. Ejemplificación creando listas. Matrices dinámicas. Alocación y liberación de memoria.

11. Directivas para preprocesador

Directiva #include. Acceso a los archivos indicados en la directiva #include.

Definición y uso de macros. Macros con y sin parámetros. Macros dentro de macros. Alcance y redefinición de macros. Macros predefinidas.

Directiva #undef. Compilación condicional. Las directivas #if y #elif. Directivas #ifdef e #ifndef.

12. Librerías

Headers estándar. Uso de headers. Contenido de los headers estándar. Funciones de la librería estándar según su área de aplicación. Aplicaciones formadas por varios archivos.

13. Herramientas básicas

Compilación con GCC. Warnings del compilador. Optimización Debugging.

Usando *make* para construir programas en C. Prerrequisitos y comandos. *MakeFile*. Reglas. Comenatrios. Variables. Macros. Funciones. Directivas. Ejecutando *make*.

Debugging de programas C utilizando GDB. Un ejemplo simple de una sesión de debugging.

Bibliografía

- C: How to program, 6th edition. Deitel y Deitel. Prentice Hall. 2010
- C in a Nutshell. Peter Prinz yTorry Crawford. Editorial O'Reilly Media. 2006.
- C estándar. British Standards Institute. Editorial Anaya Multimedia. 2005.
- Practical C. Programming. Steve Oualline. Editorial O'Reilly Media. 1997.
- The C Programming Language 2nd Edition. Kernighan y Ritchie. Prentice Hall Software Series. 1988.