



**UNIVERSIDAD RICARDO PALMA**  
**FACULTAD DE INGENIERIA**  
**EAP INGENIERIA INFORMATICA**

**Ciclo Académico 2003 – 2**

**SILABO**

**1. INFORMACIÓN GENERAL** (del plan de estudios).

**1.1. Datos administrativos.**

- 1.1.1. Curso: Taller de Programación II
- 1.1.2. Código: II 0305
- 1.1.3. Horas Teoría: 2 semanal.
- 1.1.4. Horas Taller: 6 semanales.
- 1.1.5. Créditos: 4.
- 1.1.6. Modalidad: obligatorio.
- 1.1.7. Ciclo: III.

- 1.2. Requisitos:** II 0205 Taller de programación I  
II 0206 Programación I.

- 1.3. Naturaleza:** taller.

- 1.4. Objetivos generales:** Al finalizar el curso el alumno será capaz de: Desarrollar programas usando técnicas orientadas a objetos (POO) para manejar datos (escalares, arreglos, abstractos lineales y no lineales). Cálculos numéricos básicos usando POO. Construcción de aplicaciones simples usando programación visual.

- 1.5. Contenido:** Programación tallerín. Programación estructurada multifuente. Clases y objetos (constructores y destructores implícitos y explícitos). Clases y algoritmos. Herencia de clases. Sobrecarga de operadores y conversión de objetos. Clases abstractas y concretas. Polimorfismo simple. Jerarquía de clases polifórmica. Manejo de archivos.

**2. OBJETIVOS ESPECIFICOS.**

- 2.1. Ubicar la POO en la evolución de las técnicas de programación, conociendo y manipulando las técnicas que lo antecedieron: tradicionales y estructuradas.
- 2.2. Aplicar conceptos fundamentales de la POO: clases y objetos, propiedades y métodos, mensajes, constructores y destructores en la construcción de programas.
- 2.3. Usar la sobrecarga de funciones y operadores en la programación.
- 2.4. Aplicar herencia, composición y polimorfismo de la POO.
- 2.5. Manipular datos como jerarquías de clases y objetos.
- 2.6. Aplicar plantillas en la definición de funciones y clases.
- 2.7. Manejar directamente archivos.

### **3. PROGRAMACIÓN DE CONTENIDOS.**

- 3.1. Organización y ambientes de programación.
- 3.2. Programación Tallerín.
- 3.3. Programación Estructurada Multifuente.
- 3.4. Clases y Objetos con constructores y destructores implícitos.
- 3.5. Clases y Objetos con constructores y destructores explícitos.
- 3.6. Clases y algoritmos.
- 3.7. Herencia y especialización.

#### **3.8. [Examen parcial]**

- 3.9. Sobrecarga de funciones y operadores.
- 3.10. Sobrecarga y cambio de tipos de objetos.
- 3.11. Clases abstractas y concretas.
- 3.12. Polimorfismo.
- 3.13. Plantillas de funciones y clases.
- 3.14. Manejo de archivos.
- 3.15. Recuperación
- 3.16. Entrega de notas

### **4. PROCEDIMIENTOS DIDÁCTICOS.**

- 4.1. Análisis y síntesis en el desarrollo de programas para resolver ejercicios y solucionar problemas.
- 4.2. Dialogo y exposición en la presentación de los talleres y de la sintaxis del lenguaje de programación, usando materiales y equipos disponibles.
- 4.3. Tutoría.
- 4.4. Taller guiado - construcción de programas- en ambiente de programación.
- 4.5. Absolución de preguntas y reforzamiento continuos.

### **5. EQUIPOS Y MATERIALES.**

- 5.1. Computador con software de presentación y video-proyector para presentación de talleres y sintaxis del lenguaje de programación seleccionado.
- 5.2. Mientras no se dispongan de estos equipos, se usarán pizarra y tiza/plumón.
- 5.3. Guías de talleres.
- 5.4. Un computador o estación de trabajo por alumno con:
  - 5.4.1. Un ambiente interactivo de desarrollo de programas en C/C++.
  - 5.4.2. Un ambiente visual de construcción de programas en visual C/C++.
  - 5.4.3. Acceso a Internet.

### **6. EVALUACIÓN.**

- 6.1. Continua, orientada a contrastar los conocimientos, habilidades, destrezas y actitudes adquiridos por el estudiante, dentro del marco normativo establecido por la Universidad.
- 6.2. Formativa directa e indirecta. Directa, durante los diálogos y a través de las preguntas y respuestas verbales, e indirecta, vía aplicación de pruebas.
- 6.3. Cada taller tiene su propia evaluación acumulativa (Ti); producida directa e indirectamente: directa, durante el taller, e indirecta, a través de los informes y programas producto.
- 6.4. La evaluación acumulativa (TT), es el promedio de las pruebas escritas aplicadas durante las clases de teoría del taller.

- 6.5. La nota final del curso es el promedio numérico de la evaluación acumulativa, donde la nota de cada taller (Ti) tiene peso simple y la nota relacionada con 12

$$\text{Nota Final} = \frac{\sum_{i=1} T_i + 2(TT)}{14}$$

## 7. BIBLIOGRAFÍA.

- 7.1. Docente, Guía por taller, 2da Ed. E.A.P. Ing. Informática, Lima 2000.
- 7.2. Borland Int., Manuales del C/C++, versión 4.5. Borland International, USA.
- 7.3. Microsoft, Manuales visual C++, 6. Microsoft Intemational, USA, 1998.
- 7.4. Microsoft, MSDN, Help del Visual Estudio. USA, 1999
- 7.5. Ceballos F.J, Programación orientada a objetos con C++. Rama, España 1993.
- 7.6. Sedgewick R., Algoritmos en C++. Addison-Wesley Iberoamericana, USA, 1992.
- 7.7. Joyanes A.L., Programación orientada a objetos Ira Ed. McGraw-Hill Interamericana, Fspaña, 1996.
- 7.8. Kryglinski D. J., Progrese con visual C++, Ira Ed. McGraw-Hill Interamericana, España, 1993.
- 7.9. Ceballos F.J, Visual C++. Aplicaciones para windows. Rama y Alfaomega México, 1997