Actas de las Jenui, vol. 7. 2022.
Páginas: 379-382

Experiencias de uso del Pseudocódigo y Java en la enseñanza de programación en Ciclos Formativos y Bachillerato

Antonio López García
LITE – Laboratorio de Tecnologías de la Información en la Educación
Universidad Rey Juan Carlos
Madrid, España
a.lopezg.2018@alumnos.urjc.es

Jaime Urquiza-Fuentes
LITE – Laboratorio de Tecnologías de la Información en la Educación
Universidad Rey Juan Carlos
Madrid, España
jaime.urquiza@urjc.es

Resumen

El Pseudocódigo ha sido la principal herramienta utilizada para el comienzo del aprendizaje de la programación con lenguajes textuales. Las ventajas que le suelen atribuir son su sintaxis sencilla y el uso de la lengua materna, lo que debe dar lugar a una menor carga cognitiva de los estudiantes, facilitando así las primeras experiencias con un lenguaje de programación.
Aunque se ha usado frecuentemente en la enseñanza universitaria, parece que su uso es menor en Bachillerato y Formación Profesional, probablemente por el tiempo disponible para la enseñanza de la programación en estos casos.
Este trabajo es un estudio exploratorio sobre el efecto del Pseudocódigo en un curso sobre fundamentos de la programación en Bachillerato y Ciclos Formativos de Grado Medio y Superior. En el estudio participaron 4 grupos, con un total de 46 estudiantes. Mientras el grupo que trabajó con Pseudocódigo invirtió una media de 20 horas para alcanzar un nivel aceptable de conocimientos, los otros tres grupos, que trabajaron con Java, emplearon una media de 35 horas. Estos resultados nos animan a investigar en mayor profundidad el impacto del uso del Pseudocódigo en la enseñanza de la programación en niveles preuniversitarios.

Abstract

Pseudocode has been the main tool used to start learning programming with textual languages. The advantages that they usually attribute to it are its simple syntax and the use of the mother tongue, which should lead to a lower cognitive load for students, thus facilitating their first experiences with a programming language.
Although it is frequently used in university teaching, it seems that its use is less in Baccalaureate and Vocational Training, probably due to the time available for teaching programming in these cases.
This work is an exploratory study on the effect of the Pseudocode in a course on fundamentals of programming in Baccalaureate and Intermediate and Higher Level Training Cycles. Four groups participated in the study, with a total of 46 students. While the group that worked with Pseudocode spent an average of 20 hours to reach an acceptable level of knowledge, the other three groups, who worked with Java, spent an average of 35 hours. These results encourage us to further investigate the impact of the use of Pseudocode in the teaching of programming at pre-university levels.

Palabras clave

Pseudocódigo, Java, lenguaje de alto nivel, educación, carga cognitiva, lengua materna.

1. Introducción

El Pseudocódigo es una herramienta que se ha utilizado de forma sistemática al inicio de la enseñanza en lenguajes de programación. Presentar a los estudiantes conceptos abstractos de una forma más fácil puede significar una mejor asimilación de los mismos, una mayor motivación, y consecuentemente, una mejora en los resultados. Así, se pueden encontrar estudios que detectan efectos positivos en el proceso de aprendizaje de la programación al usar Pseudocódigo, p.ej. [2].

Podría decirse que el Pseudocódigo es una descripción de un algoritmo, que se puede expresar en cualquier lengua existente. El Pseudocódigo nos puede proporcionar dos ventajas fundamentales. Por un lado, se encuentra el uso de la lengua materna. Existen estudios que vinculan el dominio de una lengua al fracaso escolar [6] o a la ansiedad que crea en los estudiantes [1]. La programación en lengua materna

puede hacer que el contexto de aprendizaje sea más asequible, reduciendo la carga cognitiva para el estudiante [1]. Por otro lado, la simplicidad de sintaxis puede facilitar la compresión. De nuevo, puede que haga disminuir la carga cognitiva que sufren los estudiantes al comenzar a aprender programación, transitando de conceptos simples a otros más complejos [4]. Se pueden encontrar entornos de programación con fines educativos que usan el Pseudocódigo como lenguaje de programación, p.ej. [5].

Es frecuente encontrar estudios, que pretenden desvelar qué lenguaje es más apropiado emplear con los estudiantes noveles de programación, ya que enseñar programación a principiantes es una tarea compleja [7].

Un estudio realizado en 2014 revela que el uso de un lenguaje sintácticamente simple como Python, en lugar de uno más complejo como Java, parece que facilita el aprendizaje de los conceptos de programación a los estudiantes iniciales [3].

La industria demanda cada vez más profesionales informáticos, pero en muchos casos el porcentaje de alumnos de informática que abandonan los estudios es elevado, puede ser la razón de que introducir a principiantes en lenguajes de programación como Python está ganado terreno, frente a otros lenguajes con una sintaxis y gramática más compleja como Java y C++ [8]. La claridad de sintaxis y un entorno de desarrollo simple pueden ser aspectos a tener en cuenta en las primeras experiencias de aprendizaje de programación..

Este trabajo pretende ser un estudio preliminar del impacto del uso del Pseudocódigo en la enseñanza de programación en niveles preuniversitarios.

2. Descripción del estudio

El estudio se realiza a lo largo de 3 cursos académicos, en diferentes trimestres y con estudiantes de enseñanzas medias de institutos españoles de Formación Profesional y Bachillerato. El enfoque frecuentemente utilizado en la enseñanza de programación dentro de la formación profesional no suele contar con el Pseudocódigo. Contaremos con cuatro grupos distintos, tres de ellos comenzarán directamente con un lenguaje de alto nivel como Java, mientras que el cuarto grupo lo hará con Pseudocódigo.

Desde un punto de vista general, los cuatro grupos son similares en cuanto a motivación, edad de los estudiantes, capacidades académicas y, especialmente, la falta de conocimientos previos de programación. Aun así, los grupos pertenecen a diferentes centros, cursos y sus estudiantes tienen diferentes circunstancias personales.

2.1. Sujetos de estudio

Los participantes en el estudio son 46 estudiantes con edades comprendidas entre 17 y 21 años, con algunas excepciones que llegan hasta los 29. Como se dijo anteriormente, están distribuidos en cuatro grupos diferentes que describimos a continuación.

El estudio se realiza en 4 institutos diferentes. En dos grupos, con alumnos de 2º curso de Bachillerato, un grupo con alumnos de 1º de Formación Profesional, Grado Superior en Informática, y un cuarto grupo con alumnos de Formación Profesional, Grado Medio en Informática. A continuación se detalla cada uno de ellos:

- GRUPO 1. Se trata de un grupo de 9 alumnos, de segundo curso de Bachillerato, con edades comprendidas entre 17 y 19 años. La asignatura donde se realiza el estudio es “Tecnología de la Información y las Comunicaciones 2”, que es elegida de forma optativa. Se imparte durante el segundo trimestre y, entre otros contenidos, está destinada al aprendizaje del lenguaje de programación Java.
- GRUPO 2. Se trata de un grupo de 24 alumnos de primer curso de Formación Profesional, Grado Superior en Informática, con edades comprendidas entre 18 y 29 años. La asignatura donde se realiza el estudio es “Programación de Ordenadores”. En este caso, esta asignatura es obligatoria, se imparte en el primer trimestre y también se usa el lenguaje Java.
- GRUPO 3. Este grupo está compuesto por 6 alumnos, de 17 y 18 años, la asignatura es la misma que el GRUPO 1 aunque corresponde a un centro diferente.
- GRUPO 4. Se trata de un grupo de 7 alumnos de primer curso de Formación Profesional, Grado Medio en Informática, con edades comprendidas entre 17 y 21 años. La asignatura donde se realiza el estudio es “Aplicaciones Ofimáticas”. En este caso, esta asignatura es obligatoria y se imparte en el tercer trimestre.

Aunque existen diferencias entre los estudiantes de cada grupo podemos contar con que todos tienen una motivación similar, puesto que los grupos 1 y 3 eligen voluntariamente cursar la asignatura mientras que los estudiantes de los grupos 2 y 4 tienen clara su intención de enfocar su vida profesional hacia la Informática.

2.2. Tratamientos

El objetivo del trabajo es hacer una primera exploración del impacto del Pseudocódigo en la enseñanza de la programación. Tres de los grupos seguirán un tratamiento basado en la enseñanza de Java mientras

Grupo	%(N) est. en nivel 5	%(N) est. en nivel 4	%(N) est. en nivel 3	%(N) est. en nivel 2	Nivel medio	Tiempo dedicado
1	33.3% (3)	44.4% (4)	22.2% (2)	0.0% (0)	4,11	40 horas
2	37.5% (9)	29.2% (7)	25.0% (6)	8.3% (2)	3,96	31 horas
3	33.3% (2)	50.0% (3)	16.7% (1)	0.0% (0)	4,17	34 horas
4	42.9% (3)	28.6% (2)	14.3% (1)	14.3% (1)	4	20 horas

Cuadro 1: Resultados de los grupos

que el cuarto se basará en la enseñanza de Pseudocódigo. A continuación, se describen ambos tratamientos.

En lo que respecta al aprendizaje de Java, el hecho de usar este lenguaje no significa que se haya puesto el foco en el paradigma orientado a objetos. Aunque se use Java, la enseñanza se ha centrado en el bloque de fundamentos de programación estructurada que comprende conceptos básicos: sintaxis de Java, tipos de datos, operadores y estructuras básicas de control de flujo (condicionales y bucles). Después de explicar cada concepto, se exponen ejemplos de utilización y posteriormente los alumnos realizarán un conjunto de ejercicios. El entorno de programación comprende un editor de texto simple y un compilador de Java en línea usado desde página web.

El enfoque basado en Pseudocódigo también se ha centrado en la enseñanza de la programación estructurada en Pseudocódigo. En este caso, el bloque de fundamentos de la programación solo se diferencia del anterior en la sintaxis del Pseudocódigo. La metodología de trabajo es similar a la anterior, con la salvedad del entorno de programación. En este caso hemos utilizado la herramienta PSeInt [5], que permite codificar los programas y visualizar sus resultados.

2.3. Variables dependientes

Durante el tiempo que duró el estudio con cada grupo se realizó un seguimiento personalizado de cada alumno, evaluando continuamente sus progresos y aprendizajes.

Las variables dependientes utilizadas en el estudio son dos: el nivel de conocimientos alcanzado por los estudiantes y el tiempo empleado por cada grupo.

Para medir el nivel alcanzado por los estudiantes se usó la siguiente rúbrica comprensiva, con una escala de 0 a 5, considerando 5 el grado más alto de compresión y 0 el más bajo:

- 0: No se realiza nada.
- 1: No se comprende la actividad que se ha planteado.
- 2: Algo de comprensión, pero no se incluyen elementos requeridos.
- 3: Comprensión parcial, con algunos de los elementos requeridos.
- 4: Comprensión, con un alto grado de elementos requeridos.
- 5: Comprensión total de la actividad, todos los elementos requeridos.

El tiempo empleado por cada grupo mide la cantidad de horas de trabajo que ha utilizado el grupo en alcanzar el nivel idóneo de comprensión y aprendizaje deseado en los conceptos sobre fundamentos de programación: conceptos básicos, sintaxis del lenguaje (Java o Pseudocódigo), tipos de datos, operadores y estructuras básicas de control de flujo (condicionales y bucles).

En el momento que hemos llegado a ver y practicar todos los contenidos del bloque inicial de fundamentos de programación, y además se ha alcanzado el nivel de puntuación media próximo a 4 en las rúbricas comprensivas de todos los alumnos de la clase, es cuando la clase tiene, a nivel general, un nivel aceptable de compresión.

2.4. Procedimiento

Durante 3 cursos académicos, y con los grupos antes mencionados, se midieron tanto el nivel alcanzado como el tiempo dedicado por cada grupo en aprender un primer bloque de fundamentos de programación con un nivel idóneo.

En cada sesión, después de la explicación que siempre iba acompañada de un conjunto de ejemplos, se proponían un conjunto de ejercicios que los alumnos realizaban y entregaban resueltos. La resolución de estos ejercicios era apoyada por el profesorado solventando dudas a los alumnos que tenían dificultades.

Tras recibir las actividades resueltas de cada alumno y trabajar directamente con ellos, calificamos su nivel. Si no se alcanzaba el nivel de comprensión y aprendizaje, se volvía atrás y se repetían los ejercicios y las explicaciones hasta alcanzarlo.

3. Resultados

El Cuadro 1 muestra un resumen de los niveles alcanzados por los estudiantes de cada grupo (filas). Desde la columna segunda a la quinta se muestran el porcentaje y el número de estudiantes que alcanzan un determinado nivel en cada grupo. La sexta colum-

na muestra el nivel medio y la séptima el número de horas de trabajo dedicadas en cada grupo.

4. Conclusiones

El Pseudocódigo es uno de los métodos más utilizados para el diseño de algoritmos, y es independiente del lenguaje de programación que se vaya a utilizar. Además, es una herramienta utilizada en la enseñanza de programación porque está escrita en lengua materna (castellano en nuestro caso), su sintaxis es sencilla y el código es claro; todo esto ayuda a facilitar la comprensión de los programas escritos en Pseudocódigo.

Según los resultados y desde un punto de vista global, se ve que los grupos que han aprendido con Java, han empleado un tiempo medio de 35 horas en adquirir un nivel idóneo, mientras que el grupo que usó Pseudocódigo empleó 20 horas. Desde un punto de vista más detallado se ve que el grupo de Pseudocódigo es el que consiguió mayor porcentaje de estudiantes con nivel 5, con un 42,9%. Este resultado se difumina considerando los otros dos niveles, ya que es el tercero en alcanzar al menos un nivel 4, con un 71,4%; y el último en el nivel 3, con un 85,7%. Aun así, creemos que la cantidad de tiempo empleada es un resultado importante a tener en cuenta.

El efecto pedagógico del uso del Pseudocódigo en idioma materno para el aprendizaje de la programación parece ser positivo. Una de las razones puede ser la menor carga cognitiva que tienen que soportar los estudiantes. Por ello, creemos que merece la pena estudiar con más profundidad este efecto ya que, por ahora, parece una forma apropiada de abordar el estudio de los contenidos de fundamentos de la programación en vez de empezar directamente con un lenguaje de programación de alto nivel como Java.

No pretendemos generalizar los resultados de este trabajo puesto que su diseño puede ser mejorado en algunos aspectos. Por ejemplo, grupo 2, grupo que obtuvo menor puntuación media, es claramente el más numeroso. Esto quiere decir que el profesor tuvo que atender a más estudiantes, dando una atención menos personalizada que en el resto de los grupos.

En definitiva, el Pseudocódigo parece ser una herramienta útil, que tendremos en cuenta y utilizaremos, profundizando en el estudio de su impacto en la enseñanza de la programación estructurada a los alumnos. Además, nos planteamos abordar otros aspectos como la transferencia de conocimientos en Pseudocódigo al aprender un lenguaje de alto nivel.

5. Trabajos futuros

Queremos seguir en la línea de este trabajo, poder repetir el estudio con un mayor número de estudiantes y ampliar la investigación a otros lenguajes de programación. Además, pretendemos encontrar una sintaxis de Pseudocódigo más natural, un nuevo juego de palabras clave que resulte más comprensible y tal vez pueda diferir de la sintaxis gramatical clásica.

Queremos realizar estudios sobre el aprendizaje de fundamentos básicos de programación en Pseudocódigo frente a Python, dado que ambos mantienen un nivel muy similar de simplicidad sintáctica y puede ser apropiado para nuestros propósitos. Finalmente, plantearemos el diseño y construcción de un entorno de programación único, para que el mismo sea usado tanto para los estudios de Pseudocódigo como de Python.

6. Agradecimientos

Este trabajo ha sido financiado por la Comunidad de Madrid, con el proyecto e-Madrid-CM (P2018/TCS-4307), también cofinanciado por los fondos estructurales (FSE y FEDER).

Referencias

[1] I-Jung Chen y Chi-Cheng Chang. *Teoría de Carga Cognitiva: Un Estudio Empírico sobre la Ansiedad y el Rendimiento en Tareas de Aprendizaje de Idiomas.* Electronic Journal of Research in Educational Psychology, 7(2), 729-746. 2009 (nº 18). ISSN: 1696-2095

[2] R. Faux. *Impact of preprogramming course curriculum on learning in the first programming course*. IEEE Transactions on Education, vol. 49, no. 1, pp. 11-15, Feb. 2006, doi: 10.1109/TE.2005.852593

[3] Theodora Koulouri, Stanislao Lauria, Robert D. Macredie. *Teaching Introductory Programming: a Quantitative Evaluation of Different Approaches*. ACM Transactions on Computing Education. Dic. 2014

[4] Jeroen J.G. van Merriënboer, Paul A. Kirschner y Liesbth Kester. *Taking the Load Off a Learner's Mind Instructional Design for Complex*. Open University of the Netherlands, Educational Psychologist. Mar. 2003

[5] Pablo Novara. PSeInt. Dic. 2004 http://pseint.sourceforge.net/

[6] Joaquina Palomar Lever, Sandra I. Montes de Oca Mayagoitia, Alma M. Polo Velázquez y Amparo Victorio Estrada. *Factores explicativos del rendimiento académico en hijos de inmigrantes mexicanos en Nueva York*. Psicología Educativa, vol. 22(2), pp. 125-133, 2016

[7] Anthony Robins, Janet Rountree y Nathan Rountree. *Learning and teaching programming: A review and discussion*. Computer Science Education. Vol. 13, 2, 137-172. 2003

[8] John M. Zelle. *Python as a first language*. Proceedings of 13th Annual Midwest Computer Conference. Vol. 2. Mar. 1999