Diseño y evaluación de actividades de aprendizaje para un curso de introducción a la programación

Pedro J. Clemente, Alberto Gómez, Julia González,
Roberto Rodríguez, Encarna Sosa
Departamento de Ingeniería de Sistemas Informáticos y Telemáticos
Universidad de Extremadura
Escuela Politécnica, Avda. de la Universidad s/n, 10071 Cáceres
{pjclemente, agomez, juliagon, rre, esosa}@unex.es

Resumen

Cuando se plantea una asignatura de introducción a la programación, además de seleccionar los contenidos y las competencias transversales que se desean alcanzar, es fundamental diseñar de manera adecuada las actividades de enseñanza para cada una de las fases del aprendizaje, así como los mecanismos de evaluación que nos permitan determinar si se han alcanzado los objetivos previstos y calificar al estudiante.

Se presenta una propuesta de recursos, actividades y evaluación que pueden utilizarse en cada fase del aprendizaje de un alumno en un primer curso de introducción a la programación.

1. Introducción

Hasta ahora, en la asignatura de introducción a la programación para ingenieros informáticos que impartimos, no se plantean explícitamente actividades de enseñanza/aprendizaje para cada una de las fases del aprendizaje de un estudiante, sino que aparecen mezcladas a lo largo del desarrollo de la asignatura y en la evaluación.

Al analizar los resultados, se observa que no todos los estudiantes alcanzan el mismo nivel de dominio de la materia. En una prueba final, es frecuente que los alumnos superen un test de conceptos básicos y trazas, pero no sepan resolver correctamente problemas. Eso indica que han adquirido un cierto nivel de comprensión de la materia , pero no el suficiente para alcanzar un objetivo fundamental.

Estamos rediseñando la asignatura de forma que la materia se presente atendiendo a distintas fases del aprendizaje, usando los recursos más útiles en cada fase, proponiendo las actividades más adecuadas para ayudar a desarrollar las competencias y empleando los métodos de evaluación más apropiados para verificar que se han alcanzado los objetivos. En muchos casos, se trata simplemente de reorganizar y hacer explícitas muchas de las actividades que hemos venido desarrollando.

2. Las fases del aprendizaje

Hay muchas teorías sobre los factores que influyen en el aprendizaje de la programación [1]. Parece claro que el estudiante va pasando por diversas fases hasta alcanzar un conocimiento profundo de la materia.

Tomando la dimensión cognitiva de la taxonomía de Bloom y la propuesta [2] como referencias, para el aprendizaje de los fundamentos de programación (independientemente del paradigma o del lenguaje que se presente), se deben diseñar actividades e instrumentos de evaluación adecuados que incidan en cada fase: motivación, conocimiento, comprensión, aplicación, análisis, síntesis y evaluación.

Aquí damos por supuesta una motivación inicial por la materia en los estudiantes de ingeniería informática que, desgraciadamente, es menos común de lo que sería deseable.

En los siguientes apartados se detalla, para cada una de las fases, los objetivos principales, los recursos, actividades y herramientas de evaluación que consideramos adecuados.

Fase 0: Motivación por los contenidos

Objetivos: Interesar al estudiante en los nuevos temas y mostrar su necesidad.

Actividades: Presentar un problema atractivo que no se puede resolver todavía o que es crítico (sobre todo para aspectos arduos como la complejidad o la especificación).

Fase 1: Conocimiento

Objetivos: terminología, conceptos básicos (estructuras de control y de datos, módulos, objetos, parámetros, etc.), clasificaciones, uso y aplicación, abstracciones.

Actividades: clases magistrales, estudio individual, búsqueda de información.

Recursos: libros, transparencias, aulas virtuales, referencias a páginas web.

Evaluación: test, definiciones, listas de conceptos. (Es fácil la autoevaluación.)

Fase 2: Comprensión

Objetivos: comprender el modelo de ejecución para los conceptos tratados (estructuras de control, paso de mensajes, algoritmos de ordenación, manipulación de estructuras de datos, etc.); problemas básicos.

Actividades: laboratorio, clases magistrales combinadas con ejemplos y resolución de problemas. Por ejemplo, se obtienen buenos resultados al explicar, de forma explícita, los papeles que pueden desempeñar las variables en los programas [3], un conocimiento básico de los programadores expertos y que no suele detallarse.

Recursos: ejemplos de problemas básicos resueltos, libros, transparencias, animaciones.

Evaluación: aplicar algoritmos sobre distintos valores, realizar trazas.

Fase 3: Aplicación

Objetivos: resolver nuevos problemas, variaciones de los problemas básicos, tanto en los algoritmos como en los dominios de aplicación. Es fundamental partir de una buena colección de problemas y aplicaciones básicos ya resueltos y de aplicaciones de ejemplo correctas.

Actividades: modificación de programas, resolución de problemas, laboratorios cerrados.

Recursos: colecciones de problemas básicos resueltos y propuestos.

Evaluación: resolución de problemas, añadir nuevas restricciones a problemas conocidos, modificación de algoritmos, completar programas.

Fase 4: Análisis

Objetivos: análisis de la complejidad, comparación de algoritmos, buscar la solución más adecuada y más eficiente para distintos problemas relacionados.

Actividades y recursos: como en la fase anterior. Además, normas, ejemplos y actividades relacionadas con las competencias de comunicación oral y escrita.

Evaluación: test, presentaciones e informes con estudio de soluciones posibles, ventajas e inconvenientes (interesante para desarrollar otras competencias transversales como la capacidad de expresión oral y escrita).

Fase 5: Síntesis

Objetivos: realización de un proyecto de programación completo, a partir de una especificación, con un estudio completo de las posibilidades, documentación completa, etc.

Actividades: podrán desarrollarse de manera individual o en grupo. El proyecto se divide en fases (definición de requisitos, diseño de la interfaz, justificación de la estructura de datos); tras cada una de ellas se produce un informe.

Recursos: ejemplos de proyectos resueltos.

Evaluación: Documentación y código del proyecto. Debe ser una parte importante de la calificación final, valorando la asimilación de los conceptos, su aplicación y combinación, etc. Se puede realizar una presentación escrita o, si es posible, también oral. Aquí pueden fomentarse competencias de trabajo en grupo.

Fase 6: Evaluación

Objetivos: la capacidad de juzgar un trabajo ya realizado (por uno mismo o por otros) debe fomentarse en cada una de las fases, proponiendo ejemplos y herramientas para comparar la bondad o eficiencia de cada aspecto concreto.

Evaluación: comparación del trabajo propio con otros.

3. Conclusiones

Creemos que fijar objetivos para cada fase del aprendizaje en cada tema, así como hacerlos explícitos para el estudiante, junto con el uso de las actividades, recursos e instrumentos de evaluación adecuados, son necesarios para mejorar el aprendizaje de los fundamentos de la programación de ordenadores.

Además, así se sistematiza la enseñanza, alejando del estudiante la idea de que se requieren ciertas habilidades innatas para programar.

Referencias

[1] Shaffer, S. *A brief overview of theories of learning to program.* http://www.ppig.org /newsletters/2005-11.html.

[2] Bofill, P., Miró, J. *Las fases del aprendizaje: un esquema para el análisis y diseño de actividades de enseñanza/aprendizaje*. Web Jenui 2007. http://jenui2007.unizar.es.

[3] Sajaniemi, J. *The Roles of Variables Home Page*. http://cs.joensuu.fi/~saja/var_roles/.