# TALLER DE PROGRAMACIÓN MULTIPARADIGMA EN UN CURSO CS1. RESULTADOS Y DESAFÍOS.

# Taller de Programación multiparadigma en un curso CS1. Resultados y Desafíos.


## Ing. ARMANDO DE GIUSTI


Director del Instituto de Investigación en Informática LIDI


Investigador Principal CONICET

Profesor Titular de la Facultad de Informática UNLP

Miembro de la Academia de la Ingeniería PBA

# Taller de Programación multiparadigma en un curso CS1. Resultados y Desafíos.

- Temas Generales Curso CS1

Parte I

- Taller de Programación UNLP

Parte II

# Taller de Programación multiparadigma en un curso CS1. Resultados y Desafíos.

## AGENDA – Parte I

- ✓ El alumno y el graduado en Informática… el desafío de los extremos…
- ✓ Algoritmos, Datos y Programas. Un curso CS1 típico.
- ✓ Contexto curricular de un curso CS1 en un Plan de Estudios de Informática.
- ✓ Integración vertical: El Ingreso y Segundo Año.
- ✓ La decisión entre un curso CS1 anual y 2 cursos cuatrimestrales que empalman.
- ✓ El desafío de un Taller de Programación como segunda parte de un curso CS1.
- ✓ La difícil decisión entre un paradigma y múltiples paradigmas.
- ✓ Breves Conclusiones

# Taller de Programación multiparadigma en un curso CS1. Resultados y Desafíos.

## AGENDA – Parte II

- ✓ Taller de Programación multiparadigma: Objetivos y Desafíos.
- ✓ Cómo implementarlo? La experiencia de la UNLP.
- ✓ Discusión de la metodología.
- ✓ Discusión de los contenidos en relación con el curso de Conceptos de Datos, Algoritmos y Programas.
- ✓ Discusión de los mecanismos de Evaluación.
- ✓ La capacitación de los docentes.
- ✓ La discusión de los Lenguajes a utilizar.
- ✓ Resultados luego de 3 años. Breves Conclusiones.

Parte II

# El alumno y el graduado en Informática... el desafío de los extremos: **El alumno**

- A los nativos digitales les gusta el trabajo en paralelo y multitarea.
- Prefieren las imágenes al texto.
- Son interactivos por naturaleza.

# El alumno y el graduado en Informática… el desafío de los extremos: **El alumno**

- Los nativos digitales no pueden vivir sin estar conectados.
- Trabajan mejor cuando lo hacen en red.
- Prefieren los juegos al trabajo “serio”.
- Trabajan por prueba y error.

# El alumno y el graduado en Informática... el desafío de los extremos: **El graduado**

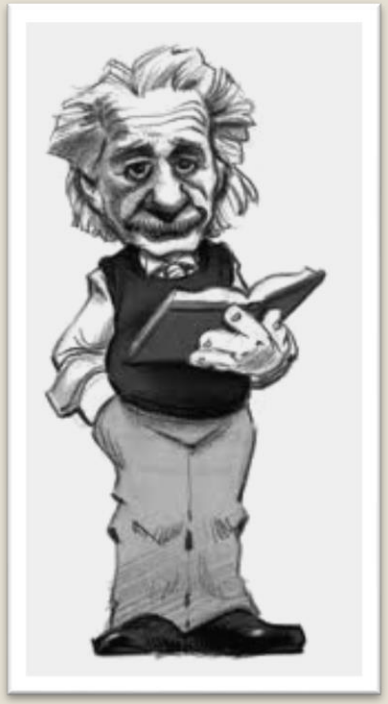
Conocimiento PREVIO

**+**

Cambio TECNOLOGICO

**+**

Ideas INNOVADORAS



Conocimiento NUEVO

**SIEMPRE**


**Aplicaciones en el MUNDO REAL**

# El alumno y el graduado en Informática. el desafío de **acercar** los extremos...

**Tránsito complejo entre perfiles**

# Algoritmos Datos y Programas.

## Un curso CS1 típico (Anual). Objetivos (version UNLP 2014)

- Analizar problemas resolubles con computadora, poniendo énfasis en la modelización, abstracción de funciones y en la descomposición funcional de los mismos.
- Obtener una expresión sintética y precisa de los problemas, con una documentación de una metodología de trabajo por el alumno.
- Estudio, expresión simbólica, implementación y evaluación de algoritmos, orientando los mismos a la resolución de las partes (módulos) en que se descomponen los problemas, a partir de un paradigma procedural/imperativo.
- Introducción de las nociones de estructuras de datos, tipos de datos y abstracción de datos.
- Introducción de los conceptos de corrección y eficiencia de algoritmos.
- Introducción de los conceptos básicos de un segundo paradigma de programación (orientación a objetos) con énfasis en la noción de re-usabilidad.
- Introducción de los conceptos básicos de la Programación Concurrente.
- Combinar los elementos mencionados anteriormente a fin de que el alumno complete el ciclo del problema a su solución con computadora, analizando simultáneamente algoritmos y datos.

# Algoritmos Datos y Programas.

## Un curso CS1 típico (Anual). Objetivos e Implementación

Densidad de los temas

1 año de duración

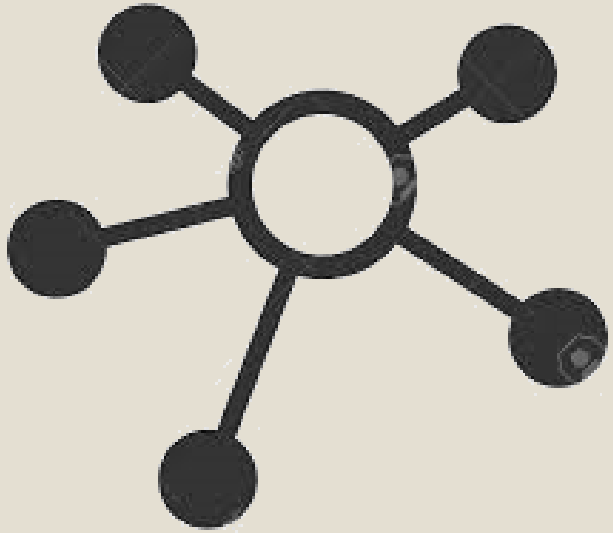
2 evaluaciones parciales prácticas. 1 evaluación integradora.
Fin de las cursadas se proyecta a Febrero del año siguiente.
Promoción teoría limitada a los alumnos con buenos resultados en el primer parcial.

Limitado. Mucho en papel y lápiz. Trabajos experimentales en máquina pensados para los alumnos en promoción.

Paradigma / Lenguaje imperativo. La introducción a POO y a Concurrencia eran más conceptuales que prácticas en máquina.

# Algoritmos Datos y Programas.

## Un curso CS1 típico (Anual).



## Relación con un plan de estudios clásico

Peso del trayecto Algoritmos y Lenguajes en un Plan de Estudios clásico (25 al 30%).

Normalmente el trayecto completo tiene entre 6 y 10 asignaturas cuatrimestrales adicionales relacionadas.

EL curso CS1 anual del primer año tiene mucho impacto en la deserción y también en las correlatividades.

Hay una correlación directa entre el éxito en un curso CS1 anual y la posibilidad de llegar al título en Informática (al menos en UNLP).

# Contexto curricular de un curso CS1 en un **Plan de Estudios de Informática.**

## Ciclo Inicial (Ingreso)

Expresión de Problemas y Algoritmos (60 hs)

## CS1 ANUAL

Algoritmos Datos y Programas (ADP)

## Segundo Año (correlativas)

Algoritmos y Estructuras de datos
Seminario de Lenguajes.
Ingeniería de Software 1.
Orientación a Objetos 1.
Fundamentos de Org. de Datos.
Introducción a los SO.

# Contexto curricular de un curso CS1 en un **Plan de Estudios de Informática.**

## Curso de Ingreso

Matemática 0

Conceptos de Organización de Computadoras

Expresión de Problemas y Algoritmos

Matemática 1

Matemática 2

Organización de Computadoras

Arquitectura de Computadoras

# Contexto curricular de un curso CS1 en un **Plan de Estudios de Informática.**

## Discusión de Objetivos: Cuatrimestralizar?

Segmentación del Aprendizaje.

Segmentación del perfil de los Alumnos.

Mejora del trabajo experimental en el 2do. cuatrimestre

Posibilidad de re-dictados y recuperación temprana de alumnos.

Posibilidad de cursos de Verano / Invierno

# Contexto curricular de un curso CS1 en un **Plan de Estudios de Informática.**

## Discusión de Objetivos: Cuatrimestralizar?

Respecto del curso anual, se pierde la posibilidad de recuperación tardía de los alumnos.

Menor tiempo de maduración de los temas.

Se pierde una evaluación final integradora de todos los conocimientos.

Imposible cubrir todos los temas teóricos en un cuatrimestre.

Qué hacer con los alumnos con dificultades en el 1er. Cuatrimestre?

# Contexto curricular de un curso CS1 en un **Plan de Estudios de Informática.**

En función de los objetivos de un curso CS1

En función del trabajo experimental.

**Balance: Cuatrimestralizar?**

En función de la formación hacia 2do. Año.

En función de la deserción de los alumnos.

# Integración vertical: CS1 y el Ingreso

## Objetivos de integración vertical en CS1

- La continuidad de contenidos entre el Ingreso y el curso CS1
- Los demos experimentales como motivación en el Ingreso.
- El trabajo experimental en el curso CS1.
- La dificultad de elegir un paradigma “único”.

# Integración vertical: CS1 y 2do año

## Objetivos de integración vertical en CS1

- La diversidad de contenidos en 2do. Año.
- Los cambios de paradigmas en 2do. Año.
- Los requerimientos de capacidad de trabajo experimental.
- Evitar la deserción en 2do. Año.

# Integración transversal: Procesadores y Sistemas Operativos

## Objetivos de integración horizontal en CS1

- La desaparición del modelo “Von Neumann”
- El paralelismo de los procesadores reales.
- Los requerimientos sobre el tema de concurrencia.
- El desarrollo del paradigma de Objetos.
- Los nuevos desafíos de la Ingeniería de Software.

# La decisión entre un curso CS1 anual y dos cursos cuatrimestrales que empalman

## ADP - ANUAL

- 800 ingresantes
- 400 recursantes
- 4 turnos presenciales anuales
- 1 turno semipresencial anual
- Unos 300 Trab. Pract. aprobados.
- Unos 500 alumnos que desertan.

## CADP - TP

- 800 ingresantes
- 200 recursantes en CADP
- 300 alumnos en Taller Prog. 2do. Sem.
- Posibilidad de redictado CADP 2$^{do}$ Sem.
- Posibilidad de redictado TP 1er. Sem.
- Mayor número de alumnos atendidos

# La decisión entre un curso CS1 anual y dos cursos cuatrimestrales que empalman

- 800 ingresantes en CADP 1er. Semestre.
- 200 recursantes adicionales en CADP
- 300 alumos en Taller de Programación 2do. Sem.
- Posibilidad de redictado CADP 2$^{do}$ semestre
- Posibilidad de redictado Taller de Prog. en el 1er. Sem.
- Mayor número de alumnos atendidos

**CADP + TP**

- Requerimientos de Recursos Humanos.
- Requerimientos de Infraestructura.
- Requerimientos en la Metodología.

# El desafío de un Taller de Programación como segunda parte de un curso CS1

Dónde “corta” un primer curso cuatrimestral?

Qué hacer con los temas teóricos no cubiertos en 1er semestre?

Cómo fortalecer el trabajo experimental en grupos reducidos?

Cómo hacer las evaluaciones de un Taller de Programación?

Qué resultado se tendrá con la deserción?

# La difícil decisión entre un paradigma y múltiples paradigmas.

Elegir un UNICO PARADIGMA (tradicionalmente imperativo u objetos) permite profundizar en el Taller el desarrollo de problemas experimentales en ese paradigma.

Elegir un único paradigma NO RESUELVE UN BUEN EMPALME con 2$^{do}$. Año (por los contenidos de los Seminarios de Lenguajes, de Algoritmos y Estructuras de Datos y de Introducción a los Sistemas Operativos).

De hecho el Ingreso apunta a la expresión de algoritmos en modo “imperativo”.

# La difícil decisión entre un paradigma y múltiples paradigmas.

Elegir MÁS DE UN PARADIGMA exige pensar en los requerimientos de los cursos de segundo año (tradicionalmente muy difícil en la UNLP).

Elegir más de un paradigma OBLIGA A RE-ESTRUCTURAR EL TALLER, ordenando el dictado de cada paradigma, con evaluaciones concretas en cada uno de ellos.

De hecho se debe trabajar también en la/s herramienta/s del Ingreso, para poder transitar desde el modo “imperativo” al modo “concurrente” sin un cambio sintáctico y semántico importante.

# La difícil decisión entre un paradigma y múltiples paradigmas.

Elegir más de un paradigma exige pensar en los requerimientos de los cursos.
Elegir el **paradigma imperativo + orientado a objetos + concurrente**
fue una decisión ambiciosa y un desafío desde varios puntos de vista.

Hay que resolver en un cuatrimestre una práctica con maquina significativa en tres paradigmas. Y una nueva modalidad de evaluación sobre máquina.

Hubo que mejorar sustancialmente la infraestructura (por esto se crean los Laboratorios Móviles).

Los Laboratorios Móviles a su vez tienen requerimientos sobre las Aulas.

La modalidad intensiva teórico-práctica fue un desafío significativo para los docentes de la cátedra.

# La difícil decisión entre un paradigma y múltiples paradigmas.

## Las dudas de un Taller experimental en CS1

Por qué el paradigma imperativo?

Por qué el paradigma orientado a objetos?

Por qué el paradigma concurrente?

Se mejoran las competencias buscadas en el alumno?

Se mejora la motivación de los alumnos?

Cuál es el efecto sobre segundo año para los alumnos?

# Taller de Programación multiparadigma en un curso CS1.

## Breves Conclusiones

Los efectos positivos del trabajo experimental para el alumno ingresante.

La importancia de aprender a resolver problemas con diferentes paradigmas y lenguajes de programación.

La conveniencia de una visión “integradora” del recorrido curricular desde el Ingreso al Taller de Programación.

La exigencia que un Taller de Programación multiparadigma para los equipos docentes.

La posibilidad de trabajar con los alumnos por cuatrimestre, con re-dictados.

# Taller de Programación multiparadigma en un curso CS1. Resultados y Desafíos.

## AGENDA – Parte II

- ✓ Taller de Programación multiparadigma: Objetivos y Desafíos.
- ✓ Cómo implementarlo? La experiencia de la UNLP.
- ✓ Discusión de la metodología.
- ✓ Discusión de los contenidos en relación con el curso de Conceptos de Datos, Algoritmos y Programas.
- ✓ Discusión de los mecanismos de Evaluación.
- ✓ La capacitación de los docentes.
- ✓ La discusión de los Lenguajes a utilizar.
- ✓ Resultados luego de 3 años. Breves Conclusiones.

# Taller de Programación multiparadigma en un curso CS1.

## OBJETIVOS

- Realizar desarrollo de programas simples en el paradigma imperativo.
- Extender el manejo de datos a datos no lineales (Arboles).
- Introducción de los conceptos básicos de un segundo paradigma de programación (orientación a objetos) con énfasis en la noción de reusabilidad.
- Desarrollo de programas simples en un lenguaje orientado a objetos.
- Introducción de los conceptos básicos de la Programación Concurrente
- Desarrollo de programas simples con un lenguaje de programación concurrente que permita interpretar los conceptos de comunicación y sincronización entre procesos.
- Combinar los elementos estudiados previamente en Conceptos de Algoritmos, Datos y programas con las tareas experimentales en diferentes lenguajes de programación, a fin de que el alumno complete el ciclo del problema a su solución con computadora.

# Taller de Programación multiparadigma. La experiencia en la UNLP.

## METODOLOGIA

Módulo 1 | Módulo 2 | Módulo 3

Cada uno de los 3 módulos

- 5 turnos
- 8 clases
- 1 profesor y auxiliares
- 2 alumnos por grupo

El curso Taller de Programación se divide en 3 módulos: Programación Imperativa, Programación Orientada a Objetos y Programación Concurrente.

Cada módulo del Taller tiene una duración de 8 clases con una carga semanal de 2 clases de 3 hs. cada una. Cada clase consta de contenidos teórico-prácticos con actividades en máquina para resolver en el aula y también fuera del horario de clase. 60 Alumnos máximo por Aula.

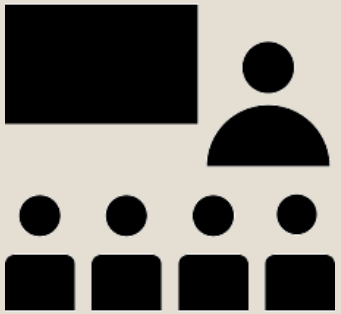
Cada turno (pueden ser 2 o más Aulas) está a cargo de un docente responsable del dictado de los contenidos teóricos-prácticos y cuenta con auxiliares docentes para las consultas de las actividades en máquina.

En cada turno, se conformarán equipos integrados por 2 alumnos para el trabajo en máquina. Cada equipo será responsable de una notebook que utilizará para el desarrollo de los ejercicios prácticos.

El material teórico y la ejercitación práctica utilizados en el curso están disponibles en el EVEA IDEAS.

# Taller de Programación multiparadigma. La experiencia en la UNLP.

## METODOLOGIA



### Asistencia (80% mínimo por módulo)

La asistencia a las clases teórico-prácticas es obligatoria.
En cada clase de Taller los alumnos tendrán presente, ausente, o ausente justificado. Los ausentes justificados no pasan a ser presentes.

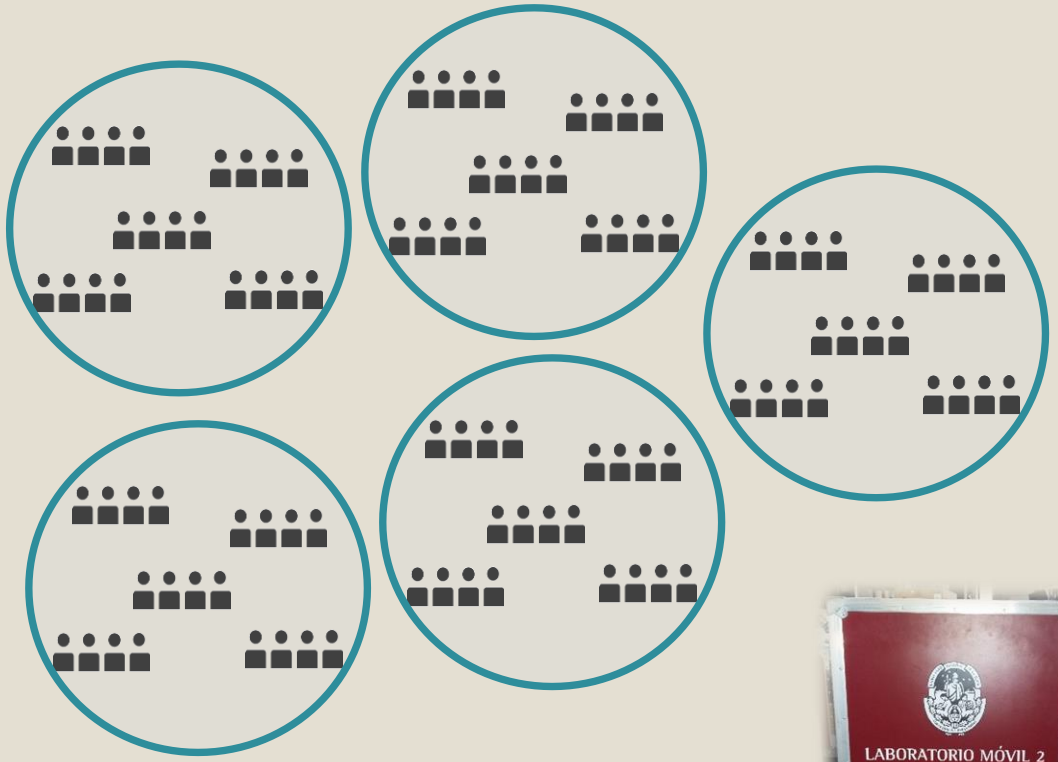
La asistencia a cada clase será tomada una única vez durante el horario de clase. Si un alumno no se encuentra en el aula por cualquier motivo, tendrá ausente.

### Evaluaciones Breves en la clase

Durante la clase se propone la realización de ejercicios prácticos que los alumnos deben resolver y enviar, por el entorno IDEAS, al auxiliar a su cargo.
La evaluación de estas actividades servirá de información para los docentes y de orientación para el alumno. El rendimiento satisfactorio de los alumnos en estas pruebas será considerado, a favor del alumno, durante la instancia de evaluación final del Taller.

# Taller de Programación multiparadigma. La experiencia en la UNLP.

## LABORATORIOS y AULAS REQUERIDAS



Con unos **300 alumnos** que cursan actualmente, se requieren al menos **5 Aulas** con **30 máquinas (60 alumnos)** cada una, con todas las capacidades incluyendo conexión a Internet.

Esto plantea exigencias importantes y dio lugar al desarrollo de los **"Laboratorios Móviles"** de la Facultad de Informática.


# Taller de Programación multiparadigma. La experiencia en la UNLP.

## METODOLOGIA EN EL AULA

Las 3 hs. de cada clase en el aula se estructuran en 3 fases:

Cómo es una clase?

- El docente responsable explica el tema a desarrollar, que puede tener componentes teóricos o directamente el problema experimental.
- A continuación los docentes (profesor y auxiliares) desarrollan en máquina 1 o más programas de demostración relacionados, que al mismo tiempo simplifican las cuestiones sintácticas para los alumnos.
- Los alumnos trabajan (y consultan) sobre los ejercicios planteados para ellos. Deben entregar parte de su tarea y otra parte la llevan para continuarla fuera del Aula.

En todo momento las máquinas de los alumnos están activas y si es necesario tienen los componentes relacionados con el tema y los ejercicios.

# Taller de Programación multiparadigma. La experiencia en la UNLP.

## METODOLOGIA DE EVALUACIÓN

Hay una **evaluación continua** en base a los trabajos experimentales que se definen en cada clase y que pueden requerir entregas.

Trabajo final por tema/paradigma que se puede realizar en equipo (2 alumnos) + evaluación individual del trabajo en máquina para aprobar cada paradigma. (con nota).

Si el alumno **aprueba** las evaluaciones individuales de los **3 paradigmas**, tiene la **promoción** con una nota promedio de sus 3 exámenes.

Si **aprueba 2 de los 3** tendrá la aprobación de trabajos prácticos, **cursada**.

Si **aprueba 1 de los 3** tendrá una instancia de **recuperación** similar a los exámenes rendidos donde será evaluado en 1 de los 2 paradigmas que desaprobó y debe aprobar para obtener la cursada.

# Taller de Programación multiparadigma. La experiencia en la UNLP.

## EL ALUMNO

Mayor **motivación** e interés.

Mayor **exigencia** en el aula y fuera del aula.

Las **evaluaciones son muy directas** y efectivas, aunque requieren un esfuerzo adicional del alumno y una gran capacidad del docente para corregirlas.

Es fundamental una gran **organización** para que el alumnos aproveche los tiempos en el aula y con los docentes, interactuando sobre los trabajos experimentales.

La opinión de los alumnos es muy favorable.

# Taller de Programación multiparadigma. La experiencia en la UNLP.

## TEMAS TEÓRICOS

**Programación Estructurada**

Estructuras de datos no lineales.
Tipo de dato Arboles. Definición y terminología asociada. Características. Operaciones.
Implementación de algoritmos fundamentales sobre estructuras de datos estáticas y dinámicas: búsquedas, ordenación, merge.
Desarrollo de programas en un lenguaje imperativo (Pascal).

1

**Programación Orientada a Objetos**

B. Programación orientada a objetos
Introducción a la POO.
Concepto de Objeto (estado y comportamiento). Clase e Instancia, Constructores. Concepto de Herencia.
Desarrollo de programas en un lenguaje orientado a objetos (Java).

2

**Programación Concurrente**

Conceptos básicos de concurrencia y paralelismo.
Procesos.
Comunicación y sincronización entre procesos.
Desarrollo de programas concurrentes y paralelos utilizando el ambiente del multirobot.
CMRE.

3

# Taller de Programación multiparadigma. La experiencia en la UNLP.

## Trabajo Experimental para el Alumno

El salto conceptual y formativo para el Alumno.

El requerimiento de operar efectivamente una máquina que tiene determinadas características, con un entorno determinado y un lenguaje con una sintaxis y una semántica específica (NO es lo mismo que lápiz y papel…)

El fortalecimiento de la prioridad del aprendizaje y el autoaprendizaje y la autoevaluación por sobre la enseñanza tradicional.

La importancia del **“saber hacer”** para el futuro del alumno en la carrera.

# Taller de Programación multiparadigma. La experiencia en la UNLP.

## CLASES: Cronograma y Evaluaciones (Imperativo-PASCAL)

| Clase | Contenidos | Actividades |
|---|---|---|
| Clase 1 | Presentación del Taller.<br>Manejo de las máquinas.<br>Operación de Merge en Listas y Vectores | Implementación de la operación de Merge de dos listas en Pascal.<br>Implementación de la operación de Merge de dos vectores. |
| Clase 2 | Ordenación en Arreglos. Método de ordenación por intercambio. | Implementación de la operación de Búsqueda y Ordenación por intercambio en Pascal. |
| Clase 3 | Recursión. Concepto. Características. Ejercitación. | Resolución de ejercicios básicos utilizando recursión. |
| Clase 4 | Arboles Binarios Ordenados. Concepto. Operaciones. Ejercitación. | Implementación de las operaciones básicas de árboles binarios ordenados en Pascal. |
| Clase 5 | Arboles Binarios Ordenados.<br>Ejercitación | Resolución de problemas utilizando el tipo de dato árbol binario ordenado. |
| Clase 6 | Resolución del Trabajo Final | Trabajo en grupo sobre el problema planteado p/c/ grupo. |
| Clase 7 | Coloquios y Evaluaciones individuales. | Trabajo individual en máquina. |
| Clase 8 | Coloquios y Evaluaciones individuales | Trabajo individual en máquina |

# Taller de Programación multiparadigma. La experiencia en la UNLP.

## CLASES: Cronograma y Evaluaciones (Objetos - JAVA)

| Clase | Contenidos | Actividades |
|---|---|---|
| Clase 1 | Conceptos básicos del lenguaje Java | Implementación de programas simples imperativos en Java para ejercitar la sintaxis. |
| Clase 2 | Introducción a la POO | Ejercitación que comprende instanciación de objetos y envío de mensajes. |
| Clase 3 | Conceptos básicos de POO utilizando Java. | Ejercitación que comprende programación de nuevas clases, instanciación de objetos de dichas clases, envío de mensajes a dichos objetos. |
| Clase 4 | Constructores. Interacción entre objetos. | Ejercitación que comprende la incorporación de constructores a las clases. Ejercitar interacción entre objetos. |
| Clase 5 | El concepto de herencia. | Ejercitación con herencia. |
| Clase 6 | Resolución del Trabajo Final | Trabajo en grupo sobre el problema planteado p/c/ grupo. |
| Clase 7 | Coloquios y Evaluaciones individuales. | Trabajo individual en máquina. |
| Clase 8 | Coloquios y Evaluaciones individuales | Trabajo individual en máquina |

# Taller de Programación multiparadigma. La experiencia en la UNLP.

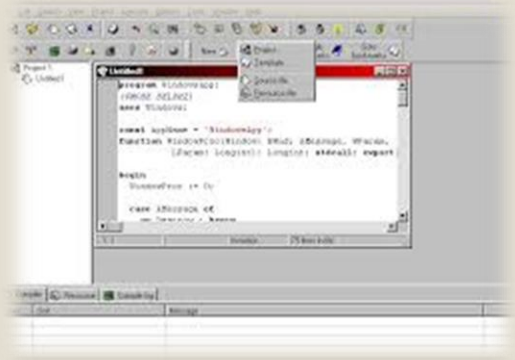
## CLASES: Cronograma y Evaluaciones (Concurrente - CMRE)

| Clase | Contenidos | Actividades |
|---|---|---|
| Clase 1 | Conceptos básicos de Concurrencia | Ejercicios que muestran los problemas de concurrencia. |
| Clase 2 | Entorno CMRE - R-info | Trabajo con el entorno CMRE- R-info. |
| Clase 3 | Memoria distribuida | Trabajo con el entorno CMRE- R-info aplicando los conceptos vistos a ejercicios con memoria distribuida. |
| Clase 4 | Memoria compartida | Trabajo con el entorno CMRE- R-info aplicando los conceptos vistos a ejercicios con memoria compartida. |
| Clase 5 | Memoria distribuida y compartida | Trabajo con el entorno CMRE- R-info aplicando los conceptos vistos a ejercicios con memoria distribuida y compartida. |
| Clase 6 | Resolución del Trabajo Final (Parte I) | Trabajo en grupo sobre el problema planteado p/c/ grupo. |
| Clase 7 | Resolución del Trabajo Final (Parte II) | Trabajo en máquina. |
| Clase 8 | Coloquios | Trabajo individual en máquina |

# Taller de Programación multiparadigma. La experiencia en la UNLP.

## Problema tipo en la Evaluación **PARADIGMA IMPERATIVO**

Se lee la información de los empleados de una consultora informática.
De cada empleado se ingresa: código de empleado, DNI, apellido y nombre, código del departamento donde trabaja (1..10), categoría de empleado (1..5) y sueldo básico.
La lectura finaliza cuando llega el empleado con DNI 0. Se debe generar la estructura ordenada por departamento y categoría.
Se pide:



a) Informar el código de departamento con mayor cantidad de empleados.

b) Informar para cada departamento y cada categoría, el monto total en sueldos que deberá abonar la consultora.

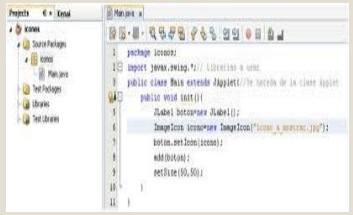
c) Informar el apellido y nombre de aquellos empleados cuyo DNI comience con 5.
Realice el programa principal que invoque a los módulos implementados

# Taller de Programación multiparadigma. La experiencia en la UNLP.

## Problema tipo en la Evaluación **POO**

Una compañía aseguradora quiere un sistema que le permita manejar seguros de automóviles y de inmobiliarios. De todo seguro otorgado se desea conocer nombre, apellido y DNI del asegurado. Número de póliza y valor del bien asegurado.

Las pólizas para automóviles se caracterizan por la marca y modelo del auto y su patente. Las pólizas para bienes inmobiliarios se caracterizan por la dirección donde se encuentra el bien, la cantidad de metros cuadrados y el número de habitación.



Realice un primer modelo de la jerarquía de clases propuesta, con los atributos de cada clase y métodos para obtener/modificar el valor de los mismos. Programe cada clase y genere constructores para los seguros de automóviles y los seguros de inmuebles que reciban la información necesaria para iniciar los atributos del objeto a crear.

Incorpore la funcionalidad para calcular la cuota mensual que deberá abonar el asegurado método (***calcularCuota***), teniendo en cuenta lo siguiente:

Para el seguro de automóviles el monto de la cuota se calcula multiplicando dos valores: el 0.75% del valor del bien asegurado y un índice calculado como $(2015 - \text{modelo} + 1)^{0.12}$

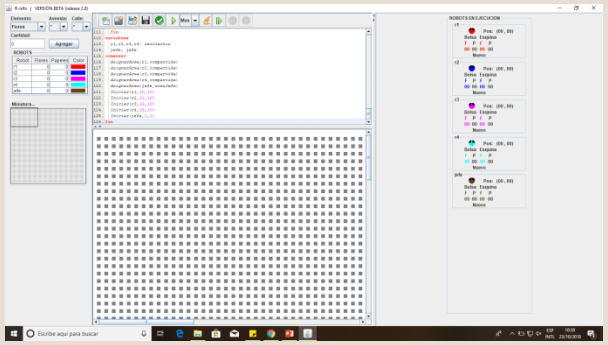
Para el seguro de inmobiliarios el monto de la cuota se calcula multiplicando dos valores: el 0.75% del valor del bien asegurado y un índice calculado como $(\text{superficie } m^2)^{0.05}$

Genere un programa principal que instancie una póliza de automóviles y una póliza de bicicletas con datos leídos desde teclado. El programa deberá imprimir el valor de la cuota mensual de cada póliza generada.

# Taller de Programación multiparadigma. La experiencia en la UNLP.

## Problema tipo en la Evaluación Concurrencia

Implemente el siguiente juego:
Existe un área compartida delimitada por las esquinas (1,1) (20,20), tres robots jugadores y robot fiscalizador.



Cada robot jugador debe tratar de juntar la mayor cantidad de papeles posible, para esto tiene tres intentos. En cada intento se posiciona en una esquina determinada al azar (dentro del área) y junta todos los papeles de esa esquina y vuelve a su esquina original.

Luego robot fiscalizador determinará cual es el robot que más papeles juntó en sus tres intentos.

**Notas:** Los robots se posicionan inicialmente en (2,2), (8,7), (15,15) y el robot fiscalizador en la esquina (1,1).

# Taller de Programación multiparadigma. La experiencia en la UNLP.

## DISCUSIÓN DE LOS MECANISMOS DE EVALUACIÓN

Importancia de la asistencia y la participación en el Aula.

Importancia del seguimiento y la corrección de las tareas experimentales en el Aula y realizadas fuera del Aula.

Trabajo en equipo y trabajo individual.

Evaluaciones individuales.

Posibilidad de promoción del examen final.

Dificultades de las evaluaciones finales para alumnos que sólo obtienen la cursada.

El régimen es más “duro” para el Alumno.

# Taller de Programación multiparadigma. La experiencia en la UNLP.

## CAPACITACION DE LOS DOCENTES: **Contenidos**

El complejo proceso de adaptar una cátedra de 800 - 1000 alumnos a un cambio importante en los contenidos y conocimientos requeridos de todos los niveles docentes.

La necesidad de profesores con dominio de los diferentes paradigmas y lenguajes.

La elaboración de trabajos experimentales que generen como resultado las competencias buscadas en el alumno resulta difícil para docentes con una estructura de práctica “en lápiz y papel¨.

La generación de material didáctico de base (proceso previo a la puesta en marcha del Taller).

La coordinación con otros docentes (Ingreso y 2$^{do}$. Año) en cuanto a contenidos.

# Taller de Programación multiparadigma. La experiencia en la UNLP.

## CAPACITACION DE LOS DOCENTES: **Metodología**

Toda cátedra ¨teórico-práctica¨ tiene sus complicaciones, porque es una metodología que diferencia poco los roles de Profesores, JTPs y Auxiliares.

Además del trabajo en el Aula, el docente debe dominar aspectos de la tecnología de los laboratorios móviles, de los entornos experimentales e incluso poder manejar un cierto nivel de fallas. Esto obligó a combinar un soporte técnico para la cátedra.

La metodología exige horas en el Aula y también mucha concentración en las correcciones. Todo método de ¨evaluación continua¨ con una fuerte restricción de tiempo (8 clases en 4 semanas por paradigma) resulta en requerimientos fuertes para los docentes.

Las evaluaciones individuales y los exámenes finales tienen también una metodología novedosa para la Facultad.

# Taller de Programación multiparadigma. La experiencia en la UNLP.

Por la relación con Conceptos de Algoritmos y Estructuras de Datos.

Por la simplicidad y la facilidad de expresar correctamente el flujo de control.

**Imperativo Pascal**

Por los conocimientos previos de los alumnos y docentes.

Por qué no C ? Python? Otros?

# Taller de Programación multiparadigma. La experiencia en la UNLP.

Por la relación con Algoritmos y Estructuras de Datos y otras asignaturas posteriores en la carrera.

Por su generalidad y posible empleo como lenguaje concurrente.

**Objetos JAVA**

Entorno JAVA elegido.

Por qué no C++ ? Smalltalk ? Otros?

# Taller de Programación multiparadigma. La experiencia en la UNLP.

Por la visualización de los aspectos complejos de la concurrencia, de un modo simple.

Por la relación con el Ingreso y los conocimientos previos de los alumnos y docentes.

**Concurrente CMRE**

Por la claridad conceptual en cuanto a los dos conceptos básicos: comunicación y sincronización.

Por los Lenguajes/bibliotecas utilizadas en cursos de años superiores.

# Taller de Programación multiparadigma. La experiencia en la UNLP.

## SINTESIS: LOS LENGUAJES POSIBLES DE UTILIZAR

La elección habilita una discusión dinámica.

Debemos fortalecer el análisis de las competencias buscadas en el alumno, simplificando el lenguaje de modo que el alumno se centre en la resolución efectiva de los problemas que se le plantean.

Resulta claro que para la filosofía de la Facultad de abrir el Seminario de Lenguajes de 2$^{do}$. Año a múltiples alternativas en diferentes paradigmas (e incluso cambiarlas anualmente) la elección de un UNICO lenguaje sería poco útil.

La opinión (y experiencia) de los docentes involucrados es muy importante para consolidar una asignatura fuertemente experimental que debe “completar” los aspectos formativos del curso de Conceptos de Algoritmos, Datos y Programas.

# Taller de Programación multiparadigma. La experiencia en la UNLP.

## RESULTADOS LUEGO DE TRES AÑOS: **Aprendizaje**

Los alumnos que aprueban el Taller han demostrado un aprendizaje significativo, con impacto en el desarrollo de sus carreras en 2$^{do}$. y 3$^{er}$. Año.

Si analizamos las evaluaciones del curso anual clásico de ADP y las comparamos con las evaluaciones actuales en CADP + Taller de Programación, resulta evidente que el alumno ha dado un “salto” de calidad en su aprendizaje.

La motivación generada por el acceso a los Laboratorios Móviles y realizar una práctica significativa en máquina en primer año representa un aspecto muy positivo para los alumnos interesados en ser profesionales de Informática.

# Taller de Programación multiparadigma. La experiencia en la UNLP.

## RESULTADOS LUEGO DE TRES AÑOS: **Cuantitativos**

Si bien el resultado de trabajos prácticos aprobados y exámenes finales aprobados es muy bueno en Taller de Programación, es necesario mirar el panorama global comparando los resultados del curso CS1 anual (Algoritmos, Datos y Programas) contra los dos cursos cuatrimestrales.

El resultado global es bastante similar: entre 230 y 280 alumnos obtienen los trabajos prácticos de ambas asignaturas, lo que es muy parecido a lo que se obtenía en ADP.

La mayor (y mejor) diferencia es que aumenta el número de finales aprobados o promovidos al cabo del año, porque los alumnos para obtener la promoción del Taller de Programación tienen que tener aprobado el final de CADP.

# Taller de Programación multiparadigma. La experiencia en la UNLP.

## REDICTADOS

La Facultad ha completado el ciclo de estos dos cursos CADP + TP con un redictado contra semestre de cada uno de ellos.

Estos redictados ayudan a limitar la deserción y si bien hay alumnos que completan primer año sólo con CADP, de hecho están en mejores condiciones que haber abandonado ADP… incluso casi terminando el ciclo lectivo y con un primer parcial aprobado (que ahora cubriría las exigencias de la cursada de CADP).

Por otra parte, a partir de 2018 hay un curso de verano de CADP que permite acceder al año siguiente al redictado de Taller de Programación en el primer semestre.

NOTA Los “redictados” tienen condiciones académicas y no todos los alumnos que no aprueban los trabajos prácticos pueden acceder a un redictado en el cuatrimestre siguiente. Esto acota los números y evita que se pueda cursar en cualquier cuatrimestre.

# Taller de Programación multiparadigma. La experiencia en la UNLP.

## RESULTADOS INDIRECTOS

El impacto del Taller de Programación y el interés de los alumnos se ha proyectado sobre la Ingeniería en Computación que está transformando sus dos primeras asignaturas (Programación1 y Programación 2) de modo de tener una modalidad “Taller” en Programación 2.

La tecnología de los Laboratorios Móviles se ha generalizado para múltiples asignaturas y actualmente la Facultad tiene 8 Laboratorios Móviles con unas 200 portables de características similares y que se pueden «mover» prácticamente por todas las Aulas de grado y Posgrado.

Indirectamente el Taller de Programación obligó a una mejora sensible en las instalaciones eléctricas de todas las Aulas de la Facultad.

# Taller de Programación multiparadigma. La experiencia en la UNLP.

## BREVES CONCLUSIONES

La decisión de implementar un curso experimental de Taller de Programación Multiparadigma ha sido positivo por:

- El impacto en el aprendizaje y la formación de los alumnos.
- El efecto de actualización sobre un gran número de docentes.
- La mejora de la articulación con el Ingreso y con 2do. Año.
- La necesaria mejora de la infraestructura que generó en la Facultad.

Hay que considerar las dificultades cuando se manejan números altos de alumnos y hay que reflexionar sobre las ventajas y desventajas de la cuatrimestralización.

Es deseable que el alumno llegue con una mejor formación informática (al menos una cierta capacidad de expresión de algoritmos simbólicos y alguna práctica concreta en máquina).

# Taller de Programación multiparadigma en un curso CS1. Resultados y Desafíos.

## REFLEXIÓN FINAL

**Tránsito complejo entre perfiles**

Creemos que desarrollar un Taller de Programación multiparadigma es positivo para este tránsito.

# Taller de Programación multiparadigma en un curso CS1. Resultados y Desafíos.


## Agradecimientos