Experiencias Comparadas de la Asignatura de Introducción a la Programación en la Titulación de Geodesia y Cartografía

Adolfo Lozano Tello
Área de Lenguajes y Sistemas Informáticos
Dpto. de Informática - Univ. de Extremadura
10071 Cáceres
e-mail: alozano@unex.es

Laura Sebastiá Tarin
Dpto. Sist. Informáticos y Computación
Univ. Politécnica de Valencia
46071 Valencia
e-mail: lstarin@dsic.upv.es

Resumen

En esta ponencia se presenta un estudio comparativo sobre la asignatura de introducción a la programación que se imparte en el primer curso de la Ingeniería en Geodesia y Cartografía en dos universidades. En la Universidad Politécnica de Valencia esta asignatura se denomina "Informática Aplicada", y en la Universidad de Extremadura "Fundamentos y Lenguajes Informáticos". Se realiza un análisis de los temarios, del método de evaluación y de la utilidad de la materia enseñada para otras asignaturas de la titulación. El objetivo de este artículo es servir de referencia a docentes, que impartan alguna asignatura de introducción a la programación en una ingeniería no informática.

1. Introducción y Motivación

La Universidad Politécnica de Valencia imparte la titulación de "Ingeniero en Geodesia y Cartografía" desde el curso 1994/95. Durante este tiempo, la asignatura de "Informática Aplicada" (desde ahora IAP) ha sido troncal de primer cuatrimestre. Los alumnos provienen de diversas universidades: Barcelona, Vitoria, y de la propia titulación de "Ingeniero Técnico en Topografía" de Valencia. En la Universidad de Extremadura, la titulación de "Ingeniero en Geodesia y Cartografía" se ha ofertado como segundo año. El alumnado viene en su práctica totalidad del Centro Universitario de Mérida desde el que hace muchos años (1975) se imparte la titulación "Ingeniero Técnico en Topografía". La única asignatura en programación de carácter troncal que se imparte en la titulación superior es "Fundamentos y Lenguajes Informáticos" (desde ahora FLI).

El objetivo de ambas asignaturas es el aprendizaje de los conceptos fundamentales de programación, utilizando ejemplos y prácticas afines a la topografía. El temario busca proporcionar al alumno una base de programación, que le permita desarrollar programas para resolver ciertos problemas concretos de su práctica profesional. Actualmente existe una gran cantidad de paquetes informáticos en el mercado que atienden a la mayoría de las necesidades de todo tipo de profesionales. Sin embargo, en ocasiones, surgen problemas específicos que exigen una solución concreta. Para estos casos, una buena base en programación permite al ingeniero construir un programa que resuelva los problemas específicos planteados.

El contenido de los temarios de ambas asignaturas es muy similar, y es acorde a otras asignaturas de iniciación a la programación que se imparten en otras titulaciones y universidades. Debido a la idiosincrasia de la titulación (y de otras ingenierías similares), el tema de operaciones con variables matriciales y su almacenamiento en ficheros cobra gran importancia. Se utilizan varias sesiones discutiendo ejemplos en estos temas.

Aunque en las explicaciones en clase se intenta justificar la construcción adecuada de un programa para resolver un determinado problema, no se profundiza en temas de eficiencia de algoritmos. Pensamos que, para una titulación de informática, este asunto debe ser un concepto básico; pero, dado el escaso número de horas de la asignatura (dos horas de teoría y dos de prácticas por grupo), este punto adquiere un segundo plano.

Del mismo modo, estructuras de datos dinámicas básicas como pilas, colas, árboles, listas, etc. no se incluyen en el temario debido al

mismo motivo.

No se ha optado tampoco por la enseñanza de la programación orientada a objetos, a pesar de ser una alternativa de enseñanza de programación que usan varias universidades para la introducción a la programación [2, 3]. No hemos optado por esta alternativa porque pensamos que el fin primordial de esta asignatura de iniciación a la programación es que el alumno asimile el concepto de algoritmo, y que adquiera los fundamentos básicos para que en un futuro pueda enfrentarse a otros paradigmas más complejos de programación, quizás ampliando sus conocimientos en otras asignaturas avanzadas.

El artículo se distribuye de la siguiente forma: en la sección 2 se exponen los temarios impartidos en las dos asignaturas con los métodos de evaluación aplicados; en la sección 3 se discuten las ventajas y dificultades halladas en la aplicación del temario; algunas ideas aportadas por compañeros que imparten otras asignaturas en la titulación sobre realización de programas para sus prácticas, son brevemente indicadas en la sección 4. En la sección 5 se exponen las conclusiones de este trabajo.

2. Temarios y Métodos de Evaluación

En la Universidad Politécnica de Valencia, el lenguaje de programación escogido en la asignatura de IAP ha sido el lenguaje C por diversas razones:

- Se trata de un lenguaje de programación de ámbito general.
- Recoge la mayor parte de las características de la programación estructurada "tradicional".
- El compilador del lenguaje es de fácil acceso a los alumnos y no consume excesivos recursos, por lo que puede ejecutarse en cualquier ordenador, permitiendo a los alumnos realizar prácticas fuera del horario lectivo.
- El código fuente es portable y puede ser compilado en la mayoría de las plataformas *hardware*.

A continuación, se resume el temario que se imparte:

- Unidad temática 1: Introducción a la programación
  - TEMA 1: Introducción a los computadores
  - TEMA 2: Elementos básicos de la programación
- Unidad temática 2: Elementos básicos de un lenguaje de alto nivel
  - TEMA 3: El primer programa
  - TEMA 4: Entrada y salida
- Unidad temática 3: Estructuras de control
  - TEMA 5: Resolución de problemas. Organigramas
  - TEMA 6: Estructuras selectivas
  - TEMA 7: Estructuras repetitivas
  - TEMA 8: Funciones
- Unidad temática 4: Estructuras de datos compuestas
  - TEMA 9: Vectores
  - TEMA 10: Estructuras de datos
- Unidad temática 5: Aspectos avanzados
  - TEMA 12: Gestión de ficheros
  - TEMA 11: Variables dinámicas

Como se puede observar, se trata de unos contenidos similares a cualquier asignatura de introducción a la programación, aunque se hace hincapié en ciertos aspectos como el uso de vectores y matrices y el tratamiento de ficheros. Además, se dedica un tiempo a la resolución de problemas mediante organigramas, para que el alumno sea capaz de construir un algoritmo independientemente del lenguaje de programación utilizado. Las clases prácticas consisten en la resolución de ejercicios basados en problemas topográficos. Se trata de problemas no muy extensos, que puedan resolverse en una o dos sesiones de prácticas. Además, estas prácticas sirven como base a los ejercicios prácticos evaluados.

En cuanto a la evaluación, se ha optado por un criterio mixto. Por un lado, se proponen dos prácticas que los alumnos deben desarrollar por parejas y sobre las que después se les realiza una prueba: bien una pequeña modificación o bien unas preguntas. Por otro lado, se realiza un examen individual escrito en el que deben resolver, en primer lugar, unas cuestiones sobre aspectos básicos, y después se proponen unos ejercicios más complejos, para cuya resolución se les permite el uso de material (apuntes, libros, etc.). La evaluación final consiste en la media de la calificación obtenida en las prácticas y en el examen.

La asignatura de FLI de la Universidad de Extremadura persigue, al igual que la asignatura anterior, ofrecer la base para que los ingenieros en Geodesia y Cartografía conozcan los métodos de programación y algoritmia, y puedan aplicarlos en la resolución de problemas topográficos y cartográficos. Se ha optado por escoger un lenguaje de programación visual [1] para explicar los elementos básicos de un programa; añadiendo las peculiaridades de la programación bajo *Windows* orientada a eventos.

Como puede verse a continuación, el temario recoge los temas básicos de una asignatura de iniciación a la programación, incluyendo algunos específicos de la programación visual.

- ß TEMA 1: Conceptos básicos de los lenguajes de programación.
- ß TEMA 2: Lenguajes de programación visual.
- ß TEMA 3: Introducción al concepto de programa.
- ß TEMA 4: Variables, Operadores y Expresiones.
- ß TEMA 5: Sentencias de control de flujo.
- ß TEMA 6: Vectores.
- ß TEMA 7: Procedimientos.
- ß TEMA 8: Controles visuales.
- ß TEMA 9: Operaciones de entrada y salida en ficheros secuenciales.
- ß TEMA 10: Operaciones de acceso a bases de datos (nivel de introducción).
- ß TEMA 11: Desarrollo de un proyecto.

Se ha optado por explicar todos los contenidos teóricos usando la sintaxis del lenguaje *Microsoft Visual Basic*, y las clases prácticas se realizan también en este entorno. Para evitar que el alumno vincule el proceso de construcción de un algoritmo con el aprendizaje de un lenguaje concreto, la idea para el próximo curso es explicar los conceptos de programación en las clases teóricas en pseudocódigo, con la notación de los lenguajes orientados a eventos, y dar las clases prácticas en *Visual Basic* u otro similar.

A diferencia de la asignatura de la UPV, en esta asignatura no se realiza examen teórico. Para superar la asignatura, el alumno debe desarrollar una práctica más o menos compleja donde se aplican los conocimientos adquiridos en clase, relacionada con temas usualmente de topografía
. El método de examinar es el siguiente: delante del ordenador cada alumno debe añadir a su práctica una modificación de los requisitos indicados en el enunciado original, en un tiempo determinado. De esta forma se intenta evitar la entrega fraudulenta de prácticas que no hayan desarrollado los propios alumnos. Si el alumno logra superar la prueba, y la práctica contiene los requisitos indicados en el enunciado, obtiene una calificación de aprobado. Esta nota se incrementará proporcionalmente a las mejoras con nuevas opciones que el alumno añada a las especificaciones básicas indicadas en la práctica.

3. Comparativa Entre las Dos Asignaturas de Iniciación a la Programación

Como se ha podido ver en la sección anterior, ambas asignaturas ofrecen un temario con principios de programación básicos, orientados a una ingeniería de una titulación no informática. Se relegan temas importantes para un ingeniero de informática como son las estructuras de datos dinámicas, la complejidad de algoritmos o la programación orientada a objetos; como ya se ha comentado, debido al insuficiente número de horas de docencia necesarias para que el alumno pueda entender estos conceptos. En cambio, los dos temarios tienen como finalidad que el ingeniero pueda adquirir el conocimiento elemental para enfrentarse al desarrollo de algoritmos.

A pesar de estas similitudes, vamos a indicar las diferencias entre el contenido y la forma de evaluación entre ambas, y qué ventajas e inconvenientes vemos en cada una.

3.1. Diferencias de Contenido

La asignatura de IAP de la UPV sigue un temario tradicional de iniciación en programación. Los alumnos asimilan las ideas de programación estructurada básicas para enfrentarse a un problema computacional. De esta forma los procesos de creación del programa en un lenguaje fuente, la compilación y creación de un fichero ejecutable quedan perfectamente claros y diferenciados.

La utilización de un compilador en modo texto, provoca que el proceso de creación de interfaces de entrada y salida sea costoso y poco

agradable. Aunque la interacción con el usuario de un programa es una parte importante de éste, no hemos considerado que sea esencial para los alumnos en este punto de su aprendizaje. Además, el tipo de prácticas desarrolladas se ha basado fundamentalmente en cuestiones de cálculos topográficos que requieren poca interacción con el usuario. La finalidad del contenido topográfico de estas prácticas es que motiven al alumno, permitiéndole entender la utilidad de la programación para su desarrollo profesional. Entre estas prácticas, cabe destacar la realizada sobre conceptos de teledetección. Se proporcionó a los alumnos unas bibliotecas de funciones para el manejo de imágenes, unos programas de ejemplo y varias imágenes. Durante una sesión práctica, se explicaron todos estos elementos y se les propusieron varios ejercicios, que consistían en completar algunas funciones que realizaban un tratamiento sobre estas imágenes. De esta forma, se pone de manifiesto que estos compiladores no limitan, en principio, el campo de los ejercicios prácticos. Además, los alumnos pueden realizar las prácticas en cualquier ordenador, ya que los compiladores de C ocupan muy poco espacio.

Por otro lado, la asignatura de FLI de la UEX, aborda la iniciación a la programación bajo la idea del desarrollo de aplicaciones en *Windows*. De esta forma, se deben explicar los conceptos básicos de los programas en *Windows*: qué es un elemento de *Windows* (formularios, botones, cajas de texto, etiquetas,...), qué es una propiedad, qué es un evento, etc. A pesar de que los alumnos están familiarizados con el entorno *Windows*, deben asimilar la idea de que pueden programar una serie de acciones cuando un usuario efectúe un evento contra un objeto del programa. Este paradigma no es fácil de comprender, y se deben emplear varias sesiones para explicar estos conceptos.

Además, el desarrollo de programas utilizando entornos visuales (en este caso *Visual Basic*) tiene el inconveniente de necesitar un hardware adecuado, no siempre disponible personalmente por los alumnos. Por este motivo, algunos alumnos tienen que desarrollar sus prácticas exclusivamente en los laboratorios.

El porqué se ha empleado un lenguaje visual orientado a eventos como única forma de iniciación a la programación se puede justificar por los siguientes motivos:

- La mayoría de los ordenadores actuales tienen como entorno del SO una interfaz orientada a ventanas; la demanda de estos programas es creciente.
- La creación de las interfaces de programas se realiza de manera muy rápida mediante estos entornos de programación. Esta tarea es pesada en sistemas operativos basados en modo texto.
- La implementación de programas en entornos visuales *Windows* es asistida. De esta forma, cuando se está escribiendo una instrucción concreta, (o un objeto, propiedad, etc.), el intérprete muestra en un menú contextual las posibilidades para completar la instrucción, e indica de forma inmediata los errores directos de sintaxis. Los beneficios para programadores noveles son extraordinarios.
- Los compiladores visuales actuales poseen módulos de conexión a BD compatibles (ODBC). Aunque en el temario, el acceso a BD desde el lenguaje de programación se ve de forma muy elemental, en otra asignatura troncal de la titulación (“Bases de Datos”) se exige una práctica sencilla de acceso a bases de datos *Access* mediante *Visual Basic.*

Sin embargo, la mayor desventaja que se ha encontrado al utilizar un lenguaje visual, además del gasto de tiempo que se emplea en explicar los conceptos de la programación orientada a entornos, es que los alumnos nunca llegan a ver cómo se construye un programa en modo texto. En cierto sentido se deforma la idea de programa cuando se introduce el concepto de evento.

Aunque no se puede afirmar categóricamente (debido a la carencia de experiencia), se puede intuir que un alumno que haya aprendido a programar en un entorno visual orientado a eventos tendrá muchas dificultades para construir un programa totalmente secuencial en modo texto. Y nos podemos preguntar, ¿un alumno que haya aprendido a programar con un compilador de lenguajes secuenciales en modo texto, puede aprender fácilmente a programar con lenguajes orientados a eventos?.

La experiencia en nuestro caso con la asignatura de FLI nos lleva a decir que sí. La asignatura de FLI en el curso 99/00 se impartió usando el lenguaje C. Los alumnos tuvieron que desarrollar una práctica para aprobar la asignatura.

Algunos de ellos, a pesar de asistir a clase, y llevar desarrollada la práctica en gran parte, no pudieron presentarla en las fechas indicadas (recordemos que bastante alumnos trabajan y les surgen proyectos a mitad de curso). Este año 00/01 se ha usado como base de la asignatura el lenguaje *VisualBasic*. Los alumnos que aprendieron a programar en C, tras asistir a las clases de las definiciones de los elementos básicos de los lenguajes orientados a entornos, y a algunas clases prácticas de *VisualBasic*, desarrollaron la nueva práctica de manera muy rápida sin dificultades.

3.2. Diferencias en el Método de Evaluación

La forma de evaluación en FLI es únicamente mediante el desarrollo de una práctica. Los alumnos deben idear el diseño de su práctica siguiendo los requisitos marcados en un enunciado. Se ha optado por esta forma de evaluar para enfrentar a los alumnos a un problema real al que pueden incorporar los conocimientos teóricos aprendidos en clase. Para motivar al alumno, el tema de la práctica consiste en un problema relacionado con la topografía.

La forma de superar la asignatura es mediante un examen en el ordenador, que consiste en hacer una modificación a los requisitos marcados en el enunciado original. De esta forma, se controla a los alumnos que no han realizado la práctica por ellos mismos. Los alumnos que han desarrollado su práctica (se puede deducir por asistir a clases prácticas, seguimiento de tutorías, etc.) no encuentran mayor dificultad en hacer la modificación en el tiempo estipulado. Para subir nota, los alumnos incorporan nuevas opciones a las obligatorias indicadas en el enunciado.

Las dificultades más importantes que se han encontrado al realizar el examen de esta forma son que algunos alumnos, aun entendiendo la práctica entregada, no la han desarrollado ellos desde el principio. Han empleado la ayuda de algún compañero o amigo para desarrollar las partes más complejas y la estructura general y, superado esto, se han aprendido (eso si con un aceptable dominio) el funcionamiento y partes del programa entregado. Por otro lado, la entrega de una única práctica lleva a olvidar la asignatura hasta los últimos días, coincidiendo además con fechas cercanas a exámenes; lo que conlleva, en algunos casos, al abandono de la asignatura.

El modo de evaluación escogido en IAP permite una evaluación más continua que en FLI, ya que las prácticas se evalúan en dos momentos del curso: después de la unidad temática 3 y antes del examen final. De esta forma, es posible que el porcentaje de abandonos sea menor; ya que la primera práctica la realiza la gran mayoría de los alumnos, y por tanto, son más los que a final de curso hacen el esfuerzo de realizar la segunda. Al igual que en FLI, es necesario verificar mediante una prueba que las prácticas han sido realizadas por ambos miembros de la pareja, ya que, en ocasiones, en los trabajos en grupo, sólo uno de los miembros, el que se desenvuelve mejor con la asignatura, se ocupa de elaborarlo.

Estas prácticas evaluadas constituyen el 50% de la nota final. La otra mitad de esta nota se obtiene a partir de la calificación del examen final. Como ya se ha comentado, este examen consta de dos partes:

- La primera parte es un examen de una hora aproximadamente en la que se evalúan conocimientos básicos de programación: problemas cortos sobre vectores, bucles, etc., alguna traza con varias funciones para evaluar los conocimientos sobre paso de parámetros, desarrollo de una estructura de datos, etc.
- En la segunda parte se permite el uso de apuntes y consta de un problema en el que hay que implementar varias funciones relacionadas con él.

Los problemas de este examen suelen ser más complejos que los propuestos en las prácticas, ya que el aspecto fundamental que se evalúa es la capacidad de resolver un algoritmo; y no tanto la sintaxis del programa, que ya se ha tenido en cuenta en la parte de prácticas. De esta forma, es posible evaluar al alumno sobre aspectos más concretos de la programación.

A pesar de que este esquema de evaluación requiere de los alumnos un trabajo continuo durante el curso, quizás los alumnos puedan encontrarse más motivados por la realización de un proyecto al final del curso, como en la asignatura de FLI.

4. Aplicación de los Conocimientos de Programación en Otras Asignaturas

Tal y como ya hemos comentado, el objetivo fundamental de estas asignaturas es que los alumnos apliquen la programación a la resolución de problemas de su vida profesional. Para ello, el primer paso sería la utilización de los conceptos aprendidos en otras asignaturas de cursos superiores.

La idea es desarrollar programas para resolver los cálculos demandados en otras asignaturas, que hasta el momento se realizaban de manera manual; o proporcionar las librerías de funciones que los implementen construyendo programas para resolver problemas más concretos.

Tras consultar a compañeros profesores de la titulación, recogemos algunas ideas que a todos nos han parecido interesantes. De hecho, algunos nos han asegurado que van a aplicar los conocimientos en programación para que los alumnos desarrollen prácticas relacionadas con sus asignaturas. Hasta la fecha, algunas de estas prácticas, o no eran propuestas, o eran demandadas en alguna hoja de cálculo o similar. Estas son algunas propuestas de posibles aplicaciones de programación discutidas con los compañeros:

- En la asignatura de "Representación y reproducción cartográfica" orientada a la producción cartográfica, se podrían desarrollar programas de algoritmos de test para el control de la calidad posicional, y programas de algoritmos raster para generalización (tratamiento de imágenes).
- En "Fotogrametría terrestre", los programas de cálculos para calibrados de cámaras serían especialmente útiles.
- En "Aplicaciones GPS" se podrían desarrollar aplicaciones que realicen cálculos de correcciones troposféricas
- En "Sistemas de información geográficos" se pueden hacer infinidad de programas que exploten las bases de datos de los SIGs existentes.

5. Conclusiones

En este artículo, hemos presentado el temario y los criterios de evaluación de dos asignaturas de introducción a la programación: FLI, de la Universidad de Extremadura y IAP, de la Universidad Politécnica de Valencia. Hemos analizado las similitudes y las diferencias entre ambas asignaturas, destacando sus aspectos más eficaces y novedosos como, por ejemplo, el uso de un lenguaje visual en FLI. Por otro lado, hemos resumido cómo los conocimientos aprendidos pueden aplicarse a otras asignaturas de cursos superiores. Este trabajo puede ser utilizado por profesores que impartan (o vayan a hacerlo) asignaturas de introducción a la programación en ingenierías, en el que pueden comparar cuál es el modelo que le parece más adecuado, y las ventajas e inconvenientes de cada uno.

Se puede encontrar más información sobre FLI en http://webepcc.unex.es/~alozano/fundam/ e IAP en http://www.dsic.upv.es/users/ia/lstarin.

Referencias

[1] Lozano Tello, A. “Contenidos de la Asignatura Fundamentos y Lenguajes Informáticos y Algunas Experiencias en la Aplicación de Internet en su Docencia”, Jornadas de Enseñanzas Universitarias sobre Informática (JENUI’00) Alcalá de Henares, págs. 143-147.

[2] Marqués Hernández, F. y Prieto Sáez, N. "Curso Piloto de las asignaturas Introducción a la programación, Algoritmos y estructuras de datos I y II", Escuela Universitaria de Informática de la Universidad Politécnica de Valencia: http://www.dsic.upv.es/asignaturas/eui/ad2/cursopiloto.html

[3] Ortega A., y colegas, “Programación Multilenguaje con Component Pascal y Java en un 1º Curso de Programación”. Jornadas de Enseñanzas Universitarias sobre Informática (JENUI’99) La Almunia de Dª Godina,, págs. 343-348.