Universidad Católica de Córdoba

Facultad de Ingeniería

Computación 3 - Año 2002 - 2do semestre

Titular: Ing. John Coppens

1 Objetivos específicos

Tres objetivos básicos:

1.1 Pascal

Avanzar en los conocimientos de la programación en Pascal. Técnica avanzadas como programación con objetos, uso de punteros. Énfasis sobre la resolución de problemas, y la expresión de situaciones reales en un programa de computación. Imperativo: el uso de programación en módulos funcionales. Programación estructurada. Resolución de problemas complejos. Programación gráfica.

1.2 C

Comparación de los idiomas C y Pascal. Introducción a los conceptos de 'compilación', 'linkeado', 'librerias de ejecución', 'interpretadores'. Es importante recalcar que el análisis y la expresión de los problemas en la vida real es más importante y anterior que el idioma de programación. Catalogación de los diferentes idiomas de programación, y ubicación de herramientas de desarrollo (los idiomas 'visual', herramientas RAD (Rapid Application Development), y tendencias futuras).

Mostrar las equivalencias y diferencias entre Pascal y C. La característica de autodocumentación en Pascal, y los peligros de programas crípticos en C. Ejemplos de utilización de C. Consejos de estilo.

1.3 Assembler

Introducción a assembler de procesadores actuales. Terminología: Conjunto de instrucciones, programación en binario, lenguaje assembler. Introducción básica a la arquitectura de una PC moderna. Ejemplos de programación en assembler.

2 Programa sintético

2.1 Unidad 1 - Pascal

2.1.1 Repaso e introducción

Repaso de los conocimientos anteriores, descripción de la materia. Metodología de trabajo. Prácticos. Donde obtener la información y bibliografía. Guía de estilo y presentación de trabajos.

2.1.2 Programación modular: Unidades

Subdivisión de problemas complejos en módulos. Uso de units. Interrelación de unidades, definición. Ventajas. Problemas.

2.1.3 Asignación dinámica: Punteros

Limitación de memoria en la arquitectura de 16 bit. Consecuencias para los programas. El límite de 64kB. Solución por asignación dinámica de memoria (punteros). Aplicación de punteros en listas enlazadas. Aún más memoria: diferencias en compiladores de 32 bits.

2.1.4 Programación estructurada: Objetos

Extensión de records a objeto. La finalidad de los objetos - ejemplos de aplicación de objetos. Herencia. Objetos virtuales. Generación de objetos dinámicos.

2.2 Unidad 2 - C

2.2.1 Idioma de programación C

Historia. Área de aplicación. Herramientas disponibles. Ventajas/desventajas. Los problemas de C y cual fué el motivo de utilizar Pascal en lugar de C para enseñanza.

2.2.2 Introducción a compiladores, linkeadores, ...

Como funciona un compilador? Tareas a realizar durante la compilación, y después. Impacto del idioma sobre los compiladores. Librería de ejecución.

2.2.3 Comparación de C - Pascal

Mediante ejemplos, mostrara las sentencias equivalentes de C y Pascal. Recalcar las diferencias de filosofía entre ambos.

2.3 Unidad 3 - Assembler

2.3.1 Que es una CPU?

Introducción al funcionamiento interno de procesadores en general. Que idioma entiende la CPU? Como y desde donde recibe la información? Como la procesa?

2.3.2 Evolución de binario a Assembler, a idiomas de alto nivel

Impracticidad de programación en binario. 'Idioma' assembler. Como escribir un programa en assembler y convertirlo en binario (ejemplos con debug, y con compiladores assembler simples).

Comparación de la efectividad de assembler con los compiladores de alto nivel, con sus optimizadores.

3 Bibliografía obligatoria

- *Algorithms + Data structures = Programs*
  Niklaus Wirth, Prentice Hall, ISBN 0-13-022418-9
- *Algoritmos + Estructuras de Datos = Programas*
  Niklaus Wirth, C.I.E / DOSSAT 2000, ISBN: 8421901729

4 Bibliografía de consulta

- *Programming and Problem Solving in Pascal*
  Schneider, G. Michael; John Wiley & Sons, ISBN: 0-471-08216-3
- *Advanced Programming and Problem Solving in Pascal*
  Schneider, G. Michael, Bruell, Steven C.; John Wiley & Sons, ISBN: 0-471-0 1128-2
- *Programación en Turbo Pascal*
  Joyanes Aguilar, Luis.; McGraw Hill, ISBN: 8448110757
- *El lenguaje de Programación C*
  Kernighan, B., Ritchie, J.; Prentice Hall, ISBN 9688802050
- *The C Programming Language*
  Kernighan, B., Ritchie, J.; Prentice Hall, ISBN: 0131103628
- *The Art of Computer Programming: 1. Fundamental algorithms*
  Donald E. Knuth; Prentice Hall, ISBN: 0201896834
- *The Art of Computer Programming, 2. Seminumerical algorithms*
  Donald E. Knuth; Prentice Hall, ISBN: 0201896842
- *The Art of Computer Programming, 3. Sorting and searching*
  Donald E. Knuth; Prentice Hall, ISBN: 0201896850
- *PC Interno 5, Programación de Sistema*
  Tischer, Michael; Marcombo, ISBN: 8426710816
- *PC Intern: System Programming: The Encyclopedia of DOS Programming Know How*
  Tischer, Michael; Bk&Cd, ASIN: 1557551456 (Fuera de publicación)
- *GNU C++ for Linux*
  Tom Swan, Que, ISBN: 0-7897-2153-8

5 Metodología

En los años anteriores se mostró extremadamente productivo, el método de integrar estrechamente la parte práctico con la teoría. En cada clase se explica los conceptos nuevos, y luego se aplican inmediatamente en ejemplos y, posteriormente, en un proyecto progresivo.

Ya que es dificil de lograr un proyecto de envergadura durante la actual enseñanza del item, se presentaron inicialmente ejemplos de programación simples, y luego se

inicio, como ejemplo de una tarea mayor, con todos sus aspectos de análisis, representación, y resolución, de un proyecto grande. En los años anteriores logramos de esta forma implementar una hoja de cálculos (en modo texto) y un simulador de un procesador PIC.

6 Calendario

Fecha	Unidad	Tema/actividad
01/08	1.1	Repaso de los conocimientos, método de trabajo
08/08	1.2	Programación modular: Unidades
15/08 22/08	1.3	Asignación dinámica: Punteros
29/08 05/09	1.4 2.1	Programación estructurada: Objetos
12/09		*Primer Parcial*
19/09	2.1	Idioma de programación C
26/09 03/10	2.2	Introducción a compiladores, linkeadores, ...
10/10 17/10	2.3	Comparación de C - Pascal - Ejemplos
24/10	3.1	Que es una CPU?
31/10	3.2	Evolución de binario a Assembler, a idiomas de alto nivel
07/11		S*egundo Parcial*
14/11		Repaso de la Materia

7 Criterios y formas de evaluación

7.1 Durante el semestre

Tanto en la parte teórica como en la parte práctica, se presentarán por lo mínimo 3 trabajos durante el semestre. Estos trabajos se evaluarán por su: originalidad, solidez de implementación, estructura, documentación.

7.2 Parciales

En la parte teórica se realizarán además dos parciales, evaluando el conocimiento sobre la materia en forma individual.

En las notas finales se agregará una evaluación del la participación del alumno en las actividades.

7.3 Examen final:

El examen final teórico consistirá en dos partes: una evaluación de los conocimientos teóricos de la matería y una confirmación individual de la participación en la realización de los trabajos presentados en grupos. El énfasis en los exámenes estará sobre la capacidad de resolución de problemas y razonamiento.

8 Condiciones para obtener la regularidad

Las condiciones formales son las que actualmente rigen para todas las cátedras, según el reglamento vigente.

9 Trabajos prácticos

En el proceso descrito en Metodología (5.), se describió la forma de relacionar la teoría con la práctica. El método de la implementación permite, en forma natural, la delegación de sub-tareas del proyecto mayor, como trabajos prácticos para los alumnos.

Estos trabajos incluyen preparación (investigación y comparación de soluciones actuales para el proyecto propuesto), análisis del problema (conversión de un problema a diagrama de flujo o pseudo-idioma), e implementación real (programación del problema en Pascal o C).

Además, la implementación de un proyecto mayor recalca la importancia del trabajo entre grupos (entiéndase como teamwork), coordinación entre los mismos, problemas de compatibilidad, etc.

En los años anteriores se logró la implementación funcional de una hoja de cálculos en modo texto (MsDOS), y un simulador funcional de un procesador PIC.