

e-Book

Actas

Actas II Jornadas Argentinas de Didáctica de la Programación

Editores:

Araceli Acosta
Belén Bonello
Cecilia Martínez
Sonia Permigiani
Nicolás Wolovick



UNC

FAMAF

ffyh
Facultad de Filosofía
y Humanidades - UNC



ACTAS

II JORNADAS ARGENTINAS DE DIDÁCTICA DE LA PROGRAMACIÓN



Universidad
Nacional
de Córdoba



Facultad
de Matemática,
Astronomía, Física
y Computación



Actas II Jornadas Argentinas de Didáctica de la Programación / Alejandro Iglesias... [et al.] ; editado por Araceli Acosta... [et al.].- 1a ed.- Córdoba : Universidad Nacional de Córdoba. Facultad de Filosofía y Humanidades, 2020.
Libro digital, PDF

Archivo Digital: descarga y online
ISBN 978-950-33-1600-9

1. Didáctica. 2. Lenguaje de Programación. 3. Formación Docente. I. Iglesias, Alejandro. II. Acosta, Araceli, ed.
CDD 004.071

COMITÉ ACADÉMICO

Araceli Acosta
Marcelo Arroyo
Francisco Bavera
Luciana Benotti
María Belén Bonello
Virginia Brassesco
Claudia Casariego
Marcela Daniele
Gladys Dapozo
Gustavo Del Dago
Maria Emilia Echeveste
Marcos Gomez
Carolina Gonzalez
Guillermo Grosso
Renata Guatelli
Marta Lasso
Maria Carmen Leonardi
Diego Letzen

Matías López Rosenfeld
Cecilia Martinez
Pablo E. Martínez López
Analía Mendez
Natalia Monjelat
Sonia Permigiani
María Valeria Poliche
Claudia Queiruga
Jorge Rodríguez
Alvaro Ruiz-de-Mendarozqueta
Claudia Cecilia Russo
Alfredo Héctor Sanzo
Fernando Schapachnik
Herman Schinca
Pablo Turjanski
Nicolás Wolovick
Dante Zanarini
Rafael Zurita

EDITORES

Araceli Acosta
Belén Bonello
Cecilia Martínez
Sonia Permigiani
Nicolás Wolovick

ILUSTRACIÓN DE TAPA

Manuel Coll – Área de Comunicación Institucional – FFyH – UNC



Esta obra está bajo una [Licencia Creative Commons Atribución-NoComercial-CompartirIgual 4.0 Internacional](https://creativecommons.org/licenses/by-nc-sa/4.0/).

Hacia una didáctica de la programación para la secundaria argentina

Pablo E. “Fidel” Martínez López¹, Alfredo Sanzo², Fernando Schapachnik³

Resumen: Dentro del marco de la iniciativa Program.AR, la Fundación Sadosky desarrolló un curso para la educación secundaria denominado “*La programación y su didáctica*” (LPYSDI), el cual se viene dictando con éxito desde 2015 a docentes de todo el país a través de un modelo de convenios con Universidades Nacionales. En este artículo presentamos el diseño de la segunda parte de este curso, denominado “*La programación y su didáctica*” (LPYSD2) concentrándonos en los objetivos perseguidos, la metodología utilizada, el marco conceptual considerado y las características del curso resultante. Este curso también se comenzó a dictar este año a través del modelo de convenios mencionado y así se integra a los esfuerzos dentro de la estrategia de la iniciativa Program.AR en pos de conseguir que la materia Ciencias de la Computación sea dictada en todos los niveles educativos del país como cultura general.

1 Universidad Nacional de Quilmes Fundación Dr. Manuel Sadosky. fidel@unq.edu.ar

2 Fundación Dr. Manuel Sadosky, Universidad de Buenos Aires, FCEyN. asanzo@fundacionsadosky.org.ar

3 Partially supported by UBACyT 20020130200032BA. Fundación Dr. Manuel Sadosky, Universidad de Buenos Aires, FCEyN. fschapachnik@fundacionsadosky.org.ar

1. Introducción

En el marco de la iniciativa Program.AR⁴, la Fundación Sadosky diseñó y dictó un curso que llamaremos LPYSD1 (*"La programación y su didáctica, parte 1"*) [FSO15]. Este curso fue concebido para su dictado a estudiantes de secundaria, y se realizaron diversas capacitaciones a docentes de secundaria para que puedan trasladarlo al aula. Desde 2014 la Fundación Sadosky capacita equipos universitarios de todo el país para que dicten dichos cursos de formación docente en programación con este material. Estos equipos son asignados mediante un proceso de selección determinado por un jurado internacional, y ya han formado a más de 1500 docentes. LPYSD1 es una primera introducción a la programación, y a una forma de enseñarla, y tiene varias características innovadoras. Sin embargo, el curso deja varias cuestiones importantes sin atender.

En este trabajo explicaremos el diseño y prueba de la continuación de LPYSD1, que llamaremos LPYSD2 (*"La programación y su didáctica, parte 2"*).

2. Planificación de la continuación

2.1. Objetivos a cumplir

El primer paso cuando nos planteamos diseñar e implementar LPYSD2 fue establecer cuáles eran los objetivos que buscábamos con tal curso. Nos planteamos como objetivos que el curso incluyese:

- Procesar/computar datos.
- Abordar problemas tecnológicos/informáticos relacionados a la experiencia cotidiana.
- Proveer al docente confianza sobre los temas que imparte.
- Incluir problemas no triviales para que los cursantes resuelvan.
- Codificar, poner en práctica el conocimiento, concretizar.

Estos temas fueron elegidos por las razones que se exponen a continuación. Este curso se basa en la experiencia de los autores sobre los temas a incluir en un curso de programación que sea continuación de un curso inicial (ver por ejemplo [SP88, AHU83]).

Procesar datos, porque en LPYSD1 no se incluyó la idea de un programa como procesador de información, lo cual creemos es central en la concepción de la programación. En esa primera parte la forma de concebir a los programas era como meras series de instrucciones para que un autómatas haga algún cambio en el escenario. En esta segunda parte (LPYSD2; se incluye la concepción de que un programa puede también recibir una entrada y producir una salida.

4 <http://program.ar>

El segundo ítem tiene que ver con el objetivo último de la iniciativa Program.AR sobre explicar las características de las soluciones tecnológicas que nos rodean hoy en día. El primer curso, al ser introductorio, trabajó sobre ejemplos extremadamente simples, y no estaba presente ese puente entre el qué y el cómo en las soluciones tecnológicas logradas con programación. Se busca con esta segunda parte del curso explicitar que con cierto algoritmo o con cierta estructura podría resolverse un problema con el que los cursantes están en contacto cotidianamente. También que puedan imaginarse cómo funcionan los programas con los que interactúan a diario, al menos en un nivel básico. Por ejemplo, formular hipótesis sobre cómo funciona un predictor de texto de un celular, o cómo encuentra Google lo buscado, etc. La inclusión de lecturas complementarias es importante también para conseguir este objetivo.

El tercer ítem está relacionado al hecho de que en el primer curso todo el contenido estaba destinado a que el docente lo usase en los cursos para sus alumnos, y hasta ahí llegaba su saber. En este curso buscamos entonces que los cursantes puedan conocer más que lo que deben dictar en un curso, y de esta forma manejarse mejor en el aula. Es decir, proveerles una serie conocimientos disciplinares adicionales al curso anterior con el objetivo de generar confianza. Gracias a esto, el cursante puede entender mejor el rol que tuvo y tiene LPYSD1 en la enseñanza básica de la programación, sin concebirlo como una sobresimplificación, y entendiendo mejor algunas de las características de aquel. El cuarto ítem, que los ejercicios sean no triviales, se puso como objetivo para construir actividades desafiantes y enriquecedoras, que permitan por un lado lograr la satisfacción de haber resuelto un problema interesante y por el otro mostrar la complejidad inherente de algunos de los problemas computacionales más comunes.

El último ítem viene de la convicción de que en didáctica, la apropiación del conocimiento es mejor cuando el cursante se ejercita y produce resultados concretos. En la temática de la programación esto significa escribir código y hacerlo

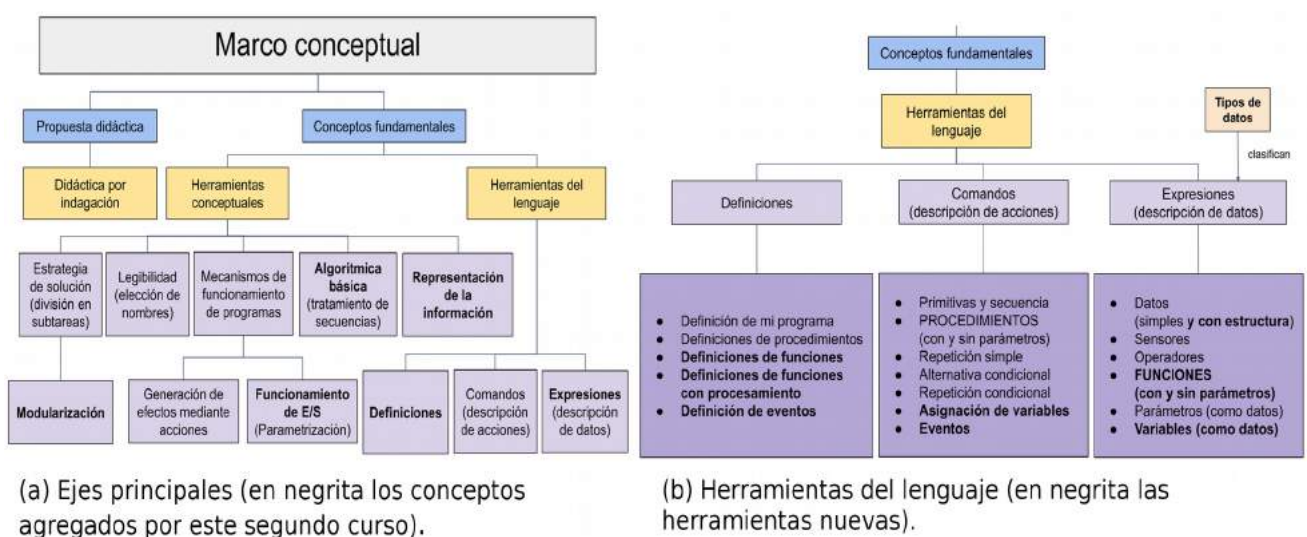


Figura 1. Marco conceptual de LPYSD2

funcionar. Además, si producen alguna aplicación, por sencilla que sea, el cursante sentirá que se llevó algo valioso.

Un objetivo implícito por provenir de LPYSD1, pero que vale la pena mencionar aquí, es reiterar la características del curso 1 de trabajar con entornos y actividades diversas, con características diferentes aunque una base común.

Mientras que los cursos tradicionales son *monolenguaje*, i.e. trabajan con un único entorno/lenguaje de programación (por ejemplo, cursos basados en Scratch [ML07, LJ11], Alice [Her07], Karel the Robot [Pat81], etc.), creemos que un curso que se focalice en conceptos fundamentales debe ofrecer actividades de más de una naturaleza (e.g. con computadoras y sin computadoras) y en más de un entorno/lenguaje, como una manera de mostrar y practicar el hecho de que los conceptos no son exclusivos de un tipo de actividad, de un lenguaje específico o de un entorno de trabajo determinado.

2.2. Extensión del marco conceptual

El marco conceptual de LPYSD1 se basó en 3 ejes: didáctica, herramientas conceptuales y herramientas del lenguaje. En este curso se plantea continuar con esos 3 ejes, y extenderlos acorde a los objetivos planteados.

La decisión de mantener el eje de la didáctica, que es la didáctica por indagación [Dos15], implica articular el curso alrededor de actividades que plantean un problema, invitan a resolverlo, a experimentarlo, y luego dan la teoría necesaria, en lugar de hacer la presentación teórica primero y la práctica luego. Una característica fundamental de esta propuesta didáctica consiste en que los cursantes experimenten esta forma de aprendizaje durante la capacitación, y luego reflexionen sobre la misma, con el objetivo de poder producir actividades propias que sigan estos principios de trabajo. De esta forma, ambos cursos se basan no solamente en que los estudiantes, destinatarios finales, aprendan mediante indagación, sino que los docentes, al capacitarse, también la experimenten, y luego la analicen y comprendan en profundidad.

En el eje de las herramientas conceptuales, los conceptos principales de LPYSD1, siendo tan fundamentales, también deben estar presentes: la determinación y explicitación de la estrategia al ofrecer la solución de un problema, y el hecho de que el programa debe ser legible por las personas. A ello se agregan los siguientes conceptos (ver la Fig. 1a):

- las dos formas de concebir los programas: la ya trabajada de producir ciertos efectos sobre el estado a través de acciones, y la nueva de transformar información a través de su procesamiento (a la que llamaremos “funcionamiento de Entrada/Salida”),

- la idea de que la información se representa de diversas formas dentro de un sistema de cómputo (representación de la información),
- la noción de algoritmo, su práctica a través de un algoritmo de recorrido de secuencias [SP88] y la sugerencia de que existen diversidad de algoritmos (en particular, algoritmos de ordenamiento y de búsqueda binaria, siempre acorde a los objetivos), y
- la modularización como forma de transformar porciones de código en un producto reutilizable para resolver nuevos problemas.

Finalmente, en el eje de las herramientas del lenguaje, que en el curso LPYSD1 estaban fuertemente concentradas en los comandos, el foco se trasladará al universo de las expresiones, dado que las mismas son la forma en que se describe la información. También se distinguirá la noción de definición de operaciones (procedimientos, funciones, eventos, etc.), de la noción de utilización de los mismos (ver la Fig. 1b), que en el curso LPYSD1 no se distinguían de forma explícita. Dentro del universo de las expresiones se propone trabajar con la idea de función (similar a la idea de procedimiento en el universo de los comandos), la noción de datos con estructura (como iniciación a las estructuras de datos, pero de forma simple), y profundizar en las nociones de operadores, de parámetros y variables, para lo cual se hará necesario incluir la idea de tipos de datos.

Se espera que a través de la comprensión de estos conceptos y herramientas se puedan explicar varias características de las aplicaciones modernas. La elección de estos conceptos implicó el desplazamiento de otros posibles. Por ejemplo, quedaron afuera la manipulación y definición de estructuras de datos más complejas, o nociones de aplicaciones prácticas como bases de datos, procesamiento distribuido o inteligencia artificial. Si bien los contemplamos, no encontramos la forma de incluirlos dentro del tiempo que le podíamos dedicar a cada tema sin perder precisión o riqueza.

Las herramientas conceptuales y las herramientas del lenguaje utilizadas tanto en LPYSD1 como LPYSD2 están basados en trabajos anteriores [MLBSO12, ML13] sobre las cuales el equipo de Program.AR en su manual [FSO15] hizo una selección y agregó el trabajo con bloques, con actividades unplugged, con desafíos concretos y la didáctica por indagación. Con el pasar del tiempo y las consecuentes revisiones se consolidó un material [MLCAP17] sobre el cual se continuó construyendo, agregando temas como representación de información, de tratamiento de datos con estructura (fundamentalmente secuencias) y de la exploración de la riqueza del mundo de las expresiones. Estos temas, con algunos desarrollos previos aún no documentados, fueron enriquecidos y desarrollados para abarcar los objetivos aquí propuestos.

3. El curso “La programación y su didáctica 2”

3.1 Características de LPYSD2

A la hora de diseñar un curso de esta naturaleza, es importante tomar algunas decisiones desde el comienzo, que luego afectarán las características del curso en sí. La primera de las decisiones ya había sido tomada, y es que este curso es una continuación de LPYSD1, por lo que los contenidos dados en ese curso son prerequisite para tomar este, y el formato de trabajo y la cantidad de horas debe ser similar al de LPYSD1: 70 hs. presenciales y 30 hs. no presenciales. Luego de eso, la más importante de estas decisiones tiene que ver con el o los entornos de programación que se utilizarán para desarrollar las actividades, ya que los mismos estarán presentes en el día a día del curso, y afectarán el tipo de producciones que se pueden lograr. También es necesario elegir estrategias didácticas para que los conceptos más complejos, como por ejemplo la idea de datos con estructura o la noción de búsqueda binaria, aparezcan de forma sencilla e intuitiva. Con esto en mente, y teniendo en cuenta los objetivos, se deben seleccionar y secuenciar los temas, y finalmente, establecer los mecanismos de evaluación.

3.1.1. Entornos de programación

Dado que el curso LPYSD1 se basó completamente en programación en entornos con bloques, y que la programación basada en texto, si bien más adecuada profesionalmente, plantea desafíos didácticos mayores (por la complejidad de manejar la sintaxis de un lenguaje, y por la mayor facilidad para cometer errores al usar texto), decidimos desde el comienzo que los entornos de programación que propusiésemos tenían que estar basados en bloques. Luego de considerar varios entornos, finalmente elegimos dos: GobstonesWeb⁵ y App Inventor⁶.

El primero de ellos, diseñado por un equipo de la UNQ, permite definir actividades completas con facilidad, focalizando al estudiante en los aspectos conceptuales elegidos y eliminando otros innecesarios, incluye el tratamiento de datos con estructura, expresiones y todos los conceptos elegidos, y tiene como bonificación adicional el hecho de que, al tener acceso al código fuente del entorno, podemos realizar ajustes en la interfaz y la presentación.

El segundo, diseñado por el MIT, permite la definición de programas que funcionen en celulares, logrando de esta manera productos de software que pueden trascender su uso áulico; si bien es más difícil focalizar los conceptos, al usarlo en combinación con el primero se puede simplificar el acceso a su complejidad.

Ambos se pueden utilizar en forma *online* a través de Internet, o instalarlos para su uso local (aunque en el caso de App Inventor esto requiere un trabajo algo complejo). De esta forma, si bien el curso resulta más fácil de dictar teniendo conexión a Internet, no depende exclusivamente de ello.

5 <http://gobstones.github.io/gobstones-jr>

6 <http://ai2.appinventor.mit.edu/>

3.1.2. Estrategias didácticas

Entre las estrategias didácticas, como primer punto decidimos reutilizar la combinación de actividades que usan computadoras con otras que no las requieren, como forma de abordar los conceptos antes de pasar a las máquinas. Pero se hizo necesario buscar formas para transmitir por indagación las ideas complejas, tales como la noción de dato con estructura, la idea de que las computadoras manejan datos y pueden procesarlos, algoritmos complejos como el de búsqueda binaria, etc.

Para la noción de datos con estructura elegimos el dominio de las cartas españolas. Una carta es un dato con estructura que posee dos partes bien diferenciadas, y sin embargo es una sola entidad.

Un mazo de cartas es una instancia del tipo lista, que al ser presentado de la forma adecuada, permite poner en claro algunas de las características de las listas en los lenguajes de programación. En forma intencional omitimos la capacidad de definir nuevos tipos de datos con estructura, limitándonos solamente a las cartas y las listas (de cartas, de números y de letras), ya que el tema de diseño de otras estructuras es mucho más complejo conceptualmente, y no era parte de los objetivos planteados.

Para transmitir la idea de que las computadoras procesan datos, extendimos la metáfora de varias actividades sin computadoras del curso LPYSD1, donde el docente funcionaba como una computadora. En este curso, el docente-autómata no solo entiende comandos, sino que además puede responder ciertas preguntas. De esta forma, resulta más sencillo mostrar el funcionamiento de E/S de los programas, y la necesidad de contar con expresiones más ricas en el lenguaje de programación.

Finalmente, para transmitir la idea de algoritmos complejos, como los de ordenamiento o de búsqueda binaria, elegimos, para los primeros, presentar solo el ordenamiento por selección en forma modularizada y dejar el estudio de otras formas para lecturas adicionales, y para el segundo, utilizar una actividad sin computadoras diseñada por el equipo de la UNER para la especialización en docencia de computación resultado de un convenio con la Fundación Sadosky. Esta actividad, en la forma de un juego por parejas, no se llega a codificar, ya que el código necesario requiere características más avanzadas como manejo de índices o recursión, ambos fuera del alcance propuesto para este curso, pero sin embargo, consigue que los cursantes descubran por indagación la estrategia de buscar en forma binaria.

Como muestra del tipo de actividades que se proponen, elegimos las involucradas en la unidad de introducción a listas, por ser representativas del estilo propuesto: una combinación de actividades sin computadoras y con computadoras que se interrelacionan para favorecer la indagación guiada.

- Comenzamos con una actividad sin computadoras que llamamos *“Delfina y su papá van al almacén”*. En ella contamos una breve historia de una visita al almacén donde el almacenero anota precios en una listita de papel, comenzando por un sachet de leche que es lo primero que compran, y luego, planteamos que el docente funciona como un autómatas que responde preguntas acerca de dicha listita. Los cursantes deben responder algunas preguntas vinculadas con la historia (tales como “¿Cuántos productos compraron en total?” o “¿Cuál fue el precio de la leche?”), para lo cual deben descubrir las operaciones de lista (primer elemento, cantidad de elementos, etc.). Las preguntas se organizan en 2 niveles, el primero con respuestas directas desde las respuestas del autómatas, el segundo con la necesidad de realizar algún tipo de cuenta o procesamiento de dichas respuestas.
- Sigue una actividad con computadoras llamada *“Descubrir la lista”*⁷, donde practican el uso de las operaciones recién descubiertas y observan cómo se concretan en el entorno de programación.
- Luego siguen otras dos actividades sin computadoras, *“Los mazos también son listas”* y *“Los segundos serán los primeros”*, diseñadas para entender que los mazos de cartas son listas, porqué las listas se acceden de a un elemento por vez, y comprender que puede haber operaciones sobre listas que describan otras listas (en particular, la operación `sinElPrimero`, que describe la lista resultante de quitar el primer elemento de una lista, necesaria para realizar luego procesamientos lineales sobre las listas). En particular, en la primera de ambas, el docente vuelve a funcionar como un autómatas que responde preguntas, en este caso sobre un mazo de cartas (con algunas pocas cartas, por ejemplo entre 7 y 10), pero no les muestra el mazo y solo puede responder preguntas sobre la primera y la última de las cartas; de esta forma se entiende que algunas preguntas requieran más trabajo que otras y que las computadoras no ven los elementos individuales de una lista, sino el grupo completo.
- Se completa esta unidad con 3 actividades con computadoras: *“La lista contraataca”*⁸, *“El regreso de la lista”*⁹ y *“El repartir de las cartas”*¹⁰. en las que se trabaja fundamentalmente con la operación `sinElPrimero`, y se presenta el primer recorrido lineal de listas.

3.1.3. Temas del curso

Los temas elegidos para cumplir los objetivos se presentan en la Fig. 2. Para cada uno de ellos se diseñaron varias actividades que permiten presentar las ideas más importantes de cada tema a través de la didáctica por indagación. El material didáctico

7 <http://bit.ly/2Hlu8vT>

8 <http://bit.ly/2We8w3E>

9 <http://bit.ly/2Wj58Eu>

10 <http://bit.ly/2JsREAd>

resultante incluye el código necesario o las fichas para llevar adelante cada una de las actividades.

- | | |
|---|--|
| 1. Repaso de los conceptos de LPYSD1 | 9. Variables |
| 2. Sensores y operadores numéricos y booleanos | 10. Uso de variables como acumuladores |
| 3. Datos con estructura: cartas | 11. Datos con estructura: listas |
| 4. Funciones simples (sin parámetros) | 12. Operaciones sencillas sobre listas (totalizaciones y transformaciones) |
| 5. Tipos de datos | 13. Operaciones más complejas sobre listas (filtros y búsquedas) |
| 6. Eventos | 14. Representación de la información |
| 7. Parámetros y parametrización de operaciones (procedimientos y funciones) | 15. Operaciones avanzadas sobre listas (ordenamiento) |
| 8. Funcionamiento de E/S | 16. Algoritmo de búsqueda binaria |
| | 17. Aplicación de integración |

Figura 2. Temas cubiertos en LPYSD2, por orden de aparición.

3.1.4. Evaluación

En un curso de estas características, donde lo que se busca es transmitir conceptos didácticos además de los conceptos propios de la disciplina de la programación, es necesario que la evaluación tenga en cuenta las mismas. En el curso LPYSD1 esto se consiguió requiriendo como evaluación que los cursantes dictasen al menos una clase usando los conceptos y didáctica aprendidos, y que la misma fuese supervisada por uno de los capacitadores. Sin embargo, en este curso eso no resulta factible, pues este curso presupone que los cursantes han tomado el curso anterior, y eso, al día de hoy, no es posible lograrlo con sencillez con estudiantes de secundaria. Por eso, decidimos que el curso iba a tener dos instancias de evaluación, cubriendo los aspectos esenciales: el didáctico, el conceptual y la aplicación de conceptos.

La primera de las instancias de evaluación elegidas consiste en diseñar en grupos una actividad didáctica para presentar alguno de los temas, similar a las existentes, y exponerla durante las clases del curso, como si de una actividad más del curso se tratase. Para ello se les provee a los grupos con una ficha que los guía en el diseño de actividades, focalizándolos en determinar los objetivos didácticos, los temas a transmitir y la complejidad de la solución obtenida. Durante la presentación de estas actividades, los cursantes evaluados toman el rol de docentes, y son sus pares los que les plantean observaciones sobre la pertinencia y oportunidad de las decisiones tomadas; de esta forma, se produce un análisis interesante sobre los aspectos didácticos, y un debate muy rico sobre cómo conseguir los objetivos didácticos propuestos inicialmente. Esta actividad se propone al promediar la primera mitad del curso, se diseña durante el curso, y se evalúa en dos de las clases dedicadas especialmente a ello al promediar la segunda mitad.

La segunda de las instancias es un trabajo de investigación, del que luego deben presentar un breve informe. En este trabajo, dividido en dos partes, la propuesta es elegir una aplicación cotidiana o parte de ella, proponer una hipótesis acerca de su funcionamiento que incluya los contenidos del curso (primera parte de la actividad), y

luego investigar acerca de la misma para corroborar, refinar o refutar las hipótesis planteadas, sacar conclusiones y confeccionar un informe final que contenga todo el proceso (segunda parte de la actividad). Ambas partes se presentan en dos clases diferentes cerca del final, con el objetivo de que las hipótesis sucedan antes que la investigación.

3.1.5. Material didáctico producido

Además del código necesario para llevar a cabo las actividades con computadoras¹¹, desarrollamos fichas para cada una de las actividades sin computadoras, diapositivas para cada una de las clases, y estamos trabajando actualmente en un manual que le permita a un docente seleccionar algunas o todas las actividades para planificar sus propias clases.

4. Resultados preliminares

Como parte del diseño del curso, equipos de dos Universidades Nacionales (la Universidad Nacional de Quilmes (UNQ) y la Universidad Nacional del NorEste (UNNE) realizaron dictados piloto durante el 2do semestre de 2018. A partir de estas experiencias, se ajustaron varias de las características del curso, diversas actividades y se refinó el material didáctico producido.

Luego desde la Fundación Sadosky se capacitó a otras 8 Universidades Nacionales (las Universidades de Buenos Aires, de Catamarca, de San Luis, del NorOeste de la provincia de Buenos Aires, de Tucumán, de Santiago del Estero, del Comahue y la Tecnológica regional Santa Fe – UBA, UNCa, UNSL, UNNOBA, UNT, UNSE, UNComa y UTN-FRSF), y las mismas comenzaron el dictado del curso durante el 1er semestre de 2019. Estos dictados están en pleno proceso mientras escribimos estas líneas.

Los cursantes que participaron de los dictados piloto del curso quedaron extremadamente satisfechos con el material ofrecido, y manifestaron su acuerdo en que los objetivos originales se cumplieron con creces. Hace falta un análisis más profundo de resultados, el cual será posible una vez finalizados los cursos que se encuentran actualmente siendo dictados.

5. Conclusiones

Uno de los grandes aprendizajes que realizamos durante la implementación de los cursos, tanto LPYSD1 como LPYSD2, es la diferencia enorme que significa para los cursantes tener un marco conceptual de referencia y reconocerlo explícitamente. De esta forma, es posible entender rápidamente cuál es el rol de una actividad

¹¹ <https://github.com/gobstones/curso-lpysd2> y <https://program-ar.gitlab.io/lpysd2/> para Gobstones-Web y App Inventor respectivamente (ambos accesibles fácilmente desde dentro de los entornos correspondientes)

determinada en el conjunto, y también construir criterios para juzgar soluciones diferentes a las propuestas, además de percibir al curso como una unidad conceptual integrada. Por esa razón, durante el diseño de este curso prestamos especial atención a la construcción de un marco conceptual que fuera en todo coherente con el de LPYSD1, y que contuviese lo necesario para conseguir los objetivos buscados.

Esperamos que este curso constituya un aporte para los docentes que deciden acercarse a nuestra disciplina.

6. Agradecimientos

Al grupo docente que dicta la materia Introducción a la Programación, primera materia de las carreras Tecnicatura Universitaria en Programación Informática y Licenciatura en Informática con orientación al Desarrollo de Software en UNQ, que viene desarrollando material didáctico, ejemplos y entornos de programación que nos permitieron focalizar con rapidez las actividades en los ejes propuestos. Al grupo de extensión de UBA, quienes siempre han brindado valiosos aportes y ahora nos dieron permiso para utilizar varias actividades de su autoría. A los equipos de todas las universidades seleccionados para dictar esta segunda parte del curso por su feedback, especialmente a los de las universidades piloto (UNQ y UNNE), que permitieron iterar el material y mejorarlo. Y como siempre, al equipo de capacitación de Fundación Sadosky (Herman, Dani, Franco, Hernán, Juli, Javi, Gus, y nuestras más recientes incorporaciones, Tomás y Fernando).

Referencias

[AHU83] Alfred V. Aho, John E. Hopcroft, and Jeffrey Ullman. *Data Structures and Algorithms*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1st edition, 1983.

[Dos15] J. Dostál. *Inquiry-based instruction: Concept, essence, importance and contribution*. PhD thesis, Palacký University, Olomouc, Czech Republic, 2015. ISBN 978-80-244-4507-6.

[FSO15] Pablo M. Factorovich and Federico A. Sawady O'Connor. *Actividades para aprender a Program.AR*. Fundación Sadosky, 2015. In Spanish. ISBN: 978-987-27-4161-7. <http://program.ar/descargas/manual-docente-descarga-web.pdf>.

[Her07] Charles W. Herbert. *An Introduction to Programming Using Alice*. Skills, 1st edition, 2007.

[LJ11] Annette Lamb and Larry Johnson. Scratch: Computer programming for 21st century learners. *Teacher Librarian*, 38(4):64–68, April 2011.

[ML07] David J. Malan and Henry H. Leitner. Scratch for budding computer scientists. *SIGCSE Bull.*, 39(1):223–227, March 2007.

[ML13] Pablo E. Martínez López. Las Bases Conceptuales de la Programación. Una nueva forma de aprender a programar. Author, E-Book, december 2013. In Spanish. ISBN: 978-987-33-4081-9. Ebook URL:
<http://www.gobstones.org/bibliografia/Libros/BasesConceptualesProg.pdf>.

[MLBSO12] Pablo E. Martínez López, Eduardo Bonelli, and Federico Sawady O'Connor. El nombre verdadero de la programación. Una concepción de enseñanza de la programación para la sociedad de la información. *Anales del SSI 2012 - Simposio sobre la Sociedad de la Información, dentro de las 41ras Jornadas Argentinas de Informática (JAIIO)*, pages 1–23, 2012.
URL: http://41jaiio.sadio.org.ar/sites/default/files/1_SSI_2012.pdf.

[MLCAP17] Pablo E. Martínez López, Daniel A. Ciolek, Gabriela B. Arévalo, and Denise Pari. The Gobstones method for teaching computer programming. In *Proceedings of the 43rd Latin American Computing Conference - XXV Simposio de Educación Superior en Computación (SIESC'17), dentro de la XLIII Conferencia Latinoamericana de Informática (CLEI'17)*, CLEI. IEEE, 2017. ISBN 978-1-5386-3057-0,
URL: <https://ieeexplore.ieee.org/document/8226428>

[Pat81] Richard E. Pattis. *Karel the Robot: A Gentle Introduction to the Art of Programming*. John Wiley & Sons, Inc., New York, NY, USA, 1st edition, 1981.

[SP88] P.C. Scholl and J.P. Peyrin. *Schémas algorithmiques fondamentaux: séquences et itération*. Université Joseph Fourier Institut Grenoblois d'études informatiques, 1988.