José Alfredo Martínez Valdés
J. Ángel Velázquez Iturbide

Estudio del Uso de Scratch en una Asignatura Introductoria de la Programación

Número 2017-03

Serie de Informes Técnicos DLSI1-URJC
ISSN 1988-8074
Grupo Docente de Lenguajes y Sistemas Informáticos I
Universidad Rey Juan Carlos

Índice

Estudio del Uso de Scratch en una Asignatura Introductoria de la Programación

José Alfredo Martínez Valdés, J. Ángel Velázquez Iturbide

Grupo Docente de Lenguajes y Sistemas Informáticos I, Universidad Rey Juan Carlos,
C/ Tulipán s/n, 28933, Móstoles, Madrid
ja.martinezv@alumnos.urjc.es, angel.velazquez@urjc.es

Resumen. Este informe presenta un estudio sobre el uso de Scratch en una asignatura introductoria de programación. Es una asignatura de un grado en videojuegos, donde se dedican dos semanas a Scratch y luego se sigue con Java. Se analiza el perfil de los alumnos, su rendimiento según la herramienta Dr.Scratch, y la opinión de los alumnos sobre el uso de Dr.Scratch y de Scratch como lenguaje de programación previo a Java. Se incluyen los enunciados de prácticas usados, los métodos de análisis, y los resultados detallados y comentados.

Palabras clave: Introducción a la programación, pensamiento computacional, Scratch, Dr.Scratch, Java.

1 Introducción

En los últimos años se viene propugnando el uso de lenguajes de bloques, como Scratch [1], para el aprendizaje de la programación en niveles educativos preuniversitarios. El objetivo docente sería el desarrollo por los alumnos del "pensamiento computacional" [2].

En este informe analizamos el rendimiento de los alumnos en el uso de Scratch como primer lenguaje en una asignatura introductoria a la programación, así como su aceptación del mismo. La estructura del informe es la siguiente. El apartado 2 presenta el planteamiento de la investigación, incluyendo el contexto educativo, los objetivos de la investigación y los materiales utilizados para realizarla. Los apartados 3, 4 y 5 presentan, respectivamente, los resultados de abordar tres preguntas de investigación: perfil de los alumnos, evaluar el uso de Dr.Scratch y la opinión de los alumnos sobre el uso de Scratch como lenguaje previo a Java. Los apartados 6 y 7 contienen, respectivamente, un resumen de resultados y unas conclusiones. Finalmente, siete apéndices contienen el enunciado de la práctica de Scratch, así como los enunciados y las respuestas a tres cuestionarios.

2 Planteamiento de la Investigación

La asignatura “Programación visual”, de primer curso del Grado en Diseño y Desarrollo de Videojuegos de la Universidad Rey Juan Carlos, es una asignatura de introducción a la programación con ciertas peculiaridades propias de la titulación. Por lo que se había observado en cursos anteriores, los alumnos respondían a diversos perfiles, unos más técnicos, pero otros más “artísticos”. Para aumentar la motivación de los alumnos y bajar la pendiente de curva de aprendizaje de la programación entre los alumnos menos técnicos, se pensó en utilizar un lenguaje de programación visual, con bloques (Scratch), durante dos semanas y después pasar a un lenguaje de programación convencional, textual (Java), que se usaría durante el resto del cuatrimestre.

La asignatura se ha impartido de esta forma en los cursos 2014-15 y 2015-16, y se ha evaluado en el curso 2016-17. En este curso, se matricularon un total de 93 alumnos en la asignatura (77 del Grado Diseño y Desarrollo de Videojuegos más 16 alumnos del doble grado en Diseño y Desarrollo de Videojuegos e Ingeniería de Computadores).

2.1 Objetivos de la Investigación

La investigación tiene como objetivo general conocer el impacto y aceptación de este enfoque de la asignatura entre los alumnos. De forma más específica, podemos identificar tres objetivos parciales:

Obj1. Conocer el perfil de los alumnos del Grado de Videojuegos.
Obj2. Evaluar el efecto del uso de Scratch sobre:
   a. Rendimiento de los alumnos.
   b. Aceptación por los alumnos.
Obj3. Evaluar el uso de Scratch como lenguaje de programación previo al uso de Java sobre:
   a. Rendimiento de los alumnos.
   b. Aceptación por los alumnos.

2.2 Materiales Disponibles

Dado que la investigación se plantea sobre una asignatura ya organizada, la investigación buscó alcanzar los objetivos los aspectos antes identificados, sin cambiar la organización ni la didáctica de esta. Sin embargo, también se vio que podía obtenerse información adicional mediante encuestas a los alumnos.

En conjunto, se dispone del siguiente material (se explican con más detalle en los apartados siguientes):

- Encuesta realizada a principio del curso. Se pidió a los alumnos que la rellenaran para autocaracterizarse.
- Prácticas: la primera (pr1) realizada en Scratch, otras realizadas en Java (pr2: expresiones, pr3: condicionales, pr4: bucles, pr5: repaso).

- Resultados devueltos por DrScratch más valoración de los alumnos.
- Tests de conocimientos de Java (dos tests).
- Examen final.
- Encuesta realizada a final de curso sobre su satisfacción con algunas cuestiones de la asignatura.

El material disponible es voluminoso, por lo que de momento nos hemos centrado en las partes más directamente relacionadas con Scratch. Asimismo, se cambió el objetivo 2 a valorar la experiencia con Dr.Scratch. En resumen, la investigación se ha realizado con los siguientes materiales y métodos de análisis, agrupados por objetivos:

Obj1. Conocer el perfil de los alumnos del Grado de Videojuegos:
- Encuesta contestada por los alumnos a comienzo del curso.

Obj2. Evaluar el uso de Dr.Scratch:
  a. Rendimiento de los alumnos:
     - Resultados de evaluar con Dr. Scratch las entregas de la práctica 1.
  b. Aceptación por los alumnos:
     - Comentarios abiertos realizados en la práctica 1.

Obj3. Evaluar el uso de Scratch como lenguaje de programación previo al uso de Java:
  a. Aceptación por los alumnos:
     - Comentarios abiertos de los alumnos realizados en la encuesta realizada a final de curso.

3 Objetivo 1: Conocer el Perfil de los Alumnos

3.1 Materiales

Para conocer el perfil de sus alumnos, la profesora diseñó una encuesta formada por siete preguntas. Se desarrolló con GoogleDocs y su URL se publicó en el campus virtual de la universidad. La profesora pidió a los alumnos que las rellenasen fuera del horario de clase.

La encuesta contenía siete preguntas, tres sobre datos personales (nombre, DNI y sexo), otra sobre su perfil (técnico, artístico, ambos o indeciso) y tres para saber si tenían o no conocimientos de programación, qué conocimientos tenían y qué lenguajes conocían. El cuestionario se encuentra en el Apéndice B.

3.2 Resultados

Se recibieron respuestas de 87 alumnos entre el 9 y el 23 de septiembre de 2016. La Tabla I presenta los datos recogidos de la encuesta, clasificados por las diversas características personales.

Tabla 1. Caracterización de los alumnos de Programación Visual en el curso 2016-17 (N=87)

Factor	Valor	# alumnos	%
Sexo	Hombre	78	89,7
	Mujer	9	10,3
Perfil	Técnico	39	44,8
	Artístico	42	48,3
	Ambos	5	5,75
	Indeciso	1	1,15
Repetidor	Sí	15	17,2
	No	72	82,8
Conocimientos previos de programación	Sí	42	48,3
	No	45	51,7

Veamos con algo de detalle los lenguajes y los conocimientos de programación previos. A priori, el factor más determinante para un alumno sepa algo de programación puede ser que sea repetidor.

De los 42 los alumnos que manifestaron tener conocimientos previos de programación, 3 afirmaron no conocer ningún lenguaje (*sic*). Por tanto, 39 alumnos conocían al menos un lenguaje, siendo 25 no repetidores y 14 repetidores (es decir, 34’72% de no repetidores y 93’33% de repetidores –todos menos uno–). Los conocimientos globales de lenguajes de programación aparecen en la Tabla 2.

Tabla 2. Lenguajes de programación conocidos por los alumnos al iniciar el curso (N=39)

Lenguaje	# alumnos
Java	24
C	14
Fortran	3
JavaScript	3
Python	2
Ruby	2
Arduino	1
GameMaker	1
HTML	1
Pascal	1
PHP	1
RGSS	1
Scratch	1
Unity	1
Visual Basic	1
VHDL	1

Veamos un desglose de lenguajes entre alumnos repetidores y no repetidores. Las Tablas 3 y 4 muestran, respectivamente, el número de lenguajes conocidos por los alumnos no repetidores y los lenguajes que conocían.

Tabla 3. Número de lenguajes de programación conocidos por los alumnos no repetidores (N=72)

Número de lenguajes conocidos	# alumnos	%
0	47	62,67
1	16	21,33
2	6	8,00
3	2	2,67
4	1	1,33

Tabla 4. Lenguajes de programación conocidos por los alumnos no repetidores (N=25)

Lenguaje	# alumnos
Java	11
C	12
Fortran	3
JavaScript	2
Python	1
Ruby	2
Arduino	0
GameMaker	1
HTML	1
Pascal	1
PHP	0
RGSS	1
Scratch	1
Unity	0
Visual Basic	1
VHDL	1

Asimismo, las Tablas 5 y 6 muestran, respectivamente, el número de lenguajes conocidos por los alumnos repetidores y los lenguajes que conocían.

Tabla 5. Número de lenguajes de programación conocidos por los alumnos repetidores (N=15)

Número de lenguajes conocidos	# alumnos	%
0	1	6,67
1	12	80
2	0	0
3	0	0
4	2	13,33

Tabla 6. Lenguajes de programación conocidos por los alumnos repetidores (N=14)

Lenguaje	# alumnos
Java	13
C	2
Fortran	0
JavaScript	1
Python	1
Ruby	0
Arduino	1
GameMaker	0
HTML	0
Pascal	0
PHP	1
RGSS	0
Scratch	0
Unity	1
Visual Basic	0
VHDL	0

Por último, se les preguntó por sus conocimientos de programación. Las Tablas 7 y 8 muestran los resultados para los alumnos no repetidores y repetidores que decían conocer algún lenguaje de programación, respectivamente.

Tabla 7. Conocimientos previos de programación de alumnos no repetidores (N=25)

Conocimiento de programación	# alumnos	%
Variables	23	92
Entrada y salida de valores	23	92
Sentencias de selección	23	92
Sentencias iterativas	19	76
Funciones y métodos	12	48
Subprogramación	12	48
Recursividad	7	28
Arrays	8	32
Algoritmos de ordenación y búsqueda	8	32
Ficheros	8	32

Tabla 8. Conocimientos previos de programación de alumnos repetidores (N=14)

Conocimiento de programación	# alumnos	%
Variables	13	92,86
Entrada y salida de valores	13	92,86
Sentencias de selección	13	92,86
Sentencias iterativas	13	92,86
Funciones y métodos	8	57,14
Subprogramación	8	57,14
Recursividad	7	50
Arrays	8	57,14
Algoritmos de ordenación y búsqueda	8	57,14
Ficheros	2	14,29

4 Objetivo 2: Evaluar el Uso de Dr.Scratch

4.1 Procedimiento

En la primera semana de clase, la profesora informó a los alumnos de Scratch, incluyendo la manera de registrarse en la plataforma de Scratch, cómo compartir proyectos, cómo acceder a otros proyectos, como explorar el código de cada proyecto y cómo descargarlos a su ordenador. También les enseñó a programar en Scratch, así como a crear y publicar sus propios proyectos, usando 11 proyectos de ejemplo.

Al finalizar las dos primeras semanas de clases, se propuso a los alumnos una práctica sobre Scratch (véase Apéndice A). Los alumnos debían desarrollar un juego con, al menos, los elementos estudiados en clase: mensajes de usuario, sonidos y movimientos, sentencias iterativas, sentencias repetitivas, eventos (p.ej. “al empezar” o “al pulsar una tecla”), sensores, operadores, variables y el lápiz. Los alumnos debían enviar a la profesora una carpeta comprimida con el código fuente de Scratch. Cada alumno también debía evaluar su práctica con la herramienta Dr.Scratch y después rellenar un formulario con los resultados devueltos por Dr.Scratch y su valoración de dicha evaluación.

Dr.Scratch es una aplicación [11] desarrollada para evaluar proyectos de Scratch. Puede observarse un resultado de evaluar un proyecto en Fig. 1. Puede observarse que la interfaz consta de varias zonas. En la zona superior izquierda se muestra la puntuación total (entre 0 y 21) y su nivel asociado (“alto”, “medio” o “bajo”). En la zona media izquierda (“mejores prácticas”) muestra un resumen de los errores detectados. En la zona inferior izquierda ofrece la oportunidad de descargar un certificado de la evaluación del proyecto. En el lado derecho de la pantalla se encuentre el apartado “mejora tu nivel”, que muestra un desglose de la puntuación en 7 elementos del pensamiento computacional: paralelismo, pensamiento lógico, control de flujo, interactividad, representación de la información, abstracción y sincronización. Cada elemento del pensamiento computacional recibe una puntuación entera comprendida entre 0 y 3.

Fig. 1. Ejemplo de evaluación de un proyecto Scratch por Dr.Scratch.

Los alumnos debían rellenar un formulario donde, aparte de identificarse, debían copiar la información recibida por Dr.Scratch en los apartados de puntuación, mejores prácticas y mejora tu nivel. Finalmente, se les pedía su opinión sobre la evaluación dada a su proyecto y, por otra parte, permitió que ellos registraran el puntaje total de sus proyectos. El formulario completo se encuentra en el Apéndice B y las respuestas de los alumnos, en el Apéndice C. No se analiza en el informe el apartado de mejores prácticas, porque es dependiente de cada proyecto Scratch.

4.2 Rendimiento de los Alumnos

Un total de 89 alumnos realizaron la entrega de la práctica de Scratch. La puntuación asignada por Dr.Scratch varió entre 10 y 20 puntos (véase la Tabla 9).

Tabla 9. Puntuación dada por Scratch a los proyectos de Scratch (N=89)

Puntuación	# alumnos	%
20	2	2,25
19	1	1,12
17	8	8,99
16	8	8,99
15	10	11,24
14	18	20,22
13	15	16,85
12	14	15,73
11	7	7,87
10	6	6,74

También es interesante desglosar los resultados en las 7 dimensiones que establece Dr.Scratch (véanse las Tablas 10 y 11).

Tabla 10. Calificaciones de Dr.Scratch desglosadas en niveles (N=89)

Dimensiones	**Nivel alcanzado**							
	0		**1**		**2**		**3**	
	#	**%**	**#**	**%**	**#**	**%**	**#**	**%**
Paralelismo	0	0	29	32,58	15	16,85	45	50,56
Pensamiento lógico	2	2,25	41	46,07	10	11,24	36	40,45
Control de flujo	0	0	0	0	64	71,91	25	28,09
Interactividad	0	0	0	0	88	98,88	1	1,12
Representación de la información	0	0	1	1,12	86	96,03	2	2,25
Abstracción	0	0	81	91,01	1	1,12	7	7,87
Sincronización	0	0	16	17,98	38	42,70	35	39,33

Tabla 11. Calificaciones de Dr.Scratch desglosadas por dimensiones (N=89, IC=95%)

Estadístico	**Dimensiones del pensamiento computacional***							**Total**
	PA	**PL**	**CF**	**IN**	**RI**	**AB**	**SN**	
Media	2,18	1,90	2,28	2,01	2,01	1,17	2,21	13,76
Desviación estándar	0,90	0,98	0,45	0,11	0,18	0,55	0,73	2,23
Mediana	3	2	2	2	2	1	2	14
Moda	3	1	2	2	2	1	2	14
Mínimo	1	0	2	2	1	1	1	10
Máximo	3	3	3	3	3	3	3	20

[a]PA: paralelismo, PL: pensamiento lógico, CF: control de flujo, IN: interactividad, RI: representación de la información, AB: abstracción, SN: sincronización

Por último, los alumnos debían copiar el mensaje que Dr.Scratch les daba en el apartado "mejores prácticas", donde les indicaba qué aspectos debían mejorar. La Tabla 12 recoge esta información.

Tabla 12. Resumen del informe presentado por Dr.Scratch en el apartado "mejores prácticas" (N=89)

Mensaje	**# alumnos**	**%**
Atributos no inicializados correctamente	87	97,75
Nombres inadecuados	47	52,81
Programas duplicados	25	28,09
Código muerto	7	7,87

Otro aspecto interesante de los resultados sería si hay alguna correlación entre las calificaciones y los factores que caracterizan la población. Desglosadas las calificaciones por factores, obtenemos los datos de la Tabla 13. En estos datos se han descartado las calificaciones de 5 alumnos que no rellenaron la encuesta inicial.

Tabla 13. Puntuación dada por Dr.Scratch a los proyectos de Scratch desglosadas por factores (N=84)

Factor	Valor	# alumnos	%	Media	DT	Mediana	IC 95%
Sexo	Hombre	75	89,29	13,75	2,25	14	13,23 - 14,27
	Mujer	9	10,71	13	1,73	13	11,67 - 14,33
Perfil	Técnico	37	44,05	13,97	2,62	14	13,10 - 14,85
	Artístico	42	50	13,38	1,75	13	12,83 - 13,93
	Ambos	4	4,76	13	2	12	09,82 - 16,18
	Indeciso	1	1,19	17	0	17	–
Repetidor	Sí	14	16.67	13.93	2,921	13	12,24 - 15,62
	No	70	83,33	13,61	2,059	14	13,12 - 14,11
Conocimientos previos de programación	Sí	42	50	13,76	2,43	13	13,01 – 14,52
	No	42	50	13,57	1,99	14	12,95 – 14,19

Haciendo análisis de varianza, no se encontraron diferencias significativas en ningún factor.

4.3 Aceptación de los Alumnos

La última pregunta del cuestionario formulaba una pregunta sobre Dr.Scratch "¿Crees que te ha ayudado a saber qué aspectos tienes que mejorar más y cómo hacerlo?".

La Tabla 14 muestra el resultado de clasificar las contestaciones de los alumnos en cuatro categorías básicas (respuesta en blanco, opinión positiva, opinión negativa y mixta). La categoría mixta recoge aquellas respuestas que afirman que Dr.Scratch les ha parecido útil, pero con reparos.

Tabla 14. Clase de respuesta de los alumnos a la utilidad de Dr.Scratch (N=89)

Clase de respuesta	# alumnos	%
En blanco	39	43,82
Sí	27	30,34
Mixta	9	10,11
No	14	15,73

Resulta interesante conocer en qué consideran los alumnos que Dr.Scratch ha sido útil o en qué no. La Figura 15 contiene el resultado de analizar las respuestas afirmativas sobre la utilidad de Dr.Scratch.

Tabla 15. Clase de respuesta positiva de los alumnos a la utilidad de Dr.Scratch (N=22)

Clase de respuesta	# alumnos
Para programar mejor	11
Para mejorar el uso de algún elemento de Scratch	8
Para conocer mejor Scratch	3

Veamos con un poco más de detalle estas clases de respuesta. En cada categoría incluimos algunas citas textuales de las respuestas de los alumnos. Se utiliza la notación A*nn* para indicar que dicha respuesta fue dada por el alumno número *nn*; las completas se encuentran en el Apéndice B.

- Para programar mejor. Son afirmaciones genéricas de que, de alguna forma, les ha ayudado a programar mejor. Por ejemplo: "Sí, me ha ayudado a fijarme en aspectos que había pasado por alto" (A43), "En algunos aspectos sí he entendido que debo mejorar (…)" (A59). A veces, las respuestas son constructivas, pero sin aclarar cómo: "Sí, se me ocurren muchas soluciones a los errores" (A33), "Sí, en que se puede simplificar más el código" (A82). Otras veces, son respuestas algo crípticas, pero siempre orientadas a la mejora: "Probablemente tenga que mejorar la originalidad" (A25).
- Para mejorar el uso de algún elemento de Scratch. Se trata de respuestas específicas, que comentan que Scratch les ha ayudado a mejorar los nombres de variables u objetos (3 respuestas), la inicialización de variables (2 respuestas), la interactividad con el usuario, la abstracción o el movimiento de los objetos (1 respuesta cada una). En algún caso, es difícil imaginar cómo se ha concretado dicha ayuda: "El movimiento de los objetos para que sea más fluido y sin errores" (A66).
- Para conocer mejor Scratch. Por ejemplo: "Sí, me ha informado de algunas funciones que no conocía" (A87).

Por último, encontramos críticas a la evaluación realizada por Dr.Scratch. Como puede verse en la Tabla 16, son de cuatro clases.

Tabla 16. Clase de respuesta negativa de los alumnos a la utilidad de Dr.Scratch (N=18)

Clase de respuesta	# alumnos
El alumno discrepa con el criterio de Dr.Scratch	8
Dr.Scratch no valora la calidad global del programa	5
Funcionalidad de Dr.Scratch	4
DrScratch no orienta para programar mejor	1

Veamos con un poco más de detalle estas clases de respuesta:

- El alumno discrepa con el criterio usado por Dr.Scratch. Podemos separar estas críticas en dos grupos:
  - Utiliza un criterio demasiado rígido (1 respuesta). En concreto, no están de acuerdo con tener que incluir elementos solamente para obtener mejor puntuación, cuando son innecesarios para el proyecto desarrollado.
  - No están de acuerdo con la obligación de tener que usar identificadores para variables u objetos que sean distintos de los dados por defecto (3 respuestas). Hay respuestas en las que se comprueba una falta de conciencia por la legibilidad de los programas: "aunque algunos errores (como el nombre del "objeto 1", que al ser una barra que hace que el rascacielos vuelva a su posición inicial no necesita un nombre específico)

son, a mi entender, irrelevantes" (A71). Sin embargo, en otros casos, el alumno podría tener algo de razón: "Hay más de treinta objetos y es casi imposible ponerles nombres significativos, puesto que todos cumplen una función parecida" (A69).
    - No están de acuerdo con el criterio de Dr.Scratch sobre inicialización de objetos, código muerto, código duplicado o simplemente de forma genérica (4 respuestas). De nuevo, algunas respuestas denotan una falta de conciencia sobre la calidad de los programas pero otras parecen estar meditadas.
- Dr.Scratch no valora la calidad global del programa. Aunque reconocen que la evaluación independiente de cada dimensión es útil, consideran que también debería evaluarse el proyecto en su conjunto. A su vez, pueden separarse en dos clases:
    - Debería valorar el comportamiento del programa (5 respuestas), al menos cuando éste es un juego. Por ejemplo: "(…) Me habría gustado que se tuviese en cuenta el aspecto final o estética o resultado final del juego. Por ejemplo, mi juego no es que sea algo demasiado complicado pero dediqué una gran parte del tiempo en crear todas las animaciones y en buscar diferentes efectos de sonido que estuviesen relacionados con la atmósfera que tiene el juego en sí. (…)" (A84).
    - No ajusta su valoración a cada programa (2 respuestas). Estas respuestas critican que Dr.Scratch valore cada proyecto sin tener en cuenta sus características ni que el enunciado de la práctica forzaba a usar ciertas características, a veces difíciles de incluir en un proyecto libre.
- Funcionalidad de Dr.Scratch. Hay dos clases de problemas:
    - El alumno no entiende los mensajes de Dr.Scratch (3 respuestas). Por ejemplo: "No se explican bien los elementos a mejorar, por lo que realmente no se entiende bien la evaluación" (A26).
    - Dr.Scratch no evalúa bien según sus propios criterios (1 respuesta): "(…) aunque creo que algún apartado no está bien evaluado, puesto que las limitaciones del propio programa hacen que la evaluación falle en algunos aspectos" (A77).
- DrScratch no orienta para programar mejor. Echa en falta que los comentarios sean más constructivos: "Veo los aspectos en los que tengo que mejorar pero no la forma en este proyecto tan sencillo" (A52).

5 Objetivo 3: Evaluar la Aceptación del Uso de Scratch para Java

Incluimos en esta sección los resultados de dos cuestionarios a fin de curso.

5.1 Cuestionario en el Primer Examen Final

A finales del curso, se proporcionó a los alumnos una encuesta de opinión su experiencia en la asignatura y, en concreto, sobre el uso de Scratch como lenguaje de

programación previo a Java (véase Apéndice D). Aparte de sus datos personales, la encuesta constaba de ocho preguntas:

- Cuatro preguntas de múltiple respuesta, que les pedía su opinión sobre si el uso inicial de Scratch les habría ayudado en el aprendizaje de Java. En la primera se les preguntaba por Java en general, mientras que las tres siguientes preguntaban por tres elementos concretos (expresiones, instrucciones condicionales e instrucciones iterativas).
- Cuatro preguntas abiertas sobre la asignatura, que les pedía su opinión sobre si alguna parte de la asignatura, sobraba, faltaba, o convendría dedicarle más o menos tiempo.

Se recogieron 76 respuestas. Dos respuestas correspondían al mismo alumno pero realizadas con 27 días de separación, por lo que sus respuestas varían bastante. Dado que ambas respuestas tienen interés (reflejan la opinión del alumno en momentos distintos de la asignatura), las mantenemos.

Pueden encontrarse todas las respuestas en el Apéndice E, ordenadas por orden de entrega. Hemos etiquetado la respuesta de cada alumno con la notación A*nn*, como en la encuesta de Dr. Scratch. Sin embargo, las respuestas de ambas encuestas no están ordenadas igual, es decir A*nn* corresponde a alumnos distintos en ambas encuestas. De hecho, las dos respuestas del mismo alumno en esta encuesta están numeradas A09 y A56.

Las preguntas de múltiple respuesta no estaban diseñadas conforme a una escala de Likert, por lo que no las analizamos.

Veamos, el análisis cualitativo de las cuatro preguntas abiertas. Para cada pregunta, damos unos datos globales de las respuestas recibidas pero solamente analizamos las que explícitamente citan a Scratch. Además, en las cuatro preguntas se mezclan las respuestas sobre Scratch, sobre todo las que opinan que debe quitarse o dedicarse menos tiempo a esta parte. Para dar mayor coherencia a estas respuestas, analizamos simultáneamente las cuatro preguntas, de forma que estas respuestas solamente se anoten una vez y de forma coherente para cada alumno. Además, recogemos algunas opiniones positivas sobre la enseñanza de Scratch.

Podemos clasificar las respuestas sobre Scratch en tres categorías: suprimirlo, dedicarle menos tiempo o sin comentarios sobre Scratch. La Tabla 17 muestra los resultados.

Tabla 17. Respuestas de los alumnos sobre la supresión de Scratch (N=76)

Respuesta	**# alumnos**	**%**
Quitar	17	22,7
Reducir	24	32,4
Sin comentario	35	44’9

Algunos alumnos que proponen reducir el tiempo dedicado a Scratch, no indican nada más salvo, en todo caso, a cuántos días lo reducirían (llegando a un día). Sin embargo, otros alumnos señalan dónde quitarían tiempo a Scratch:

- Cambiar el contenido de las clases de Scratch (2 respuestas): “Algunas de las clases dedicadas a Scratch fueron, en mi opinión, innecesarias y un poco infantiles” (A04), “Y hacer tantas prácticas en clase de Scratch resulta cansado y repetitivo cuando al final son casi iguales, hacer una criba para ver las más difíciles pero asequibles y centrarse en aquellas que tengan que ver con Java (pues hay cosas que dimos en Scratch que luego en Java no se pueden hacer, como el cambiar de disfraz, y no veo el sentido a explicar eso), le quitaría tiempo a esa parte.” (A16).
- La práctica (2 respuestas). Abogan por suprimir la práctica o no hacer obligatoria su entrega.

Siete respuestas revelan aspectos positivos en el uso de Scratch, al menos de forma potencial:

- Útil potencialmente como lenguaje previo a Java, pero que no se estudia explícitamente (5 respuestas): “Yo ya conocía Java anteriormente, por eso he contestado «sin opinión» a las preguntas sobre Scratch, pero al explicar bucles y otras cosas quizás habría venido bien precisamente que se comparen con Scratch así los conocimientos de Scratch hubieran ayudado más a aprender Java.” (A08), “(…) como concepto a la hora de introducir a la asignatura quizás funciona pero al llegar a Java no se nota lo suficiente para el tiempo dedicado a este” (A18), “Una transición de Scratch a Java más lenta, porque al no conocer el lenguaje de programación, sigue chocando mucho” (A36), “(…) Aunque ayude para entender mejor Java podría darse más rápido.” (A43), “En mi opinión, Scratch debería ser usado como herramienta para entender bucles y condiciones, haciendo menos prácticas y aprovechando mejor el tiempo” (A52).
- Útil como introducción a la programación (2 respuestas): “Sin embargo y a pesar de poseer ya conocimientos sobre él, me ayudó bastante a comprender cómo funciona y de qué trata la programación a nivel general” (A06), “Scratch, aunque fue de ayuda (…)” (A27).

5.2 Cuestionario en el Segundo Examen Final

Dado que no esperábamos un rechazo tan general de Scratch, se quiso tener una confirmación del mismo y sus razones. El análisis anterior se realizó una vez acabadas las clases, por lo que es de suponer que los alumnos que contestaron fueron alumno suspensos que se presentaron en la convocatoria de junio. Se les animó a contestar a un cuestionario en el que se les explicaba que la mayor parte de ellos habían abogado por suprimir o reducir el tiempo dedicado a Scratch. Por tanto, se les preguntaba si estaban de acuerdo en suprimirlo y por qué. El cuestionario y las respuestas se encuentran en los Apéndices F y G.

La encuesta fue respondida por 26 alumnos. Los resultados desglosados se muestran en la Tabla 18.

Tabla 18. Respuestas de los alumnos sobre la supresión de Scratch (N=26)

Respuesta	# alumnos	%	# alumnos con explicación
Sí	15	57,7	9
No	11	42,3	11

Veamos con detalle las razones aducidas tanto en un sentido como en el otro. Si comenzamos por las respuestas afirmativas, encontramos que los 9 alumnos aportaron 14 "razones simples". La Tabla 19 presenta su clasificación en tres clases de respuestas.

Tabla 19. Razones de los alumnos a favor de suprimir Scratch (N=14)

Razón	# respuestas
No ayuda a aprender a programar en Java	7
Consume tiempo que podría dedicarse a otros temas	5
Se estudia en educación secundaria	2

Veamos con un poco más de detalle estas razones:

- Scratch no ayuda a programar en Java. Coincide con la pregunta que se les formulaba. Algunos alumnos eran explícitos, mientras que otros hablaron de forma más escueta: "No aprendí gran cosa con ello" (A15). Pueden destacarse las respuestas que expresan desacuerdo con la forma de enfocar la docencia ("Hay maneras mejores de explicar los conceptos que se intentan explicar", A24) o que tienen una concepción jerárquica de los lenguajes de programación ("No resulta útil para aprender la programación más «seria»", A03).
- Consume tiempo que podría dedicarse a otros temas. El tiempo ahorrado podría dedicarse a Java. Algunos alumnos comentan aspectos concretos donde les hubiera gustado disponer de más tiempo (ficheros, resolver problemas en clase).
- Se estudia en educación secundaria.

En cuanto a las respuestas negativas a suprimir Scratch, encontramos que los 11 alumnos aportaron 19 "razones simples". La Tabla 20 presenta su clasificación en tres clases de respuestas.

Tabla 20. Razones de los alumnos contra suprimir Scratch (N=19)

Razón	# respuestas
Consume tiempo que podría dedicarse a otros temas	9
Ayuda a aprender a programar	8
Es entretenido	2

Veamos con un poco más de detalle estas razones. Normalmente, las razones primera y segunda aparecen unidas (8 de los 11 alumnos):

- Consume tiempo que podría dedicarse a otros temas. En realidad, no es un argumento contra la supresión de Scratch sino una confirmación de que

opinan que se dedica excesivo tiempo a Scratch. Tres alumnos lo justifican para mejorar el aprendizaje de Java: "… pero sí es cierto que con dedicarle 2 o como mucho 3 clases y una pequeña práctica bastaría y así emplear algo más de tiempo en explicar la parte de ficheros ya que es la más complicada" (A16). Dos alumnos abogan por reducirlo a dos clases y, curiosamente, dos alumnos proponen reducirlo al tiempo que se le dedica actualmente (A16 y A20). Un alumno avisa del peligro de que el número de clases actuales aburran: "Pero reducir el tiempo dedicado podría ser bueno ya que puede llegar a ser pesado" (A19).

- Ayuda a aprender a programar. Hay cierta variedad de opiniones sobre dicha ayuda. Cinco alumnos piensan que es útil como introducción a la programación: "Empezar la asignatura con Scratch aclara conceptos para continuar después en Java, por lo que suprimirlo sería innecesario" (A19). Otros tres alumnos piensan que es útil para alumnos que no saben programar: "Creo que para los quejuegos (*sic*) llegan nuevos sin saber nada de programación como en mi caso, es útil para entender cómo funciona la programación (…)" (A13).
- Es entretenido. Para dos alumnos se trata de una actividad "entretenida" o relajante: "Otorga algo de libertad creativa y visión de diseño del videojuego entre tanto desarrollo que llega a cansar de la carrera" (A22).

6 Resumen de Resultados

Podemos destacar las siguientes características de los alumnos matriculados en la asignatura:

- Casi un 90% son hombres.
- Hay equilibrio entre alumnos con perfil técnico y artístico.
- Bastantes alumnos tienen conocimientos de programación. Aparte de los repetidores (un sexto aproximadamente de los matriculados), una tercera parte de los alumnos nuevos conocen algún lenguaje.
- Hay un número destacado de alumnos que conocen Java. Los alumnos que conocen C son de nuevo acceso; convendría indagar de dónde proviene este conocimiento.

Veamos los resultados de puntuar las prácticas de Scratch con Dr.Scratch:

- La puntuación media es 13'76 sobre 21.
- Seis de las siete dimensiones obtienen puntuaciones medias situadas alrededor de 2, salvo "abstracción" (1'17). "Pensamiento lógico" es la segunda dimensión con menor puntuación y la única con puntuaciones de 0. Por el contrario, "paralelismo" es la dimensión con mayor mediana y moda (3), aunque no con mayor media.
- Las malas prácticas identificadas por Dr.Scratch son muy limitadas, al ser sólo de cuatro clases. Principalmente aluden a la no inicialización de atributos, seguido de nombres inadecuados y programas duplicados.

En cuanto a la valoración de los comentarios de Dr.Scratch, encontramos lo siguiente:

- Casi la mitad de los alumnos no opinan.
- Hay opiniones favorables, desfavorables y mixtas, con cierto predominio de las favorables sobre las no favorables (40’45% frente a 25’84%).
- Las opiniones positivas se deben a haberles ayudado a mejorar a programar, a mejorar en el uso de algún elemento concreto de Scratch o a un mejor conocimiento del lenguaje.
- Las opiniones negativas critican principalmente que no se haga una valoración global del programa (sobre todo de su comportamiento) y reflejan discrepancias con las valoraciones recibidas. También indican que los comentarios son poco explicativos y poco constructivos.

Al final de la asignatura, la valoración de la experiencia con Scratch puede resumirse en lo siguiente:

- Casi la mitad de los alumnos no opinan.
- Más de la mitad de los alumnos abogan por suprimir o reducir el tiempo dedicado a Scratch.
- Aunque algunos alumnos valoran positivamente el uso de Scratch como antecedente de Java, echan en falta una atención explícita a la relación y la transición entre ambos lenguajes.

Preguntados explícitamente por la supresión de Scratch, las opiniones pueden resumirse así:

- Existen una opinión bastante general de que se dedica demasiado tiempo a Scratch y menos a otros aspectos de Java.
- Las opiniones se reparten entre los que abogan por su supresión o simplemente por una reducción del tiempo dedicado.
- Muchos de los que proponen suprimir Scratch piensan que su aprendizaje no ayuda a aprender a programar en Java, mientras que muchos de los segundos, que ayuda a los que no tienen conocimientos previos.

7 Conclusiones

Hemos presentado un estudio sobre el uso de Scratch en una asignatura introductoria de programación. Se ha analizado el perfil de los alumnos, su rendimiento según la herramienta Dr.Scratch, y la opinión de los alumnos sobre el uso de Dr.Scratch y de Scratch como lenguaje de programación previo a Java. Los resultados han sido claramente peores de lo esperado, pero arrojan luz sobre algunas de las razones para dicho rechazo.

Agradecimientos. Agradecemos a Raquel Hijón su colaboración en la preparación, acceso y distribución de materiales y respuestas de los alumnos. Este trabajo se ha

financiado con los proyectos TIN2015-66731-C2-1 del Ministerio de Economía y Competitividad de España y S2013/ICE-2715 de la Comunidad Autónoma de Madrid.

Referencias

1. Resnick, M., Maloney, J., Monroy-Hernández, A., Rusk, N., Eastmond, E., Brennan, K., Millner, A., Rosenbaum, E., Silver, J., Silverman, B., Kafai, Y.: Scratch: Programming for all. Communications of the ACM 52, 11 (2009) 60-67, DOI 0.1145/1592761.1592779
2. Wing, J.: Computational thinking. Communications of the ACM 49, 3 (2006) 33-35
3. Moreno León, J., Robles, G., Román González, M.: Dr. Scratch: análisis automático de proyectos Scratch para evaluar y fomentar el pensamiento computacional. RED – Revista de Educación a Distancia 46, 10 (2015) DOI 10.6018/red/46/10

Apéndice A: Práctica 1

Programación Visual

PRÁCTICA 0

Realiza un juego en Scratch que incluya (de los 11 juegos que hemos hecho en clase):

- Mensajes al usuario
- Sonidos y movimientos
- Sentencias de control iterativas y repetitivas
- Uso de distintos eventos (al empezar, al pulsar una tecla, …)
- Sensores
- Operadores
- Variables
- Usa el lápiz para incorporar algún elemento
- … Mucha imaginación ☺

Apéndice B: Encuesta de Evaluación del Proyecto de Scratch con Dr.Scratch

Tu práctica de Scratch por Dr. Scratch

Sube a Dr.Scratch el archivo que has entregado por el Campus Virtual y rellena los resultados

2. Nombre completo

Escríbelo en mayúsculas y este orden: APELLIDO, APELLIDO, NOMBRE

_____

3. DNI con letra

Escríbelo con letra y sin guion, así: 11111111X

_____

PUNTUACIÓN

Lo que te ha salido en la puntuación, por ejemplo 14/21

_____

MEJORES PRÁCTICAS

Qué te ha puesto como mejores prácticas. Escribe el resumen en rojo, no la descripción detallada de obtienes al pulsar

_____

MEJORA TU NIVEL – PARALELISMO

Lo que te ha salido en el nivel, por ejemplo 1/3

- o 1/3
- o 2/3
- o 3/3
- o Otra: _____

MEJORA TU NIVEL – PENSAMIENTO LÓGICO

Lo que te ha salido en el nivel, por ejemplo 1/3

- o 1/3
- o 2/3
- o 3/3
- o Otra: _____

MEJORA TU NIVEL – CONTROL DE FLUJO

Lo que te ha salido en el nivel, por ejemplo 1/3

- o 1/3

- 2/3
- 3/3
- Otra: _____

MEJORA TU NIVEL – INTERACTIVIDAD CON EL USUARIO

Lo que te ha salido en el nivel, por ejemplo 1/3

- 1/3
- 2/3
- 3/3
- Otra: _____

MEJORA TU NIVEL – REPRESENTACIÓN DE LA INFORMACIÓN

Lo que te ha salido en el nivel, por ejemplo 1/3

- 1/3
- 2/3
- 3/3
- Otra: _____

MEJORA TU NIVEL – ABSTRACCIÓN

Lo que te ha salido en el nivel, por ejemplo 1/3

- 1/3
- 2/3
- 3/3
- Otra: _____

MEJORA TU NIVEL – SINCRONIZACIÓN

Lo que te ha salido en el nivel, por ejemplo 1/3

- 1/3
- 2/3
- 3/3
- Otra: _____

COMENTARIOS SOBRE LA APLICACIÓN DR.SCRATCH

¿Crees que te ha ayudado a saber qué aspectos tienes que mejorar más y cómo hacerlo?

_____

Apéndice C: Resultados de la Encuesta sobre Scratch y Dr.Scratch

En este apéndice incluimos las respuestas dadas por los alumnos al cuestionario sobre Dr.Scratch. Hay dos respuestas (A23 y A59) que incluyen un texto adicional entre paréntesis. Por error, estos alumnos dieron como respuesta dicho texto encerrado en paréntesis. Hemos buscado sus prácticas, las hemos sometido a Dr.Scratch de nuevo y hemos reproducido a continuación las "mejores prácticas" recomendadas por Dr.Scratch. En el caso del alumno A23, las puntuaciones recibidas en las dimensiones PL y RI estaban intercambiadas con las proporcionadas por el alumno, por lo cual se anotaron las nuevas puntuaciones.

Alumno	**Mejores prácticas**	**Puntuación**							**Comentarios sobre Dr. Scratch (¿crees que te ha ayudado a saber qué aspectos tienes que mejorar más y cómo hacerlo?)**
		PA	**PL**	**CF**	**IT**	**RI**	**AB**	**SN**	
A1	5 atributos no inicializados correctamente 3 nombres inadecuados	2	2	3	2	2	1	2	Sí
A2	5 programas duplicados 3 nombres inadecuados 0 código muerto 8 atributos no inicializados correctamente	3	3	3	2	2	1	3	
A3	8 atributos no inicializados correctamente 0 nombres inadecuados	2	1	2	2	2	1	2	Sí me ha ayudado
A4	1 nombre inadecuado 8 atributos mal inicializados	2	1	2	2	2	1	2	
A5	1 programa duplicado 1 nombre inadecuado 1 código muerto 23 atributos no inicializados correctamente	3	2	3	2	2	1	3	Sí

A6	25 atributos no inicializados correctamente 0 nombres inadecuados	1	3	2	2	2	1	1	
A7	2 atributos no inicializados correctamente 3 nombres inadecuados	1	3	3	2	2	1	2	
A8	6 atributos no analizados correctamente 1 nombre inadecuado	1	0	2	2	2	1	3	
A9	6 atributos no inicializados correctamente 0 nombres inadecuados	2	1	2	2	2	1	2	
A10	2 programas duplicados 1 nombres inadecuados 0 código muerto 16 atributos no inicializados correctamente	3	3	2	2	2	1	3	Sí
A11	22 atributos no inicializados correctamente 0 nombres inadecuados	3	1	2	2	2	1	3	
A12	12 atributos no inicializados correctamente	2	1	2	2	2	1	3	
A13	10 atributos no inicializados correctamente	3	1	2	2	2	1	2	No me ha gustado esta página, porque algunos de los errores que me marca no tienen sentido, pero bueno, tampoco está mal
A14	4 atributos no inicializados correctamente 1 nombre inadecuado	1	1	2	2	2	1	2	
A15	4 nombres inadecuados 3 atributos no inicializados correctamente	3	3	2	2	2	1	3	
A16	2 atributos no inicializados correctamente 0 nombres inadecuados	1	3	2	2	2	1	1	
A17	3 programas duplicados 0 nombres inadecuados 0 código muerto 4 atributos no inicializados correctamente	3	3	2	2	2	1	3	No mucho

A18	6 atributos no inicializados correctamente 1 nombres inadecuados	1	3	2	2	2	1	2	A dar un nombre específico a las variables u objetos
A19	5 atributos no inicializados correctamente 2 nombres inadecuados	2	1	2	2	2	1	1	Dar nombres específicos a variables u objetos
A20	6 atributos no inicializados correctamente 6 nombres inadecuados	1	2	2	2	2	1	2	Es una visión demasiado segregada, por separado puede ser simple pero efectivo en conjunto
A21	9 atributos no inicializados correctamente 5 nombres inadecuados	3	1	2	2	2	1	3	
A22	3 programas duplicados 0 nombres inadecuados 0 código muerto 6 atributos no inicializados correctamente	3	3	2	2	2	1	2	
A23	(Paralelismo) 63 atributos no inicializados correctamente 41 nombres inadecuados	3	1	2	2	2	1	2	No, porque hay algunos errores que son debido a los nombres, ya que dejé los de serie
A24	6 atributos no inicializados correctamente	3	1	2	2	2	1	2	
A25	3 atributos no inicializados correctamente	2	3	2	2	2	1	1	Probablemente tenga que mejorar la originalidad
A26	2 programas duplicados 0 nombres inadecuados 1 código muerto 4 atributos no inicializados correctamente	3	3	2	2	2	1	3	No se explican bien los elementos a mejorar, por lo que realmente no se entiende bien la evaluación
A27	1 atributos no inicializados correctamente 0 nombres inadecuados	1	3	2	2	2	1	2	

A28	5 atributos no inicializados correctamente 8 nombres inadecuados	1	3	2	2	2	1	2	
A29	6 atributos no inicializados correctamente 10 nombres inadecuados	3	1	2	2	2	1	3	A dar nombre a los objetos
A30	6 atributos no inicializados correctamente 2 nombres inadecuados	2	2	2	2	2	1	1	
A31	1 programa duplicado 2 nombres inadecuados 10 atributos no inicializados correctamente	3	2	2	2	2	1	3	Sí, me ha ayudado
A32	4 atributos no inicializados correctamente 0 nombres inadecuados	1	3	3	2	2	1	2	
A33	3 atributos no inicializados correctamente 1 nombre inadecuado	1	3	2	2	2	1	1	Sí, se me ocurren muchas soluciones a los errores
A34	2 programas duplicados 0 nombres inadecuados 0 código muerto 2 atributos no inicializados correctamente	3	3	3	2	2	3	3	Sí, hay que mejorar la interactividad con el usuario
A35	4 atributos no inicializados correctamente 1 nombres inadecuados	3	1	2	2	2	1	3	
A36	5 atributos no inicializados correctamente	1	3	3	2	2	1	2	
A37	8 programas duplicados 11 nombres inadecuados 0 código muerto 9 atributos no inicializados correctamente	3	3	2	2	3	1	2	
A38	2 atributos no inicializados correctamente 1 nombre inadecuado	1	1	3	2	2	1	2	
A39	7 atributos no inicializados correctamente 4 nombres inadecuados	3	1	2	2	2	1	3	

A40	1 atributos no inicializados correctamente 1 nombres inadecuados.	1	1	2	2	2	1	1	No
A41	2 atributos no inicializados correctamente 4 nombres inadecuados	1	1	2	2	2	1	1	No
A42	2 atributos ni inicializados correctamente 1 nombre inadecuado	3	0	2	2	2	1	2	
A43	2 nombres inadecuados	1	3	3	2	2	1	2	Sí, me ha ayudado a fijarme en aspectos que había pasado por alto
A44	6 atributos no inicializados correctamente 2 programas duplicados	2	3	2	2	2	3	1	Encuentro útil el apartado “Mejores prácticas”, pero creo que el resto de atributos en “Mejora tu nivel” son un poco subjetivos
A45	3 atributos no inicializados correctamente 1 nombre inadecuado	1	1	3	2	2	1	2	Sí, es posible que me haya ayudado, pues explica qué cosas en concreto son las que empeoran el código y cómo solucionarlo
A46	1 atributos no inicializado correctamente 10 nombres inadecuados	1	1	2	2	2	1	2	
A47	4 atributos no inicializados correctamente 2 nombres inadecuados	3	1	2	2	2	1	2	
A48	1 nombre inadecuado 19 atributos no inicializados correctamente	3	1	2	2	2	1	3	
A49	3 programas duplicados 0 nombres inadecuados 0 código muerto 20 atributos no inicializados correctamente	3	2	2	2	2	1	3	
A50	3 atributos no inicializados correctamente	3	1	2	2	2	1	2	En parte, sobre todo la parte de abstracción
A51	8 atributos no inicializados correctamente 3 nombres inadecuados	2	1	2	2	2	1	2	Sí

A52	4 atributos no inicializados correctamente	1	3	2	2	2	1	2	Veo los aspectos en los que tengo que mejorar pero no la forma en este proyecto tan sencillo
A53	6 atributos no inicializados correctamente 5 nombres inadecuados	3	1	2	2	2	1	3	
A54	1 programas duplicados 0 nombres inadecuados 0 código muerto 3 atributos no inicializados correctamente	3	2	2	2	2	1	3	
A55	4 programas duplicados 0 nombres inadecuados 0 código muerto 70 atributos no inicializados correctamente	3	2	2	2	2	1	3	Si
A56	1 atributos no inicializados correctamente 0 nombres inadecuados	1	3	3	2	2	1	2	No
A57	38 atributos no inicializados correctamente	1	1	2	2	2	1	2	No, creo que el programa no evalúa el resultado final, sólo se centra en el código empleado, aunque luego dentro del *gameplay* la jugabilidad no se vea afectada por los “errores” que detecta el Dr. Scratch
A58	5 atributos no inicializados correctamente	3	1	2	2	2	1	3	
A59	(Paralelismo y sincronización es donde mayor nota he obtenido) 5 atributos no inicializados correctamente 0 nombres inadecuados	3	1	2	2	2	1	3	En algunos aspectos sí he entendido que debo mejorar, pero no al 100%
A60	1 atributos no inicializados correctamente 3 nombres inadecuados	2	1	2	2	2	1	2	

A61	0 programas duplicados 0 nombres inadecuados 0 código muerto 1 atributos no inicializados correctamente	3	1	3	2	2	1	3	
A62	4 atributos no inicializados correctamente 0 nombres inadecuados	1	2	2	2	2	1	1	Sí, mucho
A63	4 atributos no inicializados correctamente 4 nombres inadecuados	1	1	2	2	2	1	2	Creo que la evaluación es inadecuada porque se basa en elementos cuantificables y no en valores propiamente dichos del juego como los *sprites*, jugabilidad, experiencia del usuario...
A64	6 programas duplicados 0 nombres inadecuados 0 código muerto 6 atributos no inicializados correctamente	3	3	3	2	2	1	3	Sí
A65	7 atributos no inicializados correctamente	2	1	2	2	1	3	1	
A66	8 programas duplicados 2 nombres inadecuados 0 código muerto 7 atributos no inicializados correctamente	3	3	3	2	2	1	3	El movimiento de los objetos para que sea más fluido y sin errores
A67	5 programas duplicados 1 nombres inadecuados 3 código muerto 10 atributos no inicializados correctamente	2	1	3	2	2	3	2	
A68	6 atributos no inicializados correctamente 0 nombres inadecuados	1	1	2	2	2	1	1	Sí, he comprobado mis errores y creo que puedo mejorar el programa aprovechando determinados bloques y sensores

A69	1 atributos no inicializados correctamente 31 nombres inadecuados	2	3	2	2	2	1	1	Hay más de treinta objetos y es casi imposible ponerles nombres significativos, puesto que todos cumplen una función parecida
A70	8 programas duplicados 1 código muerto 4 atributos no iniciados correctamente	3	3	3	2	2	1	3	Me ha enseñado a utilizar diferentes programas que tiene Scratch que yo no sabía que tenía
A71	1 programas duplicados 1 nombres inadecuados 0 código muerto 5 atributos no inicializados correctamente	3	1	2	2	2	2	3	Sí, aunque algunos errores (como el nombre del “objeto 1”, que al ser una barra que hace que el rascacielos vuelva a su posición inicial no necesita un nombre específico) son, a mi entender, irrelevantes
A72	2 nombres inadecuados 4 atributos no inicializados correctamente	3	1	2	2	2	1	3	Algunos de los errores que marca están hechos así voluntariamente y exige código, que en tu proyecto puede ser innecesario, para darte mayor puntuación. Pese a ese es una buena forma de tratar de mejorar proyectos futuros.
A73	1 programas duplicados 9 código muerto 16 atributos no inicializados correctamente	1	3	3	2	2	3	3	Me he percatado de algunos errores, aunque algún código muerto me ha sido, y es, de utilidad, por mucho que no pertenezca a un programa propiamente dicho. No obstante iré corrigiendo puliendo el proyecto adaptándolo a las exigencias del corrector.
A74	7 programas duplicados 0 nombres inadecuados 3 código muerto 3 atributos no inicializados correctamente	3	3	3	2	2	1	2	He visto otra forma de solucionar la duplicación de los programas y me ha hecho darme cuenta de cosas que no me había percatado.
A75	4 programas duplicados 4 nombres inadecuados 0 código muerto 8 atributos no inicializados correctamente	3	3	3	2	2	1	3	Sí

A76	10 atributos no inicializados correctamente	1	1	2	2	2	1	1	No entiendo los fallos, los objetos están inicializados correctamente
A77	3 atributos no inicializados correctamente 9 nombres inadecuados	3	1	2	2	2	1	2	Sí, la información es útil, aunque creo que algún apartado no está bien evaluado, puesto que las limitaciones del propio programa hacen que la evaluación falle en algunos aspectos
A78	2 programas duplicados 2 nombres inadecuados 1 código muerto 5 atributos no inicializados correctamente	3	3	3	2	2	1	2	No entiendo por qué me dice que dos programas están duplicados, los programas que me indica tienen los dos su función
A79	1 atributo no inicializado correctamente	1	1	2	2	2	1	1	
A80	4 programas duplicados 12 atributos no inicializados adecuadamente	3	3	3	2	2	1	3	Las inicializaciones de los atributos
A81	2 programas duplicados 5 atributos no inicializados correctamente	3	3	3	2	2	1	3	No
A82	2 atributos no inicializados correctamente 0 nombres inadecuados	3	2	2	2	2	1	2	Si, en que se puede simplificar más el código
A83	1 programas duplicados 0 nombres inadecuados 0 código muerto 9 atributos no inicializados correctamente	3	3	3	3	2	3	3	Aunque yo la inicialización de las variables las hago con una tecla en vez de con la banderita de Scratch, te insiste en que debes darle un valor inicial a cada variable para que estén bien inicializadas

A84	5 programas duplicados 0 nombres inadecuados 0 código muerto 26 atributos no inicializados correctamente	2	3	3	2	2	1	2	Me ha ayudado bastante a utilizar los diferentes bloques y qué problemas existen al interactuar unos objetos con otros y que soluciones se pueden conseguir. Me habría gustado que se tuviese en cuenta el aspecto final o estética o resultado final del juego. Por ejemplo, mi juego no es que sea algo demasiado complicado pero dediqué una gran parte del tiempo en crear todas las animaciones y en buscar diferentes efectos de sonido que estuviesen relacionados con la atmósfera que tiene el juego en sí. De todas maneras, esta práctica me ha ayudado mucho a practicar lo que sabía de programación y a aumentar los conocimientos para la siguiente práctica. Una de las cosas que más motiva al utilizar Scratch es que puedes ver inmediatamente los resultados del código de forma gráfica y crear juegos simples como lo que los alumnos hemos hecho
A85	5 atributos no inicializados correctamente 1 nombre inadecuados	3	1	2	2	2	1	3	
A86	5 programas duplicados 2 nombres inadecuados 0 código muerto 13 atributos no inicializados correctamente	3	3	3	2	3	3	3	
A87	3 atributos no inicializados correctamente 1 nombres inadecuados	3	1	2	2	2	1	2	Sí, me ha informado de algunas funciones que no conocía
A88	8 atributos no inicializados correctamente 1 nombre inadecuado	1	3	2	2	2	1	2	

A89	6 atributos no inicializados correctamente 0 nombres inadecuados	1	1	2	2	2	1	1	No exactamente. Mi juego lo jugué y lo di a probar a mis compañeros y pareció divertido, que es de lo que se trata un juego. Utilicé casi todas las funciones practicadas en clase, demostrando así los conocimientos adquiridos. Quizás añadirle un final al llegar a cierta puntuación o la creación de un nivel más difícil hubiese hecho un mejor juego

Apéndice D: Encuesta de Opinión sobre Scratch y Java

Encuesta Final: Estudio sobre Scratch y Java

Este formulario permite recabar información de los estudiantes del curso de Programación Visual del otoño de 2016 del Campus Móstoles de la Universidad Rey Juan Carlos, en la Comunidad de Madrid (España)

Selecciona el círculo que corresponde a tu respuesta [El conocimiento de programar adquirido con Scratch me ayudó después a aprender: Java en general]

- o Sin opinión
- o Totalmente en desacuerdo
- o Algo de acuerdo
- o De acuerdo
- o Totalmente de acuerdo

Selecciona el círculo que corresponde a tu respuesta [El conocimiento de programar adquirido con Scratch me ayudó después a aprender: las expresiones de Java]

- o Sin opinión
- o Totalmente en desacuerdo
- o Algo de acuerdo
- o De acuerdo
- o Totalmente de acuerdo

Selecciona el círculo que corresponde a tu respuesta [El conocimiento de programar adquirido con Scratch me ayudó después a aprender: las instrucciones condicionales de Java]

- o Sin opinión
- o Totalmente en desacuerdo
- o Algo de acuerdo
- o De acuerdo
- o Totalmente de acuerdo

Selecciona el círculo que corresponde a tu respuesta [El conocimiento de programar adquirido con Scratch me ayudó después a aprender: las instrucciones iterativas (bucles) de Java]

- o Sin opinión
- o Totalmente en desacuerdo

- Algo de acuerdo
- De acuerdo
- Totalmente de acuerdo

¿Hay algo que eches de menos en la asignatura?

_____

¿Hay alguna parte de la asignatura que crees que debería dedicársele más tiempo?

_____

¿Hay algo que crees que sobra en la asignatura?

_____

¿Hay alguna parte de la asignatura que crees que debería dedicársele menos tiempo?

_____

Apéndice E: Respuestas a la Encuesta de Opinión Final

Se incluyen solamente las respuestas a las cuatro preguntas abiertas.

Alumno	¿Hay algo que eches de menos en la asignatura?	¿Hay alguna parte de la asignatura que crees que debería dedicársele más tiempo?	¿Hay algo que crees que sobra en la asignatura?	¿Hay alguna parte de la asignatura que crees que debería dedicársele menos tiempo?
A1	no tengo conocimientos de java, para saber si ha faltado algo por ver	la parte de ficheros	creo que no	dedicarle mucho menos tiempo a scratch
A2	No	Ficheros	Scratch	Scratch
A3	Más explicaciones en cuanto a ficheros y arrays de más dimensiones.	Ficheros y arrays	No	Scratch
A4	Explicaciones más claras sobre las clases de java y los ficheros.	Los archivos y las clases, porque no se ha explicado lo suficiente para lo que se exige en las prácticas.	Algunas de las clases dedicadas a Scratch fueron, en mi opinión, innecesarias y un poco infantiles.	Scratch
A5	No	La parte de ficheros	No	No
A6	Sí. Me hubiera gustado aprender mejor cómo se trabaja en equipo al programar.	Sí. En mi opinión, el último tema de ficheros y registros y abordar un poco más igualmente en el tema de recursividad.	No. Personalmente, cuanto más conocimiento tenga uno, mejor.	Quizá con Scratch estuvimos demasiado tiempo, en mi opinión. Sin embargo y a pesar de poseer ya conocimientos sobre él, me ayudó bastante a comprender cómo funciona y de qué trata la programación a nivel general.
A7	El significado de muchas cosas que simplemente escribimos. Cosas como public static, serializable, etc.	Clases y ficheros.	Scratch.	El concepto de las variables.

A8	No se si es parte del temario de la asignatura, pero vendría bien algo de programación orientada a objetos. Clases, instancias, herencia, aunque fuera básico. Yo ya conocía java anteriormente, por eso he contestado “sin opinión” a las preguntas sobre scratch, pero al explicar bucles y otras cosas quizás habría venido bien precisamente que se comparen con scratch así los conocimientos de scratch hubieran ayudado mas a aprender java.	Quizás la parte de ficheros a los compañeros no les ha quedado demasiado clara en general, es posible que por falta de tiempo.	No	No
A9	Que se escriba código con los alumnos	Búsquedas	Ficheros	Variables
A10	Explicar las cosas en el ordenar en vez de en la pizarra, que se ve peor.	Ficheros.	Scratch	Scratch
A11	Mas practica en clase, plantear ejercicios y dar al alumno la posibilidad de poder resolverlo en clase.	Ficheros y arrays	El tema 1 (conceptos básicos) fue demasiado extenso	La teoría en general, creo que se podría aprender mejor en esta asignatura dedicando mas tiempo a la parte práctica.
A12	No	Ficheros	No	Arrays
A13	No	Arrays y ficheros	No	Primeros temas
A14	No	Arrays y ficheros	La parte de Scratch y las clases en la pizarra	Scratch
A15	No	En general se le ha dedicado poco tiempo a terminar los temarios	No	Scratch quitó más de lo necesario

A16	Darle un enfoque más "profesional", aunque los alumnos no tengan experiencia previa si se explica desde el principio cómo hacer un código eficiente, limpio, la subprogramación, creo que puede ser positivo. También explicar un poco más cómo hacer debug o solucionar problemas.	Subprogramación (explicar cuando se debe usar, cuándo no, qué parámetros pasar y cuándo es mejor pasarlos o no, cuándo se debe hacer que devuelva) recursividad, arrays y ficheros.	Hacer dos grupos para ver si SRec ayuda o no, para mí es perder bastante tiempo y hace que no se centre tanto en la recursividad. Además luego no se explica al resto de compañeros cómo hacerlo con/sin Srec sino que tiene que apañárselas leyendo un manual. Y hacer tantas prácticas en clase de Scratch resulta cansado y repetitivo cuando al final son casi iguales, hacer una criba para ver las más difíciles pero asequibles y centrarse en aquellas que tengan que ver con Java (pues hay cosas que dimos en Scratch que luego en Java no se pueden hacer, como el cambiar de disfraz, y no veo el sentido a explicar eso), le quitaría tiempo a esa parte.	Scratch y pseudocógico. Toda la parte del principio que es la más suave ha tenido mucho tiempo y los temas finales que es donde se pueden tener más dudas y problemas han sido algo rápidos en mi opinión, en especial ficheros. También hay explicaciones que para mí han sido demasiado largas, escribir el mismo código 3 veces cuando se podría cambiar simplemente una parte del código o volver a explicar sin necesidad de volver a escribirlo ahorraría tiempo.
A17	No	El ultimo tema es el más complicado y apenas se le ha dedicado tiempo mientras que a cosas mucho mas sencillas se les han dedicado varias clases	No	En mi opinión podría dedicarse menos tiempo a "leer" la teoría de las presentaciones y en su lugar hacer más ejemplos prácticos en clase con la profesora

A18	No.	Explicaciones de las practicas y alguna que otra corrección de estas, estaría bien y sería muy útil al menos saber en que hemos fallado en estas antes del examen, poco a poco mientras las vamos haciendo ya que usamos cosas en cada práctica que pueden no haber quedado claras en la anterior y se vuelve un poco una bola de nieve.	No sé hasta que punto esto sería útil o no, pero el scratch quizás es demasiado tiempo, como concepto a la hora de introducir a la asignatura quizás funciona pero al llegar a java no se nota lo suficiente para el tiempo dedicado a este.	Lo he mencionado antes.
A19	no	SI, creo que realmente deberia dedicarsele mas tiempo a ficheros, por ser lo mas dificil visto en un dia.	Quiza quitaria Scratch y ese tiempo se lo sumaria a ficheros.	A Scratch, no lo veo tan útil.
A20	Nada	Subrutinas y ficheros	No	No
A21	No	Ficheros	No	No
A22	Poder conocer algo mas de mis calificaciones antes del final del cuatrimestre para saber de mis errores e intentar aprender de ellos	A los ficheros, registros, subprogramacion, arrays y recursividad	Los test para evaluarnos, debido a que considero los test bastante injustos a la hora de demostrar conocimientos de programacion	A las variables
A23	No	El último tema	SRec	No
A24	No	Quizá podríamos haberle dedicado algo más de tiempo a la parte de ficheros	No	Igual deberíamos haberle dedicado menos tiempo al inicio de programación en java cuando estábamos con las variables
A25	Mejor estructuración de los temas	La parte de ficheros y registros	Los exámenes de tipo test.	Al Scratch
A26	No	Más tiempo a la última parte de Arrays y ficheros	La parte de Scratch	La parte del principio de Java de las variables
A27	Conocer la nota de las prácticas antes del final del cuatrimestre	A los ficheros y registros.	Los test, que no sirven para demostrar conocimientos, ya que con eso basta con las prácticas	Scratch, aunque fue de ayuda, ocupó demasiado tiempo

A28	Diagramas de flujo	Practicas	Scratch	Ejemplos en pseudocodigo. Más programación junto a los alumnos demostrando y arreglando los posibles fallos
A29	No	Sí. Debería hacerse mas hincapié en el tema de métodos y ficheros.	Sobra la práctica inicial de Scratch, era innecesaria.	No
A30	No	Ficheros	No	Scratch
A31	Aplicación al campo de los videojuegos	Ficheros y recursividad.	No	Scratch
A32	Ejercicios con solución en la parte de archivos y array mezclados	Leer y Escribir .dat	Scratch solo debería durar un día	S.O.P. debería darse con el Hola Mundo el primer día y listo, y el for e if también en un par de clases dejando que se practique, porque es fácil de entender.
A33	No realmente	Creo que menos al principio y sí más a arrays y ficheros	No	Scratch es útil para empezar, pero quizás una o dos clases menos estaría bien
A34	Quizás algún profesor auxiliar más. Sé que es una petición un tanto inaccesible, y confío en las capacidades de la profesora, pero creo que a veces se desborda y no llega a cumplir las expectativas del alumno. Como acabo de decir, no es culpa suya, cualquier profesor tendría el mismo problema, es culpa del poco acompañamiento que recibe para dar clase.	En mi opinión, creo que las prácticas deberían ocupar la mayor parte del tiempo, no tanto las clases teóricas. Programar es una tarea de ensayo y error, se aprende haciendo diferentes programas y fallando en el intento para después conseguir corregir el fallo.	Sobrar no sobra nada, todo es indispensable, la única pega es la manera de repartir esos apartados. Es decir, me refiero a dedicar más tiempo a programar, a practicar y no tanto a intentar entender conceptos desde fuera, sin probarlos.	Se le debería dedicar menos tiempo a la teoría, pues son conceptos abstractos que, aunque inicialmente necesarios, deberíamos ponerlos inmediatamente en práctica nosotros mismos tras la explicación para llegar a comprenderlos lo más rápido posible. Es decir, en lugar de explicar el código, es mucho mejor ir desarrollándolo poco a poco mientras el alumno sigue los pasos del docente. Creo que de esta manera se conseguiría un desarrollo más notable por parte del alumno.
A35	Que se les ponga nota a las practicas	A los ficheros	Los ficheros	A scratch
A36	Una transición de Scratch a java más lenta, porque al no conocer el lenguaje de programación, sigue chocando mucho	Ficheros	No	No

A37	Explicaciones más detalladas de cada tema y ejemplos prácticos.	Explicación y resolución de los ejercicios.	No.	A la explicación puramente teórica.
A38	Alguna explicación mas sobre conceptos y uso de NetBeans.	Ficheros.	No.	Crear la práctica final de Scratch, solo aprender con él, pero no tener que presentar nada.
A39	tiempo para practicas	ficheros	scratch	primer tema de variables
A40	Más ejercicios prácticos	La práctica	No	No
A41	No.	Ficheros y recursividad.	No.	La introducción y nociones básicas.
A42	No, en general no.	Sí, a la corrección de las prácticas en clase, y a la práctica en clase de los ejercicios más complicados.	No, creo que está equilibrada con respecto a lecciones y prácticas.	No, si acaso más tiempo a las mencionadas ante.
A43	No.	Sí, a partir de los arrays.	Algunas clases de scratch.	Scratch. Aunque ayude para entender mejor java podría darse más rápido.
A44	La rápidez en la correción de las prácticas	No	No	No
A45	Un aula más grande	A los ficheros	No, creo que está todo bien organizado y que es importante	A las instrucciones condicionales
A46	Usar otros lenguajes de programación.	Como sacar nuestro programa a un fichero.	Scratch	Las variables
A47	No.	No.	No.	No.
A48	La utilización de diagramas de flujo para plantear soluciones a problemas que requieran más esfuerzo mental del habitual al escribir el código del programa. Ejemplo: práctica de los números primos o métodos recursivos.	Creo que debería dedicarse más tiempo a trabajos de programación más amplios en lugar de muchos pequeños, así se podría relacionar todo lo aprendido en un programa más útil y por lo tanto más motivador a la vez que aparecen errores más frecuentemente que tendrán que solucionarse.	No.	Quizá la utilización de pseudocódigo tan frecuentemente, porque una vez iniciados en el lenguaje (Java), resulta casi igual de intuitivo expresarlo en pseudocódigo como en código.
A49	Métodos constructores	Arrays	No	Recursividad
A50	No	No	Si, sobran los ficheros	No

A51	No, yo creo que se ha dado lo suficiente para aprender lo básico sobre programación.	En general, se ha explicado bastante bien la asignatura. Tal vez la recursividad y la parte de ficheros es la que más cuesta entender, pero creo que se le ha dedicado suficiente tiempo a todas las partes.	No, yo creo que todo lo dado en la asignatura es útil. Es verdad que la parte ficheros es bastante diferente al resto de materia que se da en la asignatura, ya que no se trata de ningún objeto o bucle en sí, pero en realidad esta pequeña introducción que se dio sobre ficheros es muy útil e importante.	No, yo creo que se ha dedicado suficiente tiempo a todas las partes de la asignatura, tanto a la parte teórica como a la parte práctica y de ejercicios.
A52	La corrección inmediata de las prácticas individuales. Sabiendo las notas, sabes si vas bien encaminado y sabes si estás haciendo bien las cosas. Sin las correcciones, vas a ciegas y no sabes si realmente haces todo lo que se pide. Aunque luego se plantee una posible solución, considero que la nota puede aportar al desarrollo individual.	A ficheros. En general, no se si es por mi, todo lo demás lo entendí bastante bien, pero ficheros me ha caído muy pesado y considero que no se le ha dedicado el tiempo necesario para comprenderlo correctamente.	En mi opinión, Scratch debería ser usado como herramienta para entender bucles y condiciones, haciendo menos prácticas y aprovechando mejor el tiempo.	A Scratch
A53	Sí, poder programar un juego, aunque sea sencillo, con java.	Sí, a ficheros.	Sí, Scratch.	Sí, a Scratch.
A54	No	Realizar más prácticas individuales en clase	Tanta teoría	Sí, la teoría pura, son mejores los ejemplos y el practicar nosotros mismos
A55	Las notas de las prácticas	Un poco más a la parte de array	Scratch	La parte de Scratch
A56	Explicación en ficheros con más tiempo, clases... Corrección de las practicas al ritmo del curso.	Ficheros	Scratch	Variables
A57	Ahora mismo, no se me ocurre nada.	Sin duda, a los arrays.	No creo que haya algo que necesariamente sobre en la asignatura.	Dado que no le veo gran utilidad, creo que se debería dedicar menos tiempo a la recursividad.

A58	Que los programas no sean tan repetitivos y pesados de hacer.	Toda la parte de ficheros.	Scratch.	Scratch (aunque como ya he respondido antes, sobra)
A59	Poca cosa, este año ha sido el doble de mejor que el año pasado	Ficheros, sinn duda, y creo que ha sido algo general	Está más o menos equilibrada	Scratch
A60	La verdad, nada, el año pasado fue un desastre y este ha sido muchísimo mejor.	A la parte de Ficheros, más teoría para poder realizar bien las partes prácticas.	No.	No quitaría Scratch pero quizás menos tiempo dedicado a para él mejor.
A61	En las clases de teoria, enseñar más con el ordenador	A los ficheros	Quizás Scratch	A la recursividad
A62	Sí	Sí	Sí	Sí
A63	No	A la parte de ficheros en mi opinión	No	Para mí, a la parte del comienzo de la asignatura (qué es una variable, explicación teórica de los bucles, ...), aunque estaría bien explicar cada término, no creo que se les deba dedicar más de una o dos clases, porque creo que se pueden llegar a aprender mejor de forma práctica
A64	Nada en especial	A la parte de ficheros	La parte de scratch	No
A65	No	A la parte de ficheros no le dedicamos el tiempo suficiente, creo yo	No	No
A66	No	Ficheros y registros	No	El contenido del principio ocupó demasiado tiempo para su dificultad.
A67	No hemos visto como crear alguna interfaz de algún tipo,considero que estaría bien explicarlo aunque fuese brevemente.	Debería dedicarse mas tiempo a explicar como funciona el código de programación.	Las clases de teoría como tal,las enfocaría de otra manera,ya que sin ordenador donde aplicar el código explicado,explicaría pseudocodigo,dudas o alguna forma distinta de acercarse a la programación.	Creo que le dedicamos demasiado tiempo a Scratch

A68	No	Quitaría Scratch y le daría más tiempo a Java	Scratch	Scratch
A69	Más ejemplos	Ficheros	Scratch	Scratch
A70	Algo más enfocado a videojuegos, como por ejemplo diferentes comandos que se usen en algún motor sencillo para hacernos a la idea de como se aplica en nuestro ámbito.	Sobre todo a ficheros, es lo que más complicado me pareció como para dedicarle tan poco. También me quedó poco claro la recursividad, ya que hicimos pocos ejercicios complejos y ninguno obligatorio.	No creo que le sobre nada, pero que tiene que ajustar los tiempos para cada cosa más convenientemente.	Quizás un poco menos a Scratch y al principio de la asignatura donde dimos la clase String o bucles sencillos.
A71	Alguna aplicación a los videojuegos	La parte final de ficheros	No nada	A la parte de scratch
A72	Un poco más de profundidad en el aspecto Java. Quizás tambien que los examenes fuesen en ordenador y no en papel.	a Java en general	No, pero se podría aprovechar mejor el tiempo	Scratch quizás se podría haber visto en dos días máximo
A73	Una aplicación más directa con los videojuegos	A ficheros	No	No
A74	No	No	No	No
A75	No	A los ficheros	No	No
A76	Que se trate más en clase los ejercicios del tipo de la prácticas a entregar.	Explicación de las prácticas a entregar.	No	No

Apéndice F: Encuesta de Opinión sobre Suprimir Scratch

Dos preguntas más para mejorar la asignatura

En una encuesta realizada entre los alumnos de Programación Visual en enero, muchos de vosotros contestasteis que debería reducirse el tiempo dedicado a Scratch (apenas dos semanas) o incluso suprimirse. Nos resultaría muy clarificador si contestas a estas dos preguntas:

¿Estás de acuerdo con suprimir Scratch de la asignatura?

- o Sí
- o No

¿Por qué?

_____

Apéndice G: Respuestas a la Encuesta de Opinión sobre Suprimir Scratch

Alumno	¿Estás de acuerdo con suprimir Scratch de la asignatura?	¿Porqué?
A1	Sí	Es bastante común haber visto Scratch con anterioridad en la ESO y recorta tiempo de otras partes del temario.
A2	No	No suprimirla, dedicarle menos tiempo del que se merece
A3	Sí	No resulta útil para aprender la programación más "seria"
A4	No	Ayuda a entender el resto de la asignatura de una forma visual, pero si que creo que es demasiado tiempo ya que al final del temario se llega con poco tiempo.
A5	Sí	Innecesario
A6	Sí	
A7	No	Sirve bien para presentar conceptos de la programación, pero se le dedica excesivo tiempo.
A8	No	es entretenido
A9	Sí	Lo veo innecesario y consume bastante tiempo del cuatrimestre.Debería dedicarse ese tiempo a preparar mejor la parte de java, sobre todo ficheros.
A10	No	Considero que es útil para familiarizarse con las estructuras de programación que se ven más adelante. Aun así, sí que creo que quizás debería reducirse un poco el tiempo dedicado al mismo.
A11	Sí	
A12	Sí	
A13	No	No creo que sea necesario suprimirlo, pero si dedicarlo menos tiempo. Creo que para los quejuegos llegan nuevos sin saber nada de programación como en mi caso, es útil para entender como funciona la programación, pero si es cierto que se debería reducir el tiempode invertido en esa parte.

A14	Sí	Actualmente, en los institutos ya se estudia Scratch. Además, no sé si será porque yo ya sabía cosas básicas de la programación, no me aportó nada haber aprendido Scratch.
A15	Sí	No aprendí gran cosa con ello y ese tiempo se podría haber dedicado a otra cosa
A16	No	Ayuda bastante a entender un poco los conceptos de programacion basicos si no tienes ni idea de programar, pero si es cierto que con dedicarle 2 o como mucho 3 clases y una pequeña practica bastaria y asi emplear algo mas de tiempo en explicar la parte de ficheros ya que es la mas complicada.
A17	Sí	
A18	Sí	
A19	No	Empezar la asignatura con Scratch aclara conceptos para continuar después en Java, por lo que suprimirlo sería innecesario. Pero reducir el tiempo dedicado podría ser bueno ya que puede llegar a ser pesado.
A20	No	Es útil para visualizar mejor la programación al inicio, aunque ya se está empezando a impartir en las escuelas como asignatura extraescolar, es recomendable en mi opinión verlo pero con casos más genéricos y reducir su uso a la primera semana o primeras 2 semanas, como introducción al mundo de la programación, pero dedicar más tiempo a asentar mejor los conceptos básicos de java.
A21	No	Para la gente que no tiene conocimientos previos sobre programación viene muy bien para poder introducirse a dicho campo. Lo único es que no se debería estar tanto tiempo centrándose en él. Con 2 clases basta.
A22	No	Otorga algo de libertad creativa y visión de diseño del videojuego entre tanto desarrollo que llega a cansar de la carrera.
A23	Sí	
A24	Sí	Hay maneras mejores de explicar los conceptos que se intentan explicar
A25	Sí	En mi caso no me sirvió de nada de cara a la programación en java, y pienso que ese tiempo dedicado a Scratch podría emplearse en fortalecer o profundizar algunas partes del temario.

A26	Sí	Pienso que es mas necesario dedicar mas tiempo a la programacion en java para explicar mejor los conceptos y tener mas tiempo para resolver ejercicios en clase, etc.