Capacitación en programación para incorporar el pensamiento computacional en las escuelas

Gladys Dapozo[1], Raquel Petris[1], Cristina Greiner[1], María Cecilia Espíndola[1], Ana María Company[1], Mariano López[1]

1 Facultad de Ciencias Exactas y Naturales y Agrimensura -Universidad Nacional del Nordeste, Corrientes, Argentina

{gndapozo, cgreiner, rpetris}@exa.unne.edu.ar

Resumen

En este trabajo se presentan los resultados de una experiencia de enseñanza de programación que busca incorporar el pensamiento computacional en las escuelas. Se realizó una encuesta a los participantes de la capacitación. De los resultados obtenidos se destaca que la totalidad de los docentes considera importante incorporar la programación en las escuelas para promover el pensamiento lógico y como competencia demandada en la actualidad. Asimismo, surge que los docentes se han apropiado de los conceptos fundamentales de la didáctica de la programación, la mayoría logró implementar en sus espacios curriculares las prácticas propuestas, y los que no lo hicieron alegaron como dificultades que los contenidos no son obligatorios en el sistema educativo argentino, y que carecen del equipamiento necesario. Respecto de las herramientas utilizadas opinan que Scratch es más adecuada para la metodología de enseñanza que se propone.

Palabras clave: Didáctica de la programación. Pensamiento Computacional. Escuela media.

Abstract

This paper presents the results of a computer programming experience aimed to incorporate computational thinking in schools. A survey was conducted among participants. From the results it stands out that all teachers think it is important to include programming in schools to promote logical thinking and as an in-demand skill. Responses also shows that most of the teachers have assimilated the fundamental concepts of programming didactics and managed to implement the proposed practices in the classroom while those who did not claimed as difficulties that the contents are not in the curriculum and their lacking of the necessary equipment. Regarding the tools presented in the training, they deemed Scratch as more suitable for the proposed teaching methodology.

Keywords: Computer programming didactics. Computational thinking. Middle School

1. Introducción

La Fundación Sadosky (www.fundacionsadosky.org.ar) en Argentina, trabaja en la articulación entre el sistema científico-tecnológico y la estructura productiva en el ámbito de la informática y las telecomunicaciones, generando con ello un impacto positivo en la sociedad y en las posibilidades de desarrollo del país. Uno de sus objetivos es incorporar el estudio de la programación en las escuelas argentinas y contribuir al incremento de la matrícula en carreras relacionadas con las Tecnologías de la Información y las Comunicaciones (TIC). Para ello lleva adelante diversos programas, entre estos, Vocaciones en TIC y Program.ar.

El programa Vocaciones en TIC tiene como propósito despertar interés en los jóvenes para estudiar carreras vinculadas con las TIC, en forma más amena y desestructurada, mediante la programación de juegos y animaciones. Para ello, se realizan visitas a las escuelas del nivel medio a fin de realizar talleres de programación, basados en la herramienta Alice (Rebeca en español), orientados a la elaboración de juegos y animaciones, de manera sencilla y amigable.

En tanto, Program.AR tiene como objetivo llevar la enseñanza y el aprendizaje de las Ciencias de la Computación a la escuela argentina. Incluye múltiples aspectos relacionados con la difusión y popularización de la disciplina, la generación de contenidos escolares y la formación docente, entre otros. La propuesta es desarrollada de manera conjunta por la Fundación Sadosky, el portal educ.ar y el Programa Conectar Igualdad.

En este marco, se convocó a las universidades para que contribuyan con los equipos docentes y la infraestructura necesaria para desarrollar las actividades.

La Facultad de Ciencias Exactas y Naturales y Agrimensura de la Universidad Nacional del Nordeste (FaCENA-UNNE) ha adherido a esta iniciativa, y a través de la carrera Licenciatura en Sistemas de Información, apoya y contribuye al logro de los objetivos de Program.Ar. En una primera etapa, se trabajó en el

proyecto Visita a las escuelas, y luego en la formación de docentes, mediante el dictado del curso "La Programación y su didáctica".
En este trabajo se presentan los resultados de la encuesta realizada a los docentes que participaron de la capacitación a fin de recabar información sobre distintos aspectos vinculados con el propósito de llevar las Ciencias de la Computación a la escuela.
Para contextualizar este trabajo, los apartados siguientes dan cuenta del marco de conceptos en los cuales se inserta la propuesta pedagógica desarrollada, luego se describen la metodología, los resultados y las conclusiones.

1.1. Pensamiento Computacional

En el 2006, Jannette Wing manifestó su visión acerca del Pensamiento Computacional, diciendo que, al igual que el español o la aritmética "será una habilidad y una actitud de aplicación universal para todas las personas", dado que en la actualidad las TIC abarcan prácticamente todos los ámbitos de la experiencia humana y modifican las actividades cotidianas: el trabajo, las formas de estudiar, las modalidades para comprar y vender, los trámites, el aprendizaje y el acceso a la salud. Este grupo de conocimientos y herramientas son directamente aplicados bajo la forma de sistemas de información y redes de comunicación, en mayor parte digitales, con el fin de satisfacer necesidades específicas de distintos usuarios. "Enseñar el pensamiento computacional no solamente podría inspirar a las generaciones futuras a entrar en las Ciencias de la Computación dada la aventura intelectual, sino que beneficiaría a la gente en todos los campos" [1].
El objetivo del Pensamiento Computacional (PC) es desarrollar sistemáticamente las habilidades de pensamiento crítico y resolución de problemas con base en los conceptos de la computación. El PC refuerza los estándares educativos en todas las asignaturas para acrecentar la habilidad del aprendiz para solucionar problemas y comprometerse con pensamiento de orden superior. Los estudiantes se comprometen con el PC cuando usan algoritmos para resolver problemas y mejoran la solución de estos con la computación; cuando analizan textos y construyen comunicaciones complejas; cuando analizan grandes grupos de datos e identifican patrones a medida que adelantan investigaciones científicas [2].

1.2. Aprendizaje por descubrimiento

Una de las características más relevantes del aprendizaje por descubrimiento, es que el contenido a ser aprendido, no se facilita en su forma final, sino que tiene que ser descubierto por el sujeto, lo que requiere un rol activo de parte del estudiante [3] que le permitirá aplicar lo aprendido a situaciones nuevas [4]. Existen distintas formas de descubrimiento, desde un descubrimiento "puro", casi autónomo, hasta un descubrimiento guiado, orientado por el profesor. En el contexto de los procesos de enseñanza y aprendizaje en las aulas, se utiliza mayoritariamente este último [5].
Los procedimientos de la enseñanza por descubrimiento guiada, permiten proporcionar a los estudiantes oportunidades para manipular activamente objetos y transformarlos por la acción directa, así como realizar actividades para buscar, explorar y analizar. Estas oportunidades, no solo incrementan el conocimiento de los estudiantes acerca del tema, sino que estimulan su curiosidad y los ayudan a desarrollar estrategias para aprender a aprender y descubrir el conocimiento en otras situaciones [6].
Como no hay una real comprensión hasta que el alumno aplique dicho conocimiento en otras situaciones, el aprender implica describir e interpretar la situación, establecer relaciones entre los factores relevantes, seleccionar, aplicar reglas, métodos, y construir sus propias conclusiones [7].
El construccionismo en pedagogía es una teoría del aprendizaje desarrollada por Seymour Papert que destaca la importancia de la acción, es decir del proceder activo en el proceso de aprendizaje. Se inspira en las ideas de la psicología constructivista y de igual modo parte del supuesto de que, para que se produzca aprendizaje, el conocimiento debe ser construido (o reconstruido) por el propio sujeto que aprende a través de la acción, de modo que no es algo que simplemente se pueda transmitir [8].

1.3. Herramientas

Las herramientas que se describen a continuación son gratuitas y están orientadas a la enseñanza de conceptos de programación mediante el diseño de actividades que favorecen la introducción a la resolución de problemas. Tienen una interfaz amigable y pueden ser utilizadas con escaso entrenamiento, sin embargo, ofrecen distintas posibilidades y niveles de complejidad para la obtención de algún producto.

Scratch

Es una herramienta que permite crear animaciones o videojuegos. Sus creadores señalan que "*el objetivo principal no es preparar a las personas para hacer carrera como programadores profesionales sino cultivar una nueva generación de pensadores creativos y sistemáticos que se sientan cómodos programando para expresar sus ideas*" [9].
La creación de una animación en Scratch se asimila a la construcción con piezas de Lego, reconocido juego consistente en bloques de plástico interconectables. En Scratch los bloques representan instrucciones. Scratch ofrece un conjunto de instrucciones básicas a partir de las cuales se define el comportamiento de los objetos que actúan sobre un escenario. Las instrucciones se clasifican en: movimiento, apariencia, control, datos, operadores, sensores, eventos, lápiz y sonido. El personaje (objeto central) es un gato de color naranja, que además es el logo de la herramienta. Es posible añadir elementos y cambiar el fondo del escenario, lo cual brinda una gran flexibilidad a la hora de crear las animaciones.

Los bloques se encastran en el orden que van a ejecutarse. Una secuencia de bloques puede recibir un nombre, representativo de la acción que desarrolla, y ser incorporado al conjunto de bloques del repertorio original. Estos bloques pueden ser importados como un comando y combinados con otros para crear nuevas animaciones. Esta característica inspiró el nombre de este software, considerando la técnica de "scratching" ("rayar") usada por los Disc Jockeys para pasar música y reutilizar piezas, girando con sus manos para adelante y para atrás los discos, para mezclar clips de música juntándolos de manera creativa.

Scratch tiene una interfaz sencilla y amigable, de fácil y rápido aprendizaje, en relación a otros entornos creados con el mismo objetivo, y en particular resulta mucho más sencillo comparado con cualquier lenguaje de programación ya que no requiere conocer sintaxis, y el resultado de la ejecución es inmediato.

Según sus autores, la franja etaria en la que despierta mayor interés este proyecto se ubica entre los 8 y los 16 años, con un pico en los 12 años [9].

Este proyecto, que puede instalarse en Windows, Mac y Linux, recibió el apoyo de la National Science Foundation, Fundación Intel, Microsoft, Fundación MacArthur, Fundación LEGO, Fundación Code-to-Learn, Google, Dell, entre otros. Se inició en el 2003 en el Laboratorio de Medios del MIT con el lema *Imagina, Crea y Comparte* [10].

Alice

Alice es un entorno de programación integrado que ofrece una interfaz amigable para crear animaciones 3D [11]. Surgió principalmente como una iniciativa para revertir la falta de interés en las ciencias de la computación, y la disminución del interés de los jóvenes en la elección de carreras universitarias de este tipo. Su entorno innovador en programación 3D hace que el crear una animación, un juego interactivo o video sea algo fácil y motivador.

Permite construir mundos virtuales con objetos 3D. Los objetos pueden moverse, girar, cambiar color, reaccionar al ratón, etc. Presenta una interfaz que permite generar instrucciones al arrastrar y soltar elementos gráficos (drag and drop), que se corresponden con un lenguaje de programación (Java). Es posible observar en forma inmediata el código que se genera, y esto permite a los estudiantes entender con mayor facilidad la relación entre el código y el comportamiento de un objeto. El entorno visual favorece la retención y el aprendizaje, evitando la frustración de errores de sintaxis. Los objetos se vuelven obvios y el estudiante puede relacionarse con ellos y la forma en que se programan.

Si bien presenta una interfaz amigable, Alice está diseñado para ser la primera exposición a una programación orientada a objetos, por lo que resulta un tanto más compleja, a la vez que permite generar con menos esfuerzo productos de un mayor nivel de complejidad, dado que es posible controlar eventos y programar juegos interactivos. Por estas características, Alice podría ser más adecuada para estudiantes de últimos años del nivel medio, e inicio del universitario.

Aunque la construcción de animaciones es más simple que usando un lenguaje de programación convencional, el entrenamiento de docentes y alumnos demanda un esfuerzo considerable [10].

Lightbot

Es un software orientado a la enseñanza de conceptos de programación. Cuenta con distintos niveles, en los que plantea un escenario tridimensional donde el agente puede moverse hacia adelante, girar en cualquier dirección y saltar [12]. El objetivo es resolver una serie de puzzles, consistentes en llevar a un robot a la casilla azul. En un recuadro aparecen las acciones que puede realizar el robot en el nivel: avanzar, girar 90º en un sentido y en el otro, saltar y encender una bombilla, entre otros. Permite crear procedimientos que pueden ser llamados desde la zona donde se ejecutan las acciones.

Una vez seleccionadas las acciones, al pulsar un botón, el robot realizará las acciones programadas. Si son correctas, se accede al siguiente nivel. Caso contrario, se deberá repetir el nivel. Este juego no enseña un lenguaje de programación, sino que hace que el usuario, inconscientemente, especifique el algoritmo que permita al robot alcanzar la casilla de salida. La interfaz es sencilla, con colores planos y con el mínimo texto posible, de fácil aprendizaje, y las posibilidades son bastante acotadas. El foco de atención recae en el robot y en los obstáculos que debe superar [13].

N6 Max (Robot)

El N6 Max es un robot educativo basado en la plataforma Arduino. Tanto el hardware como el software utilizado son libres, es decir que las especificaciones de la electrónica y los programas necesarios para su utilización están accesibles libremente. Posee una placa controladora llamada DuinoBot la cual se puede programar utilizando el entorno Arduino. Fue fabricado por RobotGroup, una empresa argentina dedicada a la robótica educativa [14].

Para trabajar con el robot, se deben llevar a cabo una serie de acciones, como conectarlo, ingresar al entorno de programación a utilizar, verificar la correcta colocación de las baterías, conectar el módulo de comunicaciones, y luego escribir los mensajes apropiados. Como se trabaja con el robot en modo interactivo, todos los mensajes que se le envían se ejecutarán inmediatamente [15].

Si bien requiere de un conjunto de acciones para hacer funcionar al robot, éste despierta un gran interés entre los jóvenes, por la posibilidad de hacerlo actuar.

2. Metodología

El curso impartido fue diseñado por el equipo de expertos de la Fundación Sadosky. Comprende una introducción a la programación y su didáctica, a la vez que algunos elementos de algoritmia y estructuras de datos. Su objetivo es proveer formación específica a docentes de los distintos

niveles educativos no universitarios que deseen tener un primer acercamiento tanto a la programación como a su enseñanza.
La FaCENA-UNNE conformó un equipo docente (docentes-tutores), formado principalmente por profesores de la asignatura Algoritmos y Estructura de Datos I, que fue capacitado por especialistas de la Fundación Sadosky.
Se difundió a nivel nacional el lanzamiento del dictado de los cursos en las universidades seleccionadas en la convocatoria. Los docentes de los distintos niveles educativos, primario, secundario y terciario, que manifestaron interés formalizaron su inscripción mediante un formulario con el correspondiente aval de su institución.
Como objetivos del curso se propusieron: Comprender por qué enseñar Ciencias de la Computación, Conceptualizar la noción de programa, Incentivar a los alumnos a ser creadores de programas y no sólo usuarios de aplicaciones hechas por terceros, Ejecutar programas diseñados por los propios alumnos, Detectar y corregir errores de los programas propios y de los alumnos, Planificar la solución a un problema de programación como la división en subproblemas, e identificar a estos, Comprender el desarrollo de la capacidad de abstracción que permiten las nociones de estado de un programa, identificación de patrones, repetición fija o condicional y parámetros, Trabajar conceptos relacionados con las Ciencias de la Computación para desarrollar habilidades de pensamiento, Diseñar propuestas áulicas creativas aplicando los conocimientos trabajados
Para lograr los objetivos previstos, los contenidos fueron: Ciencias de la Computación. Importancia de enseñar Ciencias de la Computación. Comandos (acciones) y valores (datos). Procedimientos. Programas. Noción de programa y autómata. División en subtareas. Repeticiones simples. Alternativas condicionales. Repeticiones condicionales. Parámetros. Uso de las herramientas Alice, Scratch, Lightbot y robots N6 Max. Identificación de patrones, repetición fija o condicional y parámetros. Resolución de problemas. Modos de abordaje.
La estrategia pedagógica estuvo centrada en el aprendizaje basado en problemas [16] y en las metodologías de enseñanza de programación propuestas por la Universidad Nacional de Quilmes [17], la Universidad Nacional de Córdoba y el equipo de expertos de la Fundación Sadosky.
Con una modalidad dinámica, en las clases se desarrollaron actividades que buscan que los docentes asistentes incentiven a sus alumnos a que se animen a ser creadores de programas y no sólo usuarios de aplicaciones hechas por terceros.
Las clases fueron presenciales, con una carga horaria de 70 horas. Se dictaron 2 clases semanales de 2 hs. reloj cada una, desde julio a noviembre de 2015.
En el desarrollo del curso, cada clase se iniciaba con una breve explicación de los conceptos, en algunos casos se realizaba una actividad "un-plugged" (sin computadora) para mejorar la comprensión de los mismos. Luego se planteaban situaciones problemáticas, plasmadas en actividades predefinidas, que los participantes intentaban resolver utilizando las herramientas específicas (Scratch, Lightbot, Alice, robot), bajo la supervisión de los docentes-tutores, atentos a guiar a los cursantes para lograr el objetivo.
Para aprobar el curso, los docentes asistentes cumplieron con más del 75% de asistencia, elaboraron la planificación de las actividades para implementar en el aula los conceptos de programación y dictaron 8 hs. didácticas de clases en sus espacios curriculares. Los docentes del nivel terciario se organizaron para el dictado de las clases en escuelas del nivel medio. Más de 60 docentes aprobaron el curso cumpliendo las condiciones requeridas.
Para obtener información acerca del aprovechamiento de esta capacitación por parte de los docentes asistentes, y recabar información a modo de retroalimentación a esta interesante propuesta de la iniciativa Program.Ar, se elaboró un cuestionario en Google Docs, cuyo link se envió a los docentes para que lo completaran.
Los datos obtenidos fueron procesados con una planilla de cálculo y los resultados se describen en la siguiente sección.

3. Resultados

3.1. Caracterización de los docentes

Los docentes que realizaron la capacitación y contestaron las encuestas fueron 26. El 54% son mujeres y el 46% son varones, con una edad promedio de 42 años. Proceden mayoritariamente de la ciudad de Corrientes.
En cuanto a las titulaciones, tanto universitarias como terciarias, en su mayoría están vinculadas con la Informática y las Tecnologías.
En la Tabla 1 se puede apreciar que el 63% de los encuestados posee título universitario. Se destacan los títulos de grado y pregrado de la carrera Licenciatura en Sistemas de Información de la UNNE.

Tabla 1: Títulos universitarios

Título Universitario	Cant.	%
Ingeniero en Sistemas	1	6%
Licenciado en Sistemas de Información	7	44%
Programador Universitario de Aplicaciones	3	19%
Licenciada en Tecnología Educativa	2	13%
Profesorado Universitario en Informática	1	6%
Profesorado en Matemática y Cosmografía	1	6%
Analista de Sistemas	1	6%
Ingeniero Industrial	1	0%
Total	17	100%

En la Tabla 2 se puede apreciar que el 69% de los encuestados posee título terciario, mayoritariamente de Profesor en Tecnología (44%).

Tabla 2: Títulos terciarios

Título terciario	Cant	%
Profesor en Tecnología	8	44%
Prof. en Educación Técnica Profesional	3	17%
Docente	1	6%
Especialista en TIC	1	6%
Analista en Computación Administrativa	1	6%
Profesora de EGB I y II	1	6%
Profesor en Informática	1	6%
Técnico Superior en Informática	1	6%
Técnico en Informática administrativa	1	6%
Totales	18	100%

3.2. Nivel educativo

En la Figura 1 se puede apreciar que los docentes mayoritariamente ejercen la docencia en el nivel secundario (58%), y se desempeñan en todos los casos, excepto en uno, en el espacio curricular "Tecnología/Informática".

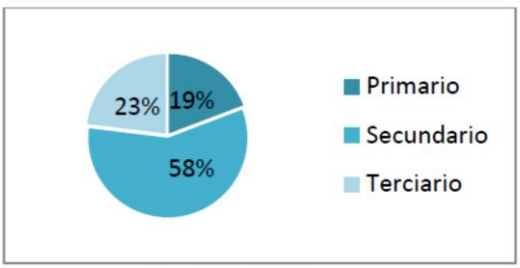

Fig. 1. Nivel educativo

La mayoría de los docentes posee una antigüedad en el espacio curricular de 1 a 5 años (ver Figura 2).

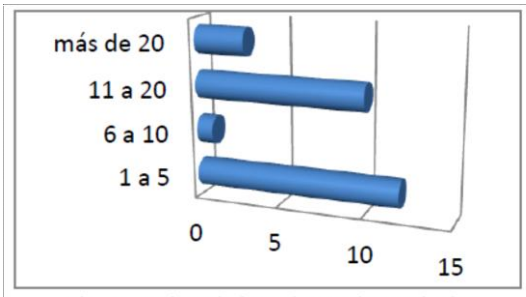

Fig. 2. Antigüedad en el espacio curricular

3.3. Experiencia en programación

De su experiencia en programación, el 58% indica que programó alguna vez. Los que programan actualmente constituyen un 23% así como los que nunca programaron, tal como se ilustra en la Figura 3.

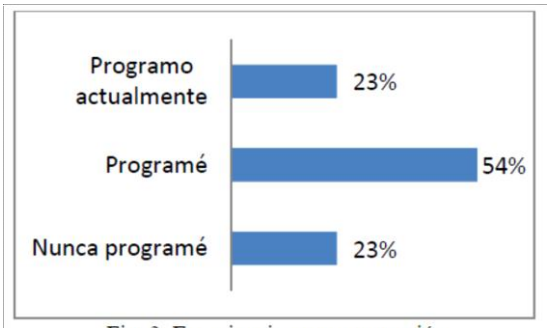

Fig. 3. Experiencia en programación

Los docentes que programan en la actualidad utilizan principalmente Visual Basic y Java, como se puede ver en la Tabla 3.

Tabla 3: Lenguajes en los que programan

Lenguajes de programación	Cantidad	%
Visual Basic	5	83%
Java	5	83%
Pascal	4	67%
PHP	3	50%
Python	1	17%
C	1	17%

Los docentes que programaron en el pasado utilizaron Visual Basic y Pascal, como se muestra en la Tabla 4. Se puede inferir que corresponde a la práctica que se da en la carrera de Sistemas.

Tabla 4: Lenguajes en los que programaron

Lenguajes de programación	Cantidad	%
Visual Basic	12	86%
Pascal	10	71%
PHP	2	14%
Java	1	7%
Python	1	7%
C	1	7%

Se comparó la experiencia en programación con el grado de dificultad con el que resolvieron los ejercicios del curso, estableciendo las siguientes categorías:
Niveles de dificultad:
1 Con mucho esfuerzo los resuelve
2 Realiza 2 o 3 intentos hasta lograrlo
3 Ninguna, son muy fáciles

Tabla 5: Experiencias en programación y niveles de dificultad

Experiencia en programación	Con mucho esfuerzo	Realiza 2 o 3 intentos	Ninguna
Nunca programé	17%	67%	17%
Programé	0%	57%	43%
Programo actualmente	0%	33%	67%

De los que nunca programaron, el 67% tuvieron que realizar 2 o 3 intentos hasta resolver el problema. En tanto, de los que manifestaron programar actualmente, el 67% no tuvo dificultades para resolver los problemas.

3.4. Herramientas utilizadas

Respecto a las herramientas utilizadas en el curso con el propósito de que los docentes asistentes enseñen a sus alumnos los conceptos básicos de la programación, el 81% considera que Lightbot es lúdica, didáctica y entretenida, seguida de Scratch, con un 69%. En tanto, el 62% de los docentes opinó que Alice es compleja. Si bien el 15% señala que la programación del robot también es compleja, no descartan que resulte de mucho interés para los alumnos (50%), tal como se muestra en la Figura 4.

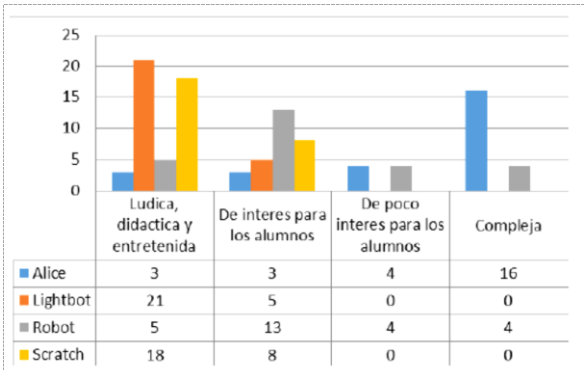

Fig. 4. Herramientas

En cuanto a cuáles de estas herramientas consideran más relevante desde el punto de vista de la didáctica para la enseñanza de la programación, el 85% se volcó hacia Scratch, mientras que señalaron que Alice es la menos relevante para este objetivo (35%), como se puede apreciar en la Figura 5.

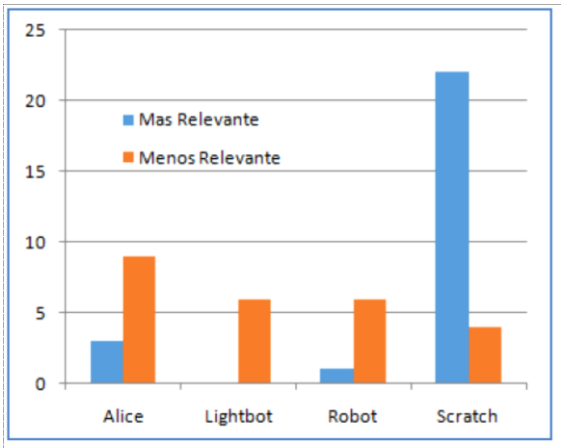

Fig. 5. Relevancia de las herramientas

Respecto de esta visión de los docentes participantes, cabe señalar que en el curso se utilizaron actividades predefinidas sobre las cuales se aplicaban diferentes técnicas de programación para lograr los resultados, siendo en la mayoría de los casos realizados con Scratch.
Las actividades con Alice fueron pocas, con el mismo enfoque de actividades predefinidas, y no se visualizó el potencial de esta herramienta para la elaboraciòn de animaciones y juegos, actividades que sí se realizan en los talleres que se dictan en el proyecto Visita a las Escuelas, también iniciativa de la Fundación Sadosky, que entusiasma a los alumnos [18].

3.5. La programación en las escuelas

A la pregunta de si considera importante incorporar la programación en el sistema educativo argentino, el 100% de los encuestados contestó positivamente, y la justificación de la respuesta se clasificó en las categorías que se muestran en la Tabla 6. Como se puede apreciar la mayoría de los docentes (46%) considera que es importante porque promueve el pensamiento lógico para la resolución de problemas, en tanto otro grupo (25%) considera que otorga habilidades y competencias demandadas en la actualidad, teniendo en cuenta que “La competencia digital requiere no solamente tener habilidad para chatear, navegar o interactuar sino también la habilidad de diseñar, crear e inventar con los nuevos medios” [19].

Tabla 6: Importancia de la programación en las escuelas

Importancia	Total	%
Pensamiento lógico para la solución de problemas	11	46%
Habilidades y competencias requeridas en la actualidad	6	25%
Actividad accesibles a docentes y alumnos	1	4%
Hace que las personas no sean meros consumidores de tecnología y que	1	4%
Estimula la creatividad	1	4%
Estimula pensamiento crítico	1	4%
Entender la funcionalidad y utilidad de un dispositivo.	1	4%
Por el avance de la tecnología	1	4%
Soberanía tecnológica y la producción de software nacional	1	4%
Totales	24	100%

Se consultó a los docentes si incorporaron los contenidos a sus asignaturas. El 42% contestó que lo hizo, el 35% hizo una incorporación parcial de los contenidos y un 23% no incorporó los contenidos en su práctica docente, tal como se puede apreciar en la Figura 6.

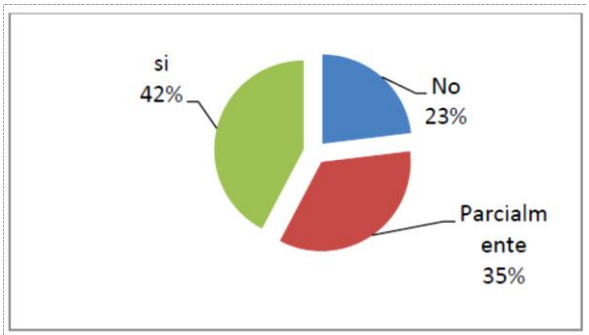

Fig. 6. Implementación de los contenidos

Entre las dificultades señaladas para la incorporación de los contenidos del curso en las aulas, los docentes destacaron como cuestión principal que los contenidos no forman parte del sistema educativo obligatorio, y que carecen de equipamiento necesario (ver Tabla 7).

Tabla 7: Dificultades para la incorporación de los contenidos

Dificultades	Cant	%
Falta de equipamiento	7	37%
No están en los contenidos curriculares	8	42%
Falta de interés en las autoridades	3	16%
Ninguna	1	5%
Totales	19	100%

3.6. Conceptos fundamentales de la didáctica de la programación

En la propuesta formativa se insistió en los conceptos de "abstracción", referido a pensar una estrategia antes de intentar la resolución de los ejercicios, la "descomposición del problema en partes", traducida a la creación de bloques en la solución diseñada y en la "legibilidad" de la solución, mediante la definición de nombres representativos a los bloques. En la Figura 7, se muestra que los docentes, en su mayoría, aplicaron la forma de trabajo propuesta.

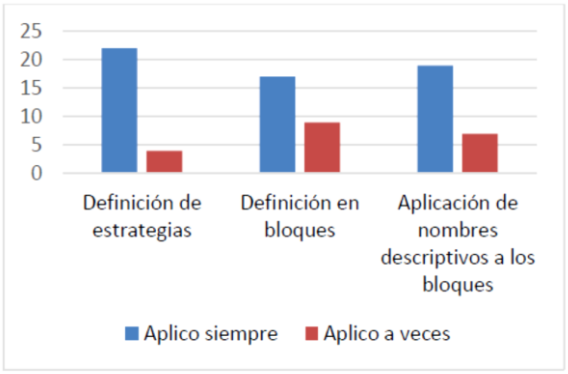

Fig. 7. Aplicación de conceptos fundamentales

4. Conclusiones

La iniciativa Program.Ar tiene como propósito promover "cambios de fondo en la enseñanza en escuelas primaras y secundarias de varios temas relacionados con la computación, convencidos de que son un elemento clave para que el país pueda aprovechar las enormes oportunidades que brindan estas tecnologías". Se cree además que aprender la "verdadera computación" (las ciencias de la computación) será muy beneficioso para que todos los alumnos argentinos desarrollen habilidades y competencias fundamentales para la vida moderna [20].
En línea con estos objetivos, se ha realizado en el contexto de la UNNE la capacitación a docentes de los distintos niveles educativos no universitarios. Se destacan los resultados favorables en cuanto a la apropiación de los conceptos fundamentales de la didáctica de la

programación por parte de los docentes asistentes, la mayoría logró aplicar en el aula las prácticas propuestas, y los que no lo hicieron alegaron como principal dificultad que los contenidos no son obligatorios y que no poseen el equipamiento necesario. Respecto de las herramientas utilizadas opinan que Scratch es la más adecuada para la metodología de enseñanza que se propone.

Como línea futura se propone revisar las actividades y modalidades para la capacitación de los docentes, elaborando estrategias o alternativas superadoras a las dificultades detectadas para lograr en el corto o mediano plazo la incorporación de la programación en las escuelas, buscando abarcar a un mayor número de instituciones y docentes.

5. Agradecimientos

A la Fundación Sadosky por la capacitación al equipo de instructores y por el material didáctico provisto.

6. Referencias

[1] Wing, J. M. (2006). "Computational Thinking". Communications of the ACO. Vol. 49, No. 3

[2] ISTE and CSTA (2011). Computer Science Teachers Association and the International Society for Technology in Education. "Pensamiento Computacional, Caja de Herramientas". Eduteka. Disponible en: http://www.eduteka.org/modulos/9/272/2062/1 .

[3] Martínez, E. R. y Zea, E. (2004). "Estrategias de enseñanza basadas en un enfoque constructivista". Revista Ciencias de la Educación. 2 (24):69-90.

[4] Bruner, J. (1966). "Toward a Theory of Instruction". Cambridge, MA: Harvard University Press.

[5] Woolfork, E.A. (1999). "Psicología Educativa". España: Pearson

[6] Good, T. y Brophy, J. (1995). "Introducción a la Psicología del Aprendizaje. Psicología Educativa Contemporánea". España: McGrawHill

[7] Bruner, J. (1980). "Investigación sobre el desarrollo cognitivo". España: Pablo del Río.

[8] Papert, S. y Harel, I. (2002). "Situar el Construccionismo". INCAE (Centro Latinoamericano para la Competitividad y el Desarrollo Sostenible) - Media Lab del Instituto Tecnológico de Massachusetts.

[9] Resnick M., Maloney J, Monroy-Hernandez A., Rusk N., Eastmond E., Brennan K, Millner A., Rosenbaum E., Silver J., Silverman B., Kafai Y. (2009). "Scratch: Programming for All". Communications of the ACM, Vol. 52, No. 11

[10] Rueda, S.; Cohen, A.; Delladio, T.; Gottifredi, S.; Tamargo, L. (2014). "Herramientas para apoyar el descubrimiento de vocaciones en Ciencias de la Computación". XX Congreso Argentino de Ciencias de la Computación. Buenos Aires.

[11] Utting I., Cooper S. Kölling M., Maloney J., Resnick M. (2010). "Alice, Greenfoot, and Scratch - A Discussion". ACM Transactions on Computing Education. Vol. 10 Issue 4, Article No. 17.

[12] Prieto, F.L. (2015). "Resolución de problemas de programación de videojuegos con planificación automática". Treball fi de grau. Grau en Enginyeria en Informàtica. Escola Superior Politécnica. Universitat Pompeau Fabra. Barcelona. Disponible en: http://repositori.upf.edu/handle/10230/25517

[13] Castro Saturio, L.; García Segador, S; Hernández, M. (2015). "Videojuegos para aprender a programar videojuegos". Trabajo de fin de grado en Ingeniería Informática. Facultad de Informática. Universidad Complutense de Madrid. Disponible en: http://eprints.ucm.es/32853

[14] N6 Max. Productos. Robots. http://robotgroup.com.ar/index.php/es/productos/robots/n6-max-detail

[15] Díaz, J.; Banchoff, C.; Martín, S.; Bogado, J.; Mel, D.; López, F. (2012). "Manual de programación con robots para la escuela". Versión 0.1.5.

[16] Torp, L.; Sage, S. (1998). "El Aprendizaje Basado en Problemas". Editorial Amorrortu.

[17] Martínez López, P.E.; Bonelli, E.A.; Sawady O'Connor, F.A. (2012). "El nombre verdadero de la programación. Una concepción de la enseñanza de la programación para la sociedad de la información". Anales del 10mo Simposio de la Sociedad de la Información (SSI'12), 41 Jornadas Argentinas de Informática (JAIIO '12), 1–23. ISSN 1850-2830.

[18] Dapozo, G.; Greiner, C.; Pedrozo Petrazzini, G.; Chiapello, J. (2014). "Vocaciones TIC. ¿Qué tienen en común los alumnos del nivel medio interesados por carreras de Informática?" IX Congreso sobre Tecnología en Educación & Educación en Tecnología. ISBN: 978-987-24611-1-9. p. 128-137.

[19] Resnick, M. Sowing (2007). "The seeds for a more creative society". Learning and Leading with Technology. (Dec. 2007), 18–22.

[20] Fundación Sadosky (2013). "CC – 2016 Una propuesta para refundar la enseñanza de la computación en las escuelas argentinas". https://sites.google.com/a/fundacionsadosky.org.ar/portal-computar/-quienes-somos/ReporteEducacionPrimariaSecundaria.pdf?attredirects=1

Dirección de Contacto del Autor/es:

Gladys Noemí Dapozo
Santa Cruz 1911
(3400) Corrientes
Argentina
e-mail: gndapozo@exa.unne.edu.ar

Raquel H. Petris
Junín 1956
(3400) Corrientes
Argentina
e-mail: rpetris@exa.unne.edu.ar

Gladys N. Dapozo. Magister en Informática y Computación (UNNE). Directora Licenciatura en Sistemas de Información (UNNE). Coordinadora en la UNNE de las actividades de Program.Ar.

Raquel H. Petris. Magister en Informática y Computación (UNNE). Profesora Titular Profesorado y Licenciatura em Matemática (UNNE). Integrante del equipo técnico actividades Program.Ar en la UNNE.