Actas de las Jenui, vol. 3. 2018.
Páginas: 263-270

¿Realmente la programación puede ser para todos? Análisis de la experiencia dentro de un MOOC

David Bañeres, Carlos Casado, Adriana Ornellas, Elena Planas, Josep Prieto, Montse Serra
Estudios de Informática, Multimedia y Telecomuncación
Universitat Oberta de Catalunya
Barcelona
{dbaneres, ccasadom, aornellas, eplanash, jprieto, mserravi}@uoc.edu

Resumen

Cuando se analiza la posibilidad de que toda persona pueda aprender a programar surgen diferentes incógnitas sobre cuál es la mejor forma de abordar la cuestión. El lenguaje de programación, la metodología, la tipología de ejercicios, o la duración de un curso son variables que pueden determinar el éxito o el fracaso del mismo. En este artículo se presenta una experiencia docente en la impartición de un MOOC (Massive Open Online Course) para iniciarse en la programación dirigido a todos los públicos. Se describen y discuten el diseño, el desarrollo y los resultados alcanzados con el objetivo de discernir si un curso de estas características puede ser factible para iniciar a cualquier persona en la programación de ordenadores.

Abstract

Many questions appear when the statement "any citizen is able to learn programming language" is analysed. The programming language, the methodology, the type of exercises or the duration of the course are variables that may affect the success or the failure of an introductory course. This paper presents a teaching experience of a MOOC course aiming at introducing programming suitable for all audiences. The design, the execution and the results are described along with a discussion whether this type of course can be feasible as an introductory course to learn language programming.

Palabras clave

Programación para todos, Scratch, MOOC, pensamiento computacional.

1. Motivación

En los últimos años observamos el crecimiento de un movimiento muy fuerte que propone que todos los estudiantes, de todas las escuelas, aprendan a programar. La programación se presenta como una manera de empoderar a los ciudadanos; de obtener un grado más de libertad dentro de una sociedad cambiante; de ser autónomo para automatizar algún proceso simple o repetitivo y sobretodo como algo accesible a todos los públicos de todas las edades. La idea de la programación como algo fácil y al alcance de todos contrasta, sin embargo, con la dificultad que los estudiantes del Grado de Informática tienen para superar las asignaturas introductorias de programación [3].

Actualmente, plataformas de MOOC como Coursera, edX o Miriada X, ofrecen numerosos cursos de introducción a la programación dirigidos a todos los públicos y que no requieren conocimientos previos. Podemos encontrar tipologías diversas de cursos con diferentes enfoques, utilizando distintos lenguajes de programación, basados en el diseño de aplicaciones, el desarrollo de juegos, entre otras opciones.

Ante la numerosa oferta de cursos dirigidos a todos los públicos de algunas de las universidades de referencia mundial como por ejemplo el MIT (Massachusetts Institute of Technology) o Berkeley, las preguntas que emergen y a las que intentaremos dar respuesta en el apartado de discusión del presente trabajo es ¿se podría aplicar esta metodología MOOC a la iniciación a la programación en la universidad? ¿sería efectiva? En otros ámbitos como, por ejemplo, aprendizaje de sistemas digitales ya se aplicado [9].

En el mundo de la educación va tomando fuerza lo que se conoce como *pensamiento computacional* (*Computational Thinking*). El pensamiento computacional, término definido por Jeannette Wing [11], es un conjunto de habilidades que implica resolver problemas, diseñar sistemas y comprender el compor-

tamiento humano, haciendo uso de los conceptos fundamentales de la informática. Quizás empezar por introducir este concepto tal como se hace actualmente en algunos proyectos piloto en educación primaria [4] puede solucionar parcialmente el problema con los estudiantes de programación de primer curso en los grados tecnológicos.

En este artículo presentamos la experiencia de un MOOC de introducción a la programación mediante la metodología *learning-by-doing* [1]. Participantes de todas las edades sin ningún conocimiento previo de programación desarrollaron durante el curso un proyecto con el lenguaje gráfico Scratch del juego clásico Arkanoid[1]. De forma intuitiva los participantes en el último proyecto llegaron a utilizar conceptos complejos de programación como listas, clases e instancia. Por lo tanto, sin introducir estos conceptos de forma teórica, los participantes fueron capaces de utilizarlos para desarrollar su propio proyecto.

El presente artículo se estructura de la siguiente forma. La sección 2 introduce el contexto donde se enmarca la experiencia. La sección 3 describe los objetivos del curso, mientras que la metodología de aprendizaje y los contenidos del curso se explican en las secciones 4 y 5, respectivamente. Los resultados de la experiencia se analizan y se discuten en las secciones 6 y 7, y finalmente en la sección 8 presentamos las conclusiones del estudio.

2. Contexto de la experiencia

Desde el año 2013 el grupo INVENTA[2], creado por profesores de los estudios de Informática Multimedia y Telecomunicación de la Universitat Oberta de Catalunya (UOC) y apoyado por el área de globalización y cooperación de la misma universidad, promueve actividades con el objetivo de desarrollar el pensamiento computacional mediante el acercamiento de la programación a la sociedad.

Buena parte de las actividades impulsadas hasta ahora, tales como los talleres presenciales impartidos en numerosas escuelas de educación primaria catalanas (Barcelona, Lleida y Sabadell, entre otras), los talleres en hospitales infantiles y los clubes de programación, tienen como público objetivo niños y niñas y jóvenes. Estos talleres, muy bien valorados tanto por parte de los docentes como de los participantes, nos permiten acercar la programación a los más pequeños e iniciarlos en esta disciplina desde edades tempranas. También otras actividades, dirigidas al público adulto, se han impulsado hasta el momento. Destacamos la formación de voluntarios tecnológicos, adultos sin formación técnica requerida, que han sido formados en Scratch con el objetivo de convertirse en formadores en escuelas, hospitales o clubes de programación. Otro ejemplo de acción realizada es el MOOC que se describe en este artículo.

En términos numéricos, hasta la actualidad alrededor de 200 familias han participado en los talleres celebrados durante el Scratch Day; cerca de 900 niños y niñas han participado en los talleres realizados en escuelas; más de 200 niños y niñas han participado en los clubes de programación; y más de 2000 personas han participado en nuestras formaciones presenciales y online. Actualmente INVENTA colabora con diversas entidades latinoamericanas (Colombia, Ecuador y Perú, entre otras) para organizar actividades similares, de las cuales ya se han beneficiado 45 formadores y 200 niños y niñas.

3. Objetivos del curso

El MOOC de *Programación para todos con Scratch* está dirigido a todas aquellas personas cuyo interés es iniciarse en el mundo de la programación; profesionales de la educación que deseen introducir una herramienta interactiva en sus clases; y padres y madres que deseen introducir de una forma lúdica a sus hijos e hijas en el mundo de la programación y, en su defecto, en el mundo de la tecnología.

El objetivo general del curso es introducir a los participantes en los fundamentos de Scratch: el construccionismo y el pensamiento computacional. El enfoque del curso es eminentemente práctico, explicando el funcionamiento básico de Scratch y descubriendo todas sus potencialidades a partir de actividades guiadas.

A lo largo del curso también se presentan herramientas y estrategias para, una vez finalizado el MOOC, usar Scratch como herramienta de aprendizaje de forma lúdica e interactuar con diferentes dispositivos físicos, así como otros entornos de programación similares que pueden usarse tanto en un aula como en casa.

4. Metodología de aprendizaje

El curso se basa en los principios del aprendizaje basado en problemas, en la teoría construccionista de Seymour Papert [7] y en la metodología *learning-by-doing* (aprender haciendo) [1] que pone énfasis en la práctica frente a la teoría. Esta manera de aprender permite experimentar y descubrir siguiendo la propia lógica y aprender de los propios aciertos y errores. Por esta razón, esta metodología es idónea para fijar conocimientos y hacer que el aprendizaje sea efectivo. Además, es una metodología que se adapta perfectamente al ritmo de cada persona teniendo en cuenta el tiempo de dedicación.

[1] Ejemplo de juego Arkanoid con Scratch https://scratch.mit.edu/projects/14794072/

[2] Página web del grupo: http://inventa.uoc.edu

Figura 1: Entorno Open edX con el ejemplo de descripción de los bloques de control.

El curso también introduce el concepto de pensamiento computacional como un conjunto de habilidades que implica resolver problemas, diseñar sistemas y comprender el comportamiento humano, haciendo uso de los conceptos fundamentales de la informática.

Además, incluye actitudes o disposiciones relacionadas con:

- La confianza en el manejo de la complejidad.
- La persistencia en el trabajo con problemas difíciles.
- La tolerancia a la ambigüedad.
- La capacidad para hacer frente a los problemas no estructurados.
- La capacidad de comunicarse y trabajar con otros para conseguir un objetivo común o una solución.

El lenguaje utilizado en el MOOC ha sido Scratch [8], un entorno de programación pensado para enseñar a niños y niñas y adolescentes a programar. Para facilitar su uso las instrucciones no tienen que escribirse si no que son bloques que deben arrastrarse al espacio de programación. Además, están agrupadas según su utilidad. Para los niños y niñas resulta muy sencillo utilizar el entorno y aprenden a usar estructuras como los bucles o los condicionales de una manera intuitiva, sin ser conscientes de estar programando. Nuestra experiencia en los cursos de formación de profesorado nos demuestra que no sólo a los niños y niñas les resulta sencillo aprender a programar con Scratch, sino que también a los adultos.

El curso contiene diferentes recursos de aprendizaje:

- Vídeos: Algunos de los conceptos se presentan en formato vídeo. Básicamente los conceptos más teóricos, antecedentes en el pensamiento computacional y la interacción entre Scratch y los dispositivos físicos.
- Ejemplos: Los conceptos relacionados con Scratch se presentan mediante ejemplos. Se decidió evitar hacer vídeos sobre las instrucciones (bloques) de Scratch ya que usar esta metodología de presentación sería incompatible con la metodología *learning-by-doing*. El participante tiene una explicación en texto de cada grupo de instrucciones de Scratch y un ejemplo que recoge algunas de las instrucciones (ver Figura 1). El ejemplo está hecho con Scratch donde el usuario puede interactuar, ver el código y modificarlo a su antojo.
- *Test yourself* (pruébate a ti mismo): Después de la explicación de cada grupo de bloques, hay una actividad no evaluable para practicar. La mayoría de las veces consiste en modificar el ejemplo propuesto introduciendo alguna funcionalidad

Semana	Recursos de aprendizaje	Actividades
0	- Introducción al curso y a la plataforma Open edX (video)	
1	- Introducción a Scratch y al pensamiento computacional. Antecedentes y alternativas a Scratch. La comunidad de usuarios Scratch (vídeos)	- Resolver las cartas introductorias de Scratch - Diseñar una carta Scratch (actividad evaluable)
2	- Los bloques básicos de Scratch (ejemplo, *test yourself*)	- Cuestionario I (actividad evaluable) - Desarrollo juego Arkanoid (Parte I) (actividad evaluable) - Revisión por pares diseñar una carta de Scratch
3	- Los bloques avanzados de Scratch: Clones, Paso de mensajes y Listas (ejemplo, *test yourself*)	- Desarrollo juego Arkanoid (Parte II) (actividad evaluable) - Diseño del proyecto personal (actividad evaluable) - Revisión por pares Arkanoid (Parte I)
4	- Interacción con el micrófono, cámara, Makey-Makey y Lego-WeDo (vídeo)	- Cuestionario II (actividad evaluable) - Desarrollo juego Arkanoid (Parte III) (actividad evaluable) - Desarrollo del proyecto personal (actividad evaluable) - Revisión por pares Arkanoid (Parte II) - Revisión por pares de diseño del proyecto personal (actividad evaluable)
5	- Antes y después de Scratch y alternativas (vídeo)	- Revisión por pares Arkanoid (Parte III) - Revisión por pares de desarrollo del proyecto personal

Cuadro 2: Contenidos y planificación del curso.

adicional. Scratch permite clonar un proyecto y modificarlo sin alterar el proyecto inicial. Esta funcionalidad permite copiar el ejemplo y modificarlo como se quiera sin afectar al ejemplo inicial.

Las actividades de evaluación son de dos tipos:

- Cuestionario automático: La plataforma MOOC Open edX[3] permite crear cuestionarios que se evalúan automáticamente. Es una forma de evaluación efectiva en los MOOC cuando hay un gran volumen de participantes ya que fácilmente se pueden evaluar los conocimientos teóricos de todos los participantes.
- Proyectos con evaluación *peer-review* (por pares): Los cuestionarios no son útiles cuando se debe evaluar una creación como puede ser un proyecto con Scratch. Por esta razón se adopta la evaluación por pares de los proyectos realizados por los participantes. Cada actividad tiene una rúbrica definida por los profesores con diferentes indicadores. Los participantes, para obtener nota de su propia actividad, deben de evaluar con dicha rúbrica tres proyectos de otros compañeros. Nótese que esta evaluación permite ver cómo otras personas han resuelto el mismo problema y aprender de los errores de otros y mejorar los propios conocimientos.

Finalmente, comentar que el curso tiene un foro asociado. Además, el curso se ha configurado para que cada recurso (ejemplo y actividad) tenga una entrada en el foro, clasificando fácilmente las consultas y comentarios por recurso. De esta forma, los participantes pueden acceder fácilmente a todos los comentarios relacionados con un contenido concreto.

5. Contenidos y planificación del curso

Los contenidos y actividades del curso que se desarrollaron durante 5 semanas se describen en el Cuadro 1. Dentro de cada semana se presentan los contenidos y el tipo de recurso docente. En relación con las actividades, se especifica si es una actividad evaluable. El contenido está planificado para que el participante invierta aproximadamente de 4 a 6 horas semanales.

Los vídeos, además de estar enlazados en el curso, tenían una lista de reproducción propia en Youtube[4]. Los proyectos se definieron de dos tipos:

- Proyecto Arkanoid[1]: Se propuso la implementación del juego clásico Arkanoid. El proyecto se dividió en 3 partes basándose en tres niveles de dificultad que dependía de los bloques y características que se introducían en el juego. El

[3] Página web del proyecto Open edX: https://open.edx.org/

[4] Lista de reproducción Youtube: http://goo.gl/jc9HUu

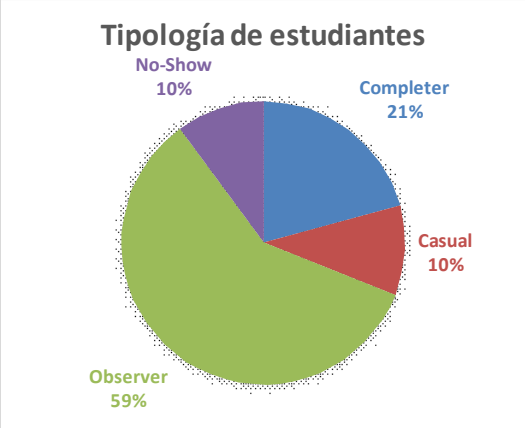

Figura 2: Distribución del tipo de participante.

participante tenía un guion con todas las acciones que debía hacer para implementar el juego. En cada parte, el guion iba decrementando el detalle de las acciones hasta llegar a la parte 3 donde el participante sólo tenía la propuesta de qué tenía que implementar sin ninguna explicación de cómo debía resolverlo.

- Diseño e implementación de un proyecto propio: En este caso, el participante diseñaba un juego o animación en Scratch para posteriormente implementarla con todos los conocimientos que adquiría en el curso. En este caso, la rúbrica de evaluación se basaba en funcionalidad e originalidad de la propuesta juntamente con la dificultad de la implementación.

6. Resultados

En esta sección describimos los resultados de participación del curso, el rendimiento de los participantes y la opinión de los mismos.

Antes de exponer los resultados es importante comentar las características del curso:

- El curso se realizó en una instancia de la plataforma Open edX instalada en la universidad. La ventaja de utilizar esta plataforma libre es que permite realizar un MOOC sin depender de las restricciones y condiciones que actualmente existen para impartir el curso en las plataformas estándares. La desventaja es que se pierde el potencial de publicidad y de público que tienen implícitas estas plataformas.
- La publicidad del curso fue mediante los canales propios de la universidad y los foros públicos existentes de publicidad de MOOC. Esta característica hizo que el número de participantes fuera de 1799. Un número reducido en comparación a los cursos de las plataformas MOOC estándares.

Si analizamos la participación observamos un alto índice de abandono. Es un hecho conocido en este

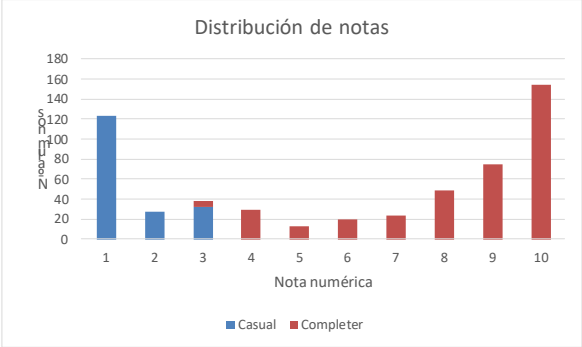

Figura 3: Distribución de notas por tipo de participante.

tipo de cursos. Normalmente, la media de participantes que completan un MOOC es aproximadamente del 10% [6]. Según Wilkowski [10], se puede clasificar el tipo de participante según su nivel de implicación en el MOOC. Existe el que sólo se matricula (*no-show*), el que observa la metodología y algún recurso docente (*observer*), el que realiza alguna actividad (*casual learner*) y el que completa el curso (*completer*). La Figura 2 refleja la distribución de los participantes matriculados en este MOOC según esta tipología. La distribución es una distribución clásica de MOOC donde existen muchos participantes matriculados, pero pocos que terminan el curso. En este caso podemos ver que sólo un 21% completan el curso, un 10% realizan algunas actividades (en este curso definimos *casual learner* como aquel participante que ha realizado menos de 3 actividades de evaluación sobre las 8 disponibles), un 59% miraron algún vídeo y el resto (un 10%) no hicieron ninguna acción.

Si observamos las notas finales de los participantes que realizaron alguna actividad (ver Figura 3), vemos una distribución razonable en función del esfuerzo realizado en el curso. Los participantes del tipo *completer* tienden a tener notas de aprobado o superiores. Además, la figura muestra que casi un 42% de los *completers* obtienen una nota superior a 9. Es decir, en el caso de seguir el curso, la mayoría completan casi todas las actividades de forma correcta. Pero ¿cuáles son los participantes que siguieron el curso y lo superaron? El Cuadro 2 disgrega las notas por nivel de estudios completados. Este nivel de estudios es reportado por el participante en el momento de matricularse y que en algunos casos no se reporta (marcado como *no especificado*). Observamos que la mayoría de participantes provienen de estudios universitarios o superiores o bien ciclos formativos de grado superior. El número de participantes que no siguen el curso en todos los casos se acerca al 70%, excepto aquellos con estudios superiores que es más

Nivel estudios	Total matriculados		No-show / observer		Suspendidos		Aprobados	
	Abs.	Porc.	Abs.	Porc.	Abs.	Porc.	Abs.	Porc.
Ninguno	21	1%	14	67%	4	57%	3	43%
Primaria	74	4%	49	66%	15	60%	10	40%
Secundaria	1063	59%	720	68%	137	40%	206	60%
Ciclo superior	235	13%	165	70%	23	33%	47	67%
Grado o equivalente	268	15%	191	71%	26	34%	51	66%
Máster o doctorado	57	3%	47	82%	6	60%	4	40%
No especificado	81	5%	56	69%	10	40%	15	60%
Total	**1799**		**1242**	**69%**	**221**	**40%**	**336**	**60%**

Cuadro 2: Rendimiento de los estudiantes por nivel de estudios.

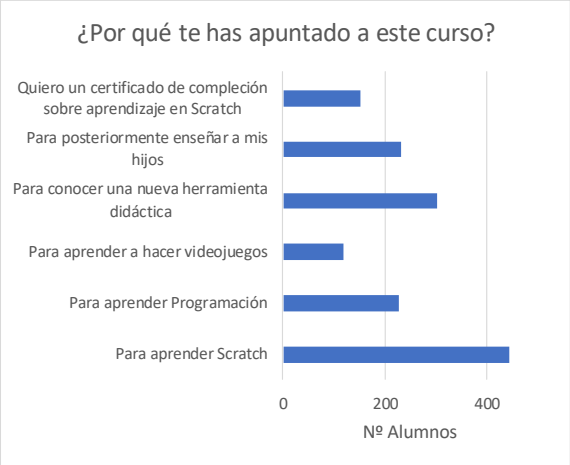

Figura 4: Razón de matricularse.

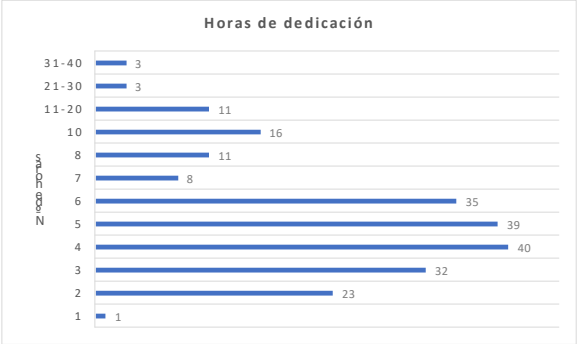

Figura 5: Horas de dedicación

del 80%. Probablemente este colectivo simplemente exploró el curso y sus contenidos. Respecto a la superación del curso, es interesante comprobar que en los colectivos de mayor presencia la ratio fue del 60%-67%, sensiblemente superior a la media del curso (60%). Finalmente, un resultado interesante es que hubo un 5% de participantes sin ningún nivel de estudios o primaria. Por ejemplo, tenemos constancia, por los mensajes de presentación dentro del foro, de amas de casa y jóvenes de secundaria matriculados con supervisión de los padres. En estos casos, podemos ver que los porcentajes se invierten (40% de aprobados) y la mayoría no superan el curso o abandonan.

Los participantes también recibieron una encuesta al inicio del curso para saber sus expectativas y opinión. Al inicio del curso, entre otras cosas, se les preguntó sobre la razón de inscribirse. La Figura 4 muestra las respuestas sobre una muestra de 645 respuestas que corresponde a un 36% de los matriculados. Nótese que era posible seleccionar más de una opción. Es interesante observar que un alto porcentaje (30%) indican que quieren aprender simplemente la herramienta Scratch. Otros quieren transmitir el conocimiento aprendido como profesores a distintos niveles educativos (21%) o como padres de familia (16%). Otro dato interesante es que algunos escogen el curso para introducirse a la programación (15%).

Al final del curso, se realizó otra encuesta valorando diferentes ítems respecto a su creencia sobre qué habían aprendido, las horas de dedicación y sus expectativas de futuro. Respondieron 215 participantes que corresponden a un 39% de los que obtuvieron nota final (*casual* y *completers*) y a un 58% de los que completaron el curso (*completers*). De éstos, un 88% manifestaron que después del curso se ven capaces de aprender un lenguaje estándar de programación tipo Java o C. Es un resultado muy positivo, teniendo en cuenta que muchos de estos participantes no han tenido nunca contacto con un lenguaje de programación.

Sobre las horas de dedicación, la Figura 5 muestra las horas por semana que los participantes indican que han invertido. La distribución es de 3 a 6 horas por semana, que son las horas definidas en la planificación del curso. Por lo tanto, el tiempo de dedicación diseñado fue el correcto.

Finalmente, se les preguntó sobre qué estarían dispuestos a aprender después de este curso. El Cuadro 2 muestra las respuestas más relevantes. Muchos de los participantes se decantan por el aprendizaje de la herramienta Scratch y todas sus vertientes de la enseñanza hacia primaria con Scratch Jr. (26%) o secundaria con Scratch avanzado (73%) o bien

¿Qué quieres aprender después de Scratch?	Abs.	Porc.
Scratch Jr.	58	26%
Scratch Avanzado	166	73%
Robótica con Scratch	150	66%
App Inventor	104	46%
Introducción a programación de software	74	33%
Introducción a programación web	86	38%

Cuadro 3: Preferencias de aprendizaje

combinado con la robótica (66%). Debemos tener en cuenta que hay vídeos explicativos en el curso sobre todas estas posibilidades y por lo tanto es razonable que los participantes busquen sobre las diferentes opciones descritas para seguir con su aprendizaje. Si nos focalizamos en otros contenidos, los participantes están dispuestos a aprender otros lenguajes visuales como App Inventor (46%) y hasta aprender programación de software (33%) o web (38%). Resumiendo, el curso abre las expectativas de aprendizaje de los participantes y les permite perder el miedo a aprender otros conceptos y contenidos relacionados con la programación.

7. Discusión

Una pregunta que emerge después de realizar este curso es ¿se podría aplicar esta metodología a la iniciación a la programación en la universidad? ¿sería efectiva? Consideramos que como curso introductorio antes de fundamentos de programación sería una posibilidad, ya que muchas veces lo que falla en los estudiantes de primer curso es practicar la capacidad de abstracción [5] y el hecho de intentar resolver problemas de forma metódica. El hecho de introducir el pensamiento computacional en primera instancia e intentar resolver problemas simples sin necesidad de introducir primero la algoritmia, podría mejorar los resultados de los estudiantes. Los resultados de opinión de la sección anterior avalan esta hipótesis. Después de realizar el curso, un 88% de los encuestados se sienten capaces de aprender un lenguaje de programación estándar. Además, después del curso, algunos alumnos reafirman que querían aprender un lenguaje de programación (33%) o web (38%).

Otra opción sería introducir este curso en secundaria. Existe una petición reiterada de los docentes universitarios de la modificación del plan de aprendizaje en secundaria [2] para incluir competencias relacionadas con la ciencia y la tecnología informática y en las comunidades autónomas como Madrid (Decreto del currículo de la Educación Secundaria Obligatoria, Comunidad de Madrid. Decreto 48/2015, de 14 de mayo[5]) y Cataluña (Decreto de la ordenación de la enseñanza secundaria obligatoria, Generalitat de Catalunya. DECRET 187/2015, de 25 de agosto[6]) se incluye la programación en el currículum de la asignatura de tecnología. Por eso, cursos como éstos podrían servir para introducir a estos colectivos a la programación. La valoración final del curso en una escala de Likert (1 a 5) fue de media un 4,29. Por lo tanto, podemos ver la gran aceptación de los participante que mayormente lo terminaron. En relación con la dificultad del mismo los participantes valoraron en la misma escala un 3,58. Es decir, aunque la valoración del curso fue muy positiva, los participantes consideraron que el curso no era nada fácil y tenía sus dificultades.

8. Conclusiones y trabajo futuro

En este artículo hemos presentado un MOOC de iniciación a la programación mediante el lenguaje de programación Scratch. Se diseñó un curso con el objetivo de acercar, mediante el pensamiento computacional combinado con la metodología *learning-by-doing,* la programación a una persona sin ningún conocimiento previo sobre esta disciplina.

Los resultados han sido muy satisfactorios en aquellos participantes que han terminado el curso y con un alto grado de satisfacción y motivación para seguir aprendiendo conceptos relacionados con la ciencia y la tecnología. Creemos que cursos introductorios como éste pueden ser un factor a tener en cuenta para promover el aprendizaje de conceptos de la tecnología informática tanto en jóvenes como en la sociedad en general.

Como trabajo futuro nos proponemos hacer un piloto en cursos de educación secundaria para explorar la efectividad del curso y la opinión de los estudiantes. Alternativamente, también queremos explorar el curso como introductorio a los estudiantes de programación de primer curso del grado de Informática.

Agradecimientos

Este trabajo ha sido parcialmente financiado por el Elearn Center de la Universitat Oberta de Catalunya en el programa de ayudas a la promoción y diseño de MOOC.

[5] http://www.bocm.es/boletin/CM_Orden_BOCM/2015/05/20/BOCM-20150520-1.PDF

[6] http://portaldogc.gencat.cat/utilsEADOP/PDF/6945/1441278.pdf.

Referencias

[1] Karen Brennan, y Mitchel Resnick, New Frameworks for Studying and Assessing the Development of Computational Thinking. *American Educational Research Association (AERA) conference*, 2012.

[2] Xavi Canaleta, Fermín Sánchez Carracedo, Inés Jacob, Ángel Velázquez y Merche Marques. Declaración AENUI-CODDII por la inclusión de asignaturas específicas de ciencia y tecnología informática en los estudios básicos de la enseñanza secundaria y bachillerato. *En Actas de las XX Jornadas de Enseñanza Universitaria de la Informática*, pp. 229-236, Oviedo, 2014

[3] Jorge García, José C. Riquelme, Mariano González, y Isabel Nepomuceno. Diez años innovando en la enseñanza de los fundamentos de la programación: resultados y conclusiones. *En Actas de las XVIII Jornadas de Enseñanza Universitaria de la Informática*, pp. 9-16, Ciudad Real, 2012.

[4] Shuchi Grover, y Pea Roy. Computational thinking in K–12: A review of the state of the field. *Educational Researcher* 42(1), 38-43, 2013

[5] Carme Lacave, Ana Isabel Molina y Juan Giralt. Identificando algunas causas del fracaso en el aprendizaje de la recursividad: análisis experimental en las asignaturas de programación. *En Actas de las XIX Jornadas de Enseñanza Universitaria de la Informática*. pp. 225-232, Castelló de la Plana, 2013.

[6] Daniel F. O. Onah, Jane Sinclair y Russell Boyatt. Dropout rates of massive open online courses: behavioural patterns. *Proceeding of EDULEARN'14,* pp. 5825-5834, 2014.

[7] Seymour Papert. Mindstorms: Computers, Children and Powerful Ideas. *NY: Basic Books*, 1980.

[8] Mitchel Resnick, John Maloney, Andrés Monroy-Hernández, Natalie Rusk, Evelyn Eastmond, Karen Brennan, Amon Millner, Eric Rosenbaum, Jay Silver, Brian Silverman, y Yasmin Kafai. Scratch: Programming for All. *Communications of the ACM*, 52(11), 60-67, 2009.

[9] Mercé Rullán, Elena Valderrama, Jean-Pierr Deschamps, Lluís Terés, David Bañeres y Joaquin Saiz. Hacia un nuevo modelo docente: curso blended de sistemas digitales. *VIII Congrés Internacional de Docència Universitària i Innovació (CIDUI)*, no.2, 2014.

[10] Julia Wilkowski, Amit Deutsch, y Daniel M. Russell. Student skill and goal achievement in the mapping with google MOOC. *Proceedings of the first ACM conference on Learning@ scale conference*. ACM, 2014.

[11] Jeannette Wing. Computational Thinking. *Communications of the ACM*, 49(3), 33-35, 2006.