

# Supporting and Assessing First Year Programming: The Use of WebCT

**H.M. Sayers, M.A. Nicell, S.J. Hagan**

School of Computing and Intelligent Systems

Faculty of Informatics

University of Ulster

hm.sayers@ulster.ac.uk; ma.nicell@ulster.ac.uk; sj.hagan@ulster.ac.uk

## Content

- [1.Introduction](#)
- [2.Design](#)
- [3.The Evaluation](#)
- [3.1 Support Materials](#)
- [3.2 Assessment](#)
- [3.3 Comparison of performance 'with' and 'without' WebCT support](#)
- [3.4 General](#)
- [4. Conclusions and Future Work](#)
- [5. References](#)

## Abstract

Java is used to teach introductory programming to a large cohort enrolled on a variety of degree programmes presented on the University of Ulster's Magee Campus. There are two introductory programming modules taught in the first year through the traditional medium of lectures underpinned by tutorials and practical laboratory classes. Typically the class is composed of students ranging widely in ability, age, experience and motivation. Historically, students seemed to have great difficulty in achieving a reasonable standard within the twelve-week semester and each year their performance in programming presented a significant hurdle to their academic progression. In an effort to provide opportunities for guided self study and individual progress monitoring, an integrated suite of on-line resources was developed within a WebCT environment. This arrangement provided for stepped tutorials which mapped onto the lecture materials, informal (non-assessed) progress tests and formal (assessed) class tests. Upon completion, the assessed tests returned the score attained along with model solutions for those questions scored as incorrect, and the non-assessed tests returned an immediate response of correct or incorrect along with appropriate feedback. This paper presents the resources provided within the WebCT environment along with a study of students' experiences. The use of WebCT for coursework assessment was used in conjunction with the more traditional methods of laboratory tests and written tests ( a 50-

50 split). Further, a comparison of both coursework and examination results with those of a previous cohort of first year students who did not have access to the WebCT resources and who were assessed solely using traditional methods is presented. This comparison indicates that academic concerns that the adoption of on-line multiple-choice assessment would have a deleterious impact on performance within the more traditional time-limited, end-of-semester written examination were unfounded.

## 1. Introduction

Introductory computer programming is studied by approximately one hundred and fifty year-one students over the first semester. Of these, only around 40% are enrolled on mainstream computing degree courses. Teaching programming to this large and very diverse group in itself poses a unique challenge. This challenge is compounded, however, by a preconception within the student body that the topic is "hard" and "male oriented" leading to perhaps less than optimistic expectations of success even before the students begin their studies.

Undoubtedly, programming is best learned through practice - programming is a skill, not just a collection of information absorbed by the learner. The effectiveness of the traditional lecture as a platform for teaching programming is a subject of much debate and various novel attempts have been made to incorporate different methods into the traditional teaching environment ([Jenkins, 1998](#); [Jenkins 2001](#); [Wolfman, 2002](#); [nic Gearailt, 2002](#)). There have also been efforts to categorise students into different learning/ability groups prior to orchestrating specific teaching techniques and assessment strategies towards each depending on identified needs ([Davis et al., 2001](#); [Jenkins & Davy, 2000](#)). These efforts have enjoyed varying degrees of success, although their educational benefit is questioned as to whether they produce better results (i.e. an increased understanding of the subject) or merely increase the numbers of students who successfully pass the module. As class sizes continue to increase and the time students are able to reserve for study falls, the authors moved to maintain formal lectures as central to the delivery framework while expending considerable effort in the development of on-line study aids which can be accessed ad libitum both on and off campus.

The majority of students (85%) within the 2002/3 first year students (who formed the subjects of this work) had no previous programming experience and 93% of those who had experience had little or no confidence in their programming skills prior to undertaking the module. The students received three hours of lectures, two hours of practical sessions and one hour of tutorial each week. The practical sessions and the tutorials underpinned the material covered in the lectures. During supervised practical sessions, students were presented with staged programming problems which mapped to the material presented in the preceding lecture. The following tutorials afforded the opportunity to 'trouble shoot' difficulties encountered during the practical sessions.

The additional materials presented on-line attempted to address those topics which students reported as the most challenging aspects of the subject area in an earlier qualitative study designed to establish perceived difficulties and areas in need of further focus ([Sayers et al., 2003](#)). Arrays and parameter passing in methods were considered to be the most difficult topics to master, followed by iteration, selection and input/output. The class was unevenly divided by gender with two-thirds male, yet very little differences in the mean rating of difficulty were found between males and females.

The most common use of the World Wide Web (WWW) within a university teaching environment is as an information delivery tool - the WWW is a convenient way to distribute standard course and module materials such as assignments, lecture notes, laboratory exercises, course syllabi and other related materials. It has been noted, however, that it is not so much the technology itself but how it is used and how it serves to enhance the students' learning experience that determines its effectiveness ([Pilgrim, 2000](#)). To facilitate learning on-line, the University of Ulster has adopted a software application called WebCT, an integrated set of course/module management tools and educational tools which combine to facilitate learning, communication and collaboration among all stake holders in the delivery and assessment of a module. As is common with most computer-assisted assessment systems, testing is limited to multiple-choice questions (MCQs) because marking for free response answers cannot be easily automated. This raises issues such as the suitability of MCQs for assessing such a practical subject, as well as the design of the questions themselves to ensure that they address the appropriate learning outcomes. Daly (1999) divides an introductory computer programming course into two main areas: the syntax and semantics of the programming language, and program design. Based on such research and the authors' experiences, it was therefore felt appropriate to incorporate MCQ as an assessment technique for only part of the coursework element of the module (the syntax and semantics of the language), leaving scope for the assessment of program design and implementation to the more traditional assessment techniques. Bhalerao and Ward (2001) present an interesting solution in the form of a hybrid system combining MCQ testing with free response questions which are anonymously distributed among the learners themselves for peer assessment.

Research shows that the quality of learning correlates closely with teachers' skills in asking the right questions (Beyer, 1997), yet teachers often carry out this task without sufficient preliminary analysis of the nature of the questions, their relevance to a respective level of cognitive difficulty, or their contribution to the creation of better learning opportunities for students. As mentioned above, designing the on-line questions is an extremely important aspect of the success of the resource itself, and the questions for inclusion in this new resource were designed to address levels 1 to 3 of Bloom's taxonomy (Bloom & Krathwohl, 1984), namely knowledge, comprehension and application. Analysis, synthesis and evaluation (levels 3 to 6) were considered more suitable for assessment using traditional methods due to the restrictions of the software. The Use Case model of Booch et al (1999) was used as support to help with the creation of qualified questions based on

Bloom's taxonomy and to identify the functional options to be made available to students.

Providing students with self-assessment tests is one way of making them more aware of their own learning and enables them to monitor their own competence ([Carbone and Schendzielorz, 1997](#)). Students were therefore provided with self-assessment tests on each week's material and were additionally provided with support materials in two other areas: tutorials to support the lecture content, and worked examples to support both lecture and laboratory exercises. All materials are complementary to those used in the formal teaching environment, and, based on experiences documented from other on-line learning environments (for example, [Grey and Miles, 2002](#)), provide supplementary information in the form of different programming examples and scenarios.

The design and layout of the materials are presented in the next section, followed by a description of the evaluation techniques, results and statistical analyses, and finally the conclusions.

## 2. Design

The screenshot shows a Microsoft Internet Explorer browser window displaying the WebCT interface for 'Algorithmic programming I'. The browser title is 'Algorithmic programming I - WebCT 3.7.2a - Microsoft Internet Explorer'. The address bar shows the URL: [http://odl.ulst.ac.uk/SCRIPT/corn350m1\\_sern02\\_00/scripts/serve\\_home](http://odl.ulst.ac.uk/SCRIPT/corn350m1_sern02_00/scripts/serve_home). The WebCT interface includes a navigation menu on the left with options like 'Control Panel', 'Course Menu', and 'e-Learning Hub'. The main content area features a banner for 'Algorithmic Programming I' and a list of links: 'Assessments', 'Tutorials', 'Worked Examples', 'Practice Tests', 'Student Progress', and 'Student Grades'. The page is dedicated to providing quality support materials for module COM158.

Figure 1: The Opening Page

The opening page of the developed environment presents the students with links to all of the available resources (Figure 1). There are six links provided:

- Assessments -this link provided access to the formal on-line assessments for the module. Access was restricted to prescribed times on given days. The link was opened for a limited period (an equivalent scenario to a time-limited examination).
- Tutorials -this link presented more in-depth coverage of weekly lecture materials with further references and worked examples. Figure 2 below presents a typical introductory tutorial on 'Arrays'.
- Worked Examples -this link provided access to further programming problems based on each week's lecture and practical materials, with worked solutions demonstrated step-by-step. More challenging problems are also presented for those "Rocket Scientists" ([Jenkins, 2000](#)) or "Space Cadets" ([Davis et al., 2001](#)) who were perhaps finding the module too easy. Figure 3 below presents a worked example on the design and scripting of 'Methods'.
- Practice Tests -this link presented students with a variety of non-assessed tests which they could use as a means of gauging their own knowledge and progress. These can be undertaken at will and repeated as many times as required. Figure 4 below presents a sample of the practice test questions for 'Methods'.
- Student Progress -this link allowed students to monitor their progress through the content of the on-line resources.
- Student Grades -this link allowed students to view the results of the formal, on-line graded assessments for the module.

The screenshot shows a Microsoft Internet Explorer browser window displaying a WebCT course page. The browser title is "Algorithmic programming I - WebCT 3.7.2a - Microsoft Internet Explorer". The address bar shows the URL: [http://odl.ltsn.ac.uk/SCRIPT/cont158ent\\_sen102\\_03/scripts/serve\\_home](http://odl.ltsn.ac.uk/SCRIPT/cont158ent_sen102_03/scripts/serve_home). The page header includes navigation links: MYWEBCT | RESUME COURSE | COURSE MAP | RESOURCES | LOGOUT | HELP. Below the header, there are tabs for "Arrays" (selected), "View", and "Designer Options". The main content area is titled "Algorithmic programming I" and includes a breadcrumb trail: Home » Tutorials » Arrays. An "ACTION MENU" contains links for Previous, Next, Contents, Retrace, and Refresh. The main section is titled "Arrays" and contains the following text:

- Java has a built in array class which you can use to hold multiple values so long as those values are of the same data type.
- However, in Java, arrays are much more restrictive. For example, you may not change the size of an array once you have created it.
- Java provides a zero-based array that is defined much as variables are. You first declare what type of data will be stored in an array, give the array a name, and then define how large it is. Consider the following example:

```
int[] intArray = new int[50];
```

- This array would hold 50 ints numbered from 0-49. All array elements created in this manner receive a default initial value. Primitive data types receive an initial value of zero. Elements of type **boolean** receive an initial value of **false**.
- Having instantiated an array in Java, you can access the elements of the array using index syntax as shown below:

```
int Array [5] = 6;
```

- Java also allows you to define an array at the time of initialization such as in the following example:

```
int[] intArray = {0,1,2,3,4};
```

- Filling and extracting values from an array is carried out as follows:

```
int[] intArray= new int[50];
for(int i = 0; i < 50; i++){//for loop to fill the array
```

The left sidebar contains a "Control Panel" with options like "Take Guided Tour", "View Designer Map", "Add Page or Tool", "Manage Files", "Manage Course", "Change Settings", and "Content Assistant". Below that is a "Course Menu" with links to "Homepage", "e-Learning Hub", "Assessments", "Tutorials", "Worked Examples", "Practice Tests", "Student Progress", and "Student Grades".

Figure 2: The 'Arrays' Tutorial.

The screenshot shows a web browser window titled "Algorithmic programming I - WebCT 3.7.2a - Microsoft Internet Explorer". The address bar shows the URL: [http://odl.ulst.ac.uk/SCRIPT/cont158en1\\_sen102\\_03/scripts/serve\\_home](http://odl.ulst.ac.uk/SCRIPT/cont158en1_sen102_03/scripts/serve_home). The page content includes a navigation menu on the left, a main header with "WebCT" and "Example 9 Methods: View Designer Options", and a section titled "Algorithmic programming I". The section contains a "Worked Example 9" titled "Learning to program from scratch" with a "Methods" sub-section. The text describes a scenario where a company buys items in two sizes (Large: £24.99, Small: £19.99) and asks the user to write methods for calculating net cost, postage, and VAT. The VAT formula is given as  $VAT = ((net\ cost + postage) * 17.5) / 100$ . The page also includes an "ACTION MENU" with links for Previous, Next, Contents, Retrace, and Refresh.

WebCT  
Hide Navigation

Control Panel  
Visible to Designers  
Take Guided Tour  
View Designer Map  
Add Page or Tool  
Manage Files  
Manage Course  
Change Settings  
Content Assistant

Course Menu  
Homepage  
e-Learning Hub  
Assessments  
Tutorials  
Worked Examples  
Practice Tests  
Student Progress  
Student Grades

MYWEBCT | RESUME COURSE | COURSE MAP | RESOURCES | LOG-OUT | HELP

Example 9 Methods:  View  Designer Options

**Algorithmic programming I**  
Home » Tutorials » Arrays » Worked Examples » Example 9 Methods

ACTION MENU: [Previous](#) [Next](#) [Contents](#) [Retrace](#) [Refresh](#)

**Worked Example 9**  
**Learning to program from scratch**  
**Methods**

Scenario:  
A certain company buys items in two sizes at the following net prices:  
Large: £24.99  
Small: £19.99  
Assume that these figures are constant variables.

(a) Write a local static method, which accepts as parameters the numbers of each product purchased and returns the net cost. This method should be called **cost**.

(b) The company is charged postage for each item that it purchases. The postage charge is £2.00 for a small item and £2.50 for a large item. These figures will be constant variables. Write a static method, which accepts as parameters the numbers of each product purchased and returns the total postage cost. This method should be called **postage**.

(c) VAT is charged on both net cost and postage at 17.5%. VAT is therefore calculated using the formula:  $VAT = ((net\ cost + postage) * 17.5) / 100$ .  
Write a main method which:

- prompts for and reads in from the keyboard the number of each product purchased
- calculates and displays (by calling the methods **cost** and **postage** above) the net cost and postage cost, followed by the total cost and the VAT to be paid.

This scenario is broken down into sections to make the writing of the methods easier. Let

Worked Examples: Page 9/9

start | Inbox - Micro... | Java - Micro... | portfoli2DUB... | SPSS Mana... | Algorithmic pr... | Paint Shop Pro... | 13:09

Figure 3: A worked example for 'Methods'.

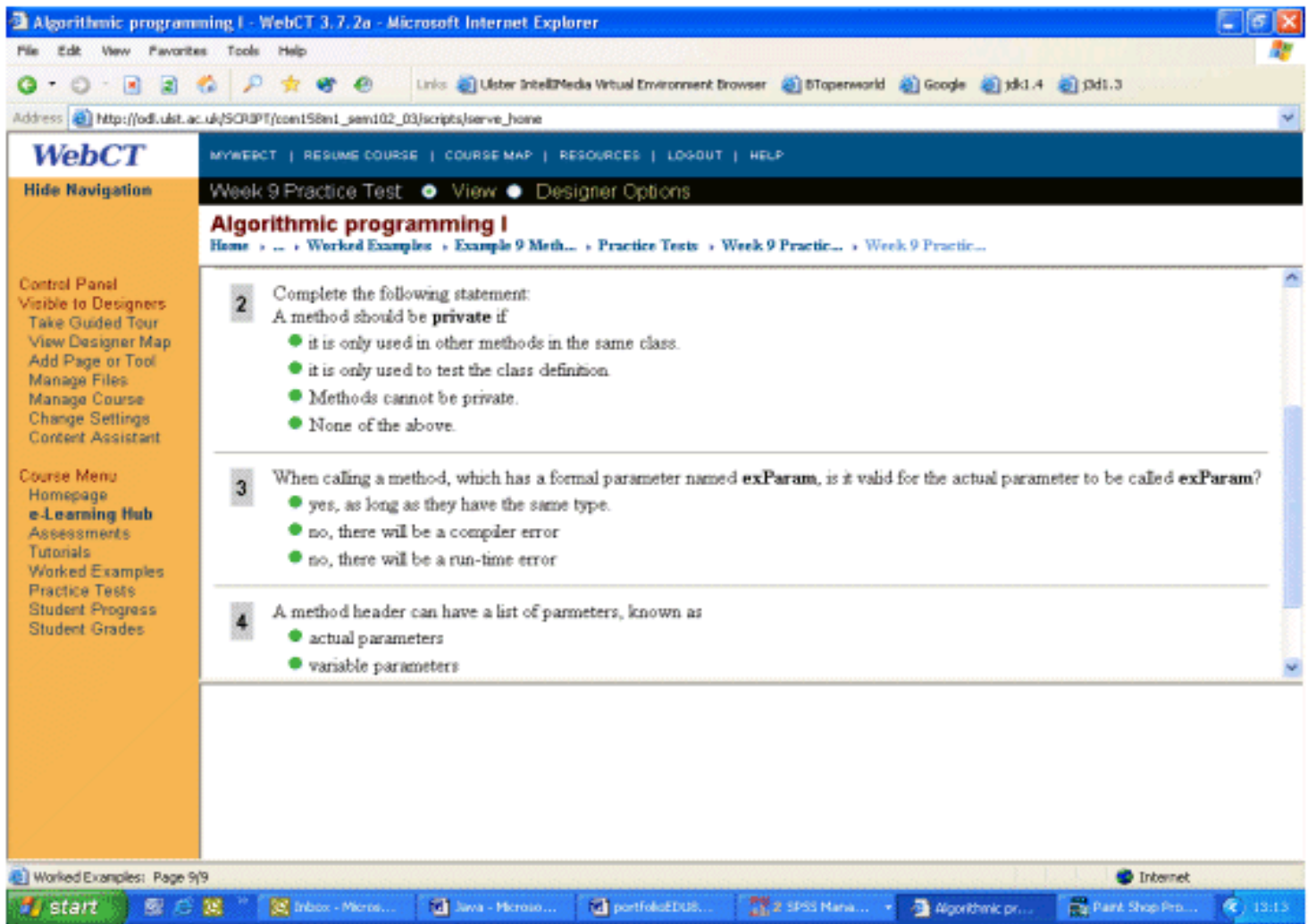


Figure 4: A practice test for 'Methods'.

### 3. The Evaluation

The students generally seemed very receptive to the provision of WebCT resources and many commented on its usefulness. It was decided to evaluate its effectiveness formally by distributing questionnaires at the end of the module. The aim was to discover general opinions about the environment as well as the materials provided, in particular to ascertain which aspects of the course materials thus presented were found most helpful as a study aid and why. Were the materials provided for arrays and methods, for example, useful in facilitating an understanding of these reportedly more difficult topics?

The questionnaire was divided into three sections: "support materials", "assessments" and "general". The first section, "support materials", presented questions on the usefulness of each of the support materials provided within the environment, namely tutorials, worked examples and practice tests. Questions were also asked on the usefulness of the materials according to topic. The second section, "assessments", presented questions on formal assessments using WebCT, and the

feedback provided. Finally, the "general" section presented questions on ease of use of the software along with some more open questions to ascertain which aspects of the learning environment students liked best and least and why, as well as providing an opportunity to make suggestions for improvements. Students were finally asked if they would like to see similar support materials for other modules.

### 3.1 Support Materials

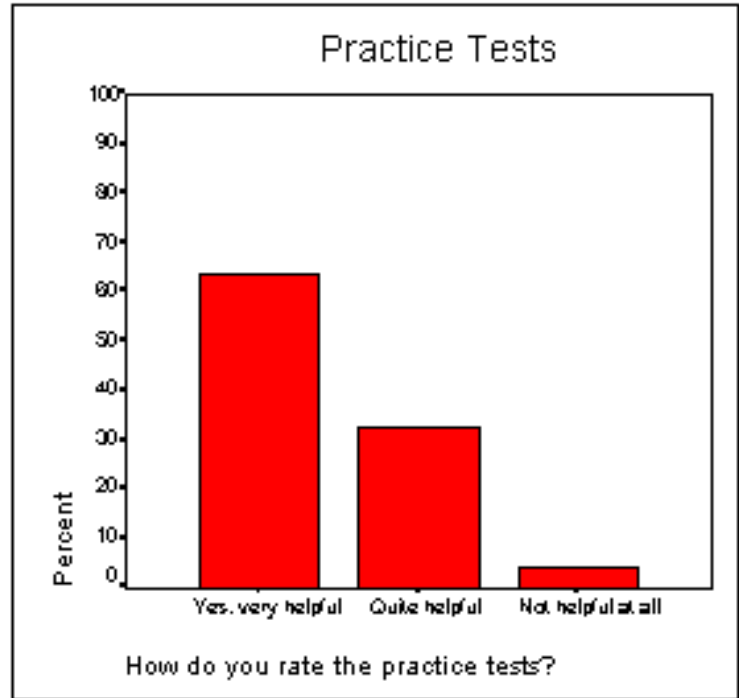
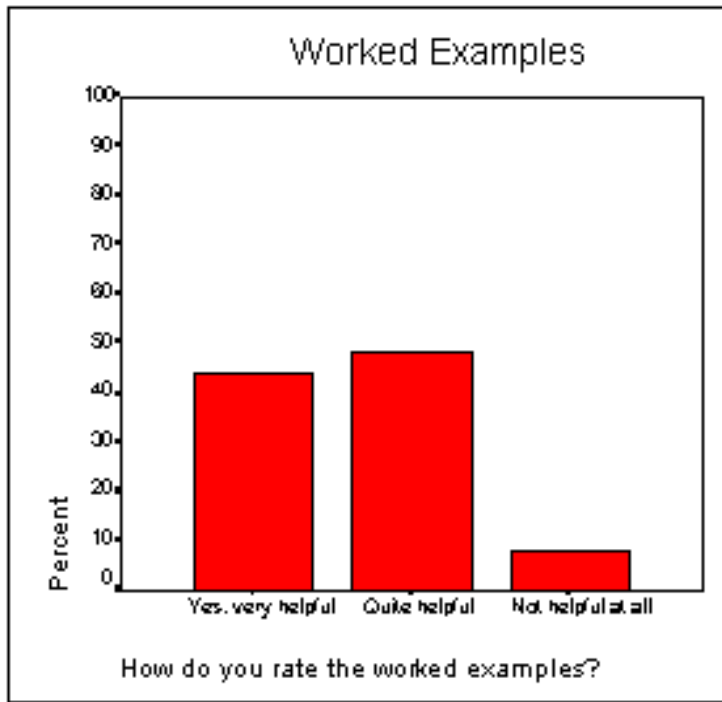


Figure 5: Worked Examples

Figure 6: Practice Tests

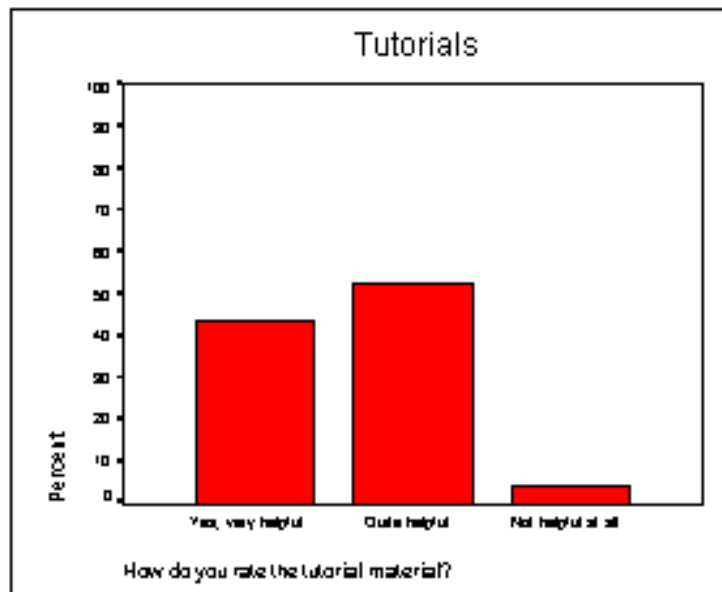


Figure 7: Tutorials

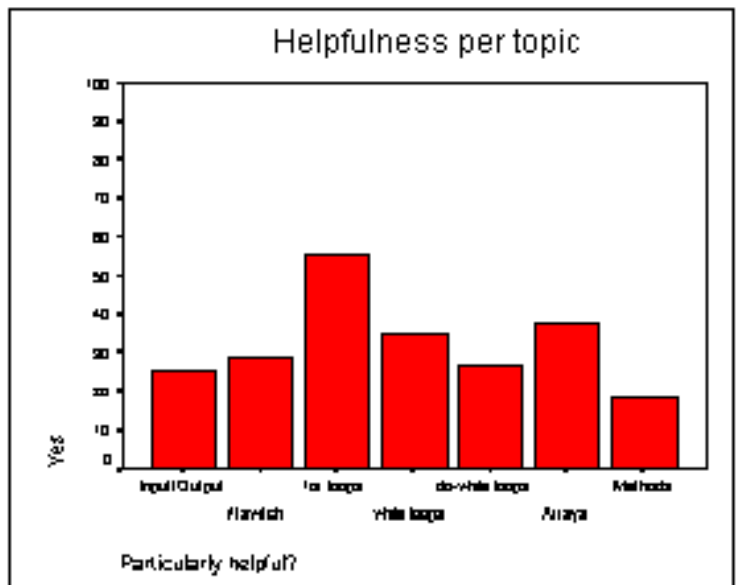


Figure 8: Helpfulness

Figures 5, 6 and 7 above show, from the 77 respondents, the percentages who rated the Worked Examples, Practice Tests and Tutorials either "very helpful", "quite helpful" or "not helpful at all". The results show that the highest percentage of students rated the Practice Tests as the most helpful. Participants were given the opportunity to explain why this was the case and the main reason given was their usefulness in helping students to assess their own knowledge and the practice provided for the formal coursework assessments. It is encouraging to note that all three areas were mainly considered either very helpful or quite helpful. A small percentage rated the materials not helpful at all but failed to provide a reason. This makes it difficult for us, as developers, to possibly improve the materials for their benefit.

Since arrays and methods had been identified by an earlier programming group as the most difficult topics within this module, followed closely by iteration, students were asked to state whether or not they found the support materials particularly helpful for each specific topic. Figure 8 above displays the percentage of positive responses per topic. The results show that the additional materials on iteration were considered particularly helpful, more so than the material on arrays. The material on methods was deemed not to be particularly helpful. This results suggests that either methods are found so difficult that many of the students cannot master the support materials, or that the support materials themselves need to be revised and edited to address the students' needs in a more effective way. Further analysis and refinement of this is required.

### **3.2 Assessment**

Students were asked to rate WebCT as a means of assessing coursework, and were asked whether or not they considered sufficient feedback on completed coursework to have been supplied. Figure 9 shows that the majority of the students were satisfied with on-line assessment, and results certainly show a well-balanced distribution of marks. Students were also generally satisfied with the feedback provided (see results in Figure 10). This was as expected since the assessments were designed to provide full explanation in the case of incorrect answers.

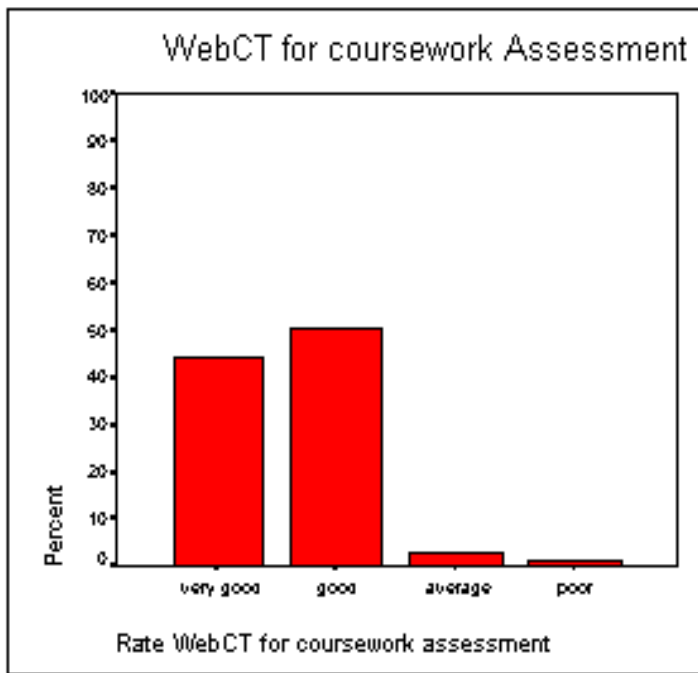


Figure 9: Coursework Assessment

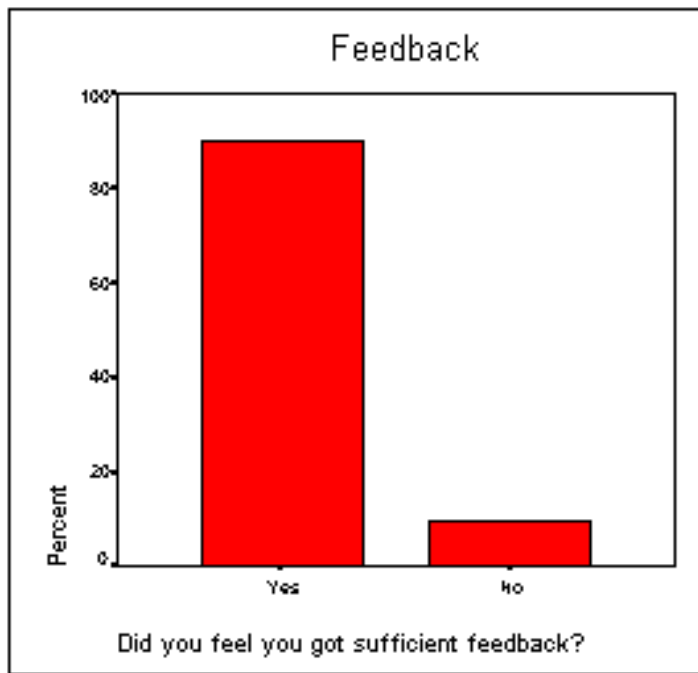


Figure 10: Feedback

### 3.3 Comparison of performance 'with' and 'without' WebCT support

Previous to this study, the coursework component of the module had been assessed by one end-of-semester, time-limited, written class test (conducted under formal examination conditions) and 'scoring' of numerous small Java programmes (submitted as source code solutions to posed problems). Although the knowledge and expertise of an individual student could be gauged from their performance in the class test, it proved more difficult to assess an individual's competence from the other assessed elements. The authors modified the assessment regime to include the written class test (as before) and two on-line, multiple choice tests delivered via the WebCT environment (conducted under formal examination conditions within the computing laboratory). Each of these assessment episodes returned test scores and feedback within a few seconds of the candidate having concluded the assessment. This change in format necessarily limited both the 'size' and complexity of the tasks which could be set for the students to complete and effectively removed any opportunity for discussion and cooperation within their peer group in deriving solutions to a common problem. The authors were keen to examine if this would impact on the scores attained within both the continuous assessment and in the end-of-semester examination. Toward this end the performance of students undertaking the module supplemented by the WebCT environment was compared with that of the cohort from the previous year for whom this support was not in place. Profile statistics for both groups are presented in Table 1 below.

	Without WebCT support (2001/2002 enrolment)	With WebCT support (2002/2003 enrolment)
Number in Group	96	131
Continuous Assessment		
Mean	61.0%	51.6%
S.D.	12.6	17.3
Min. score	30.0%	21.0%
Max. score	87.0%	92.0%
Final Examination		
Mean	44.8%	47.1%
S.D.	23.5	23.8
Min. score	6.0%	2.0%
Max. score	95.0%	98.0%

*Table 1: Statistical profile of assessment and examination performance with and without WebCT support.*

Despite the large standard deviations reported within Table 1 (reflecting the variability within the student scores) the returned means do suggest that with the support of the WebCT environment candidates on average obtained lower scores in the continuously assessed element of the module but performed slightly better in the final, written examination. A more formal analysis of student performance with and without the support of the WebCT environment was undertaken via a Mann-Whitney U Test of the continuous assessment scores and the examination scores of the two cohorts. Tables 2 and 3 present the results obtained.

	Class	N	Mean rank	Sum of Ranks
Exam	2001-2002	96	109.2	10483.0
	2002-2003	131	117.5	15395.0
	Total	227		
Coursework	2001-2002	96	137.4	13190.0
	2002-2003	131	96.9	12688.0
	Total	227		

*Table 2: Mann-Whitney Test ranks*

	Examination	Coursework
Mann-Whitney U	5827.0	4042.0
Asymp. Sig. (2 tailed)	0.345	0.000

*Table 3: Mann-Whitney U Test statistics*

Performance within the final written examination did not appear to differ significantly (2-tailed  $p=0.345$ ), thus allaying concerns that the increased deployment of multiple choice tests as the vehicle for continuous assessment would prove detrimental to the candidate's overall ability to write and analyse Java code, which remains the focus of the written examination component.

Performance in coursework was however found to be significantly different across the two cohorts (i.e. on average, students did attain lower scores when undertaking multiple-choice tests within the WebCT environment). This could be a consequence of a number of factors - many authors express the concern that continuous assessments are often subject to plagiarism (for examples, see [Daly & Waldron, 2002](#); [Woit and Mason, 2003](#)), and other contributing factors might include examination stress and the imposed time constraints. Perhaps, however, we now have a more accurate picture of the programming ability of the students who were assessed using WebCT, since it has been found in studies of non on-line assessment strategies that there are often significant inconsistencies in the actual programming skills of some students and their coursework scores, with high scores related to students with weak programming skills ([Woit and Mason, 2003](#)).

### 3.4 General

92% of the 77 respondents found the WebCT environment easy to use (Figure 11) and 99% stated that they would like to see similar resources available for other modules on their course (Figure 12).

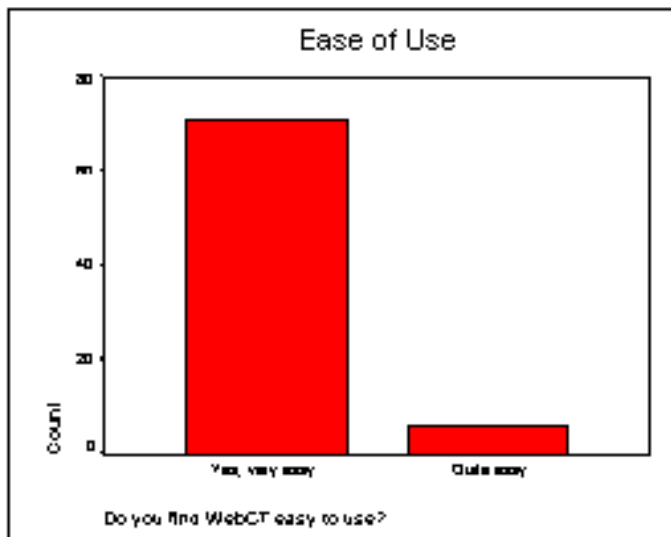


Figure 11: Ease of Use

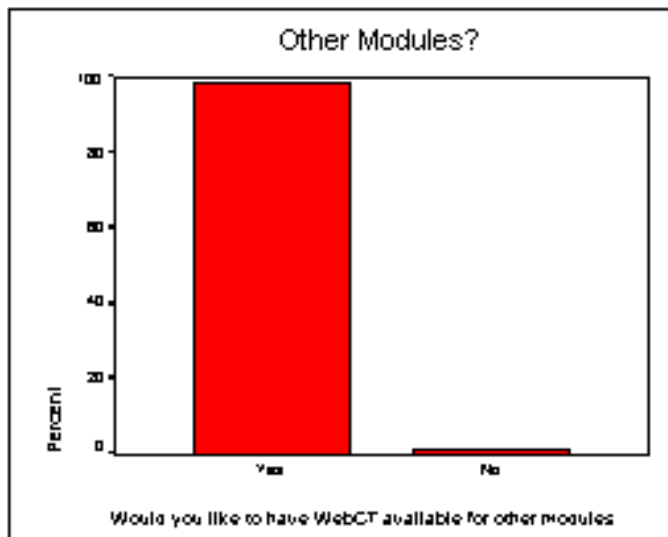


Figure 12: Availability on other modules

The responses to the aspects of the support environment liked most and least correspond with the ratings shown in the figures above. Most students liked the Practice Tests best, reinforcing the view that students are primarily motivated by assessment within a module of study. Reasons given point to this fact - most commented on the fact that they were as named - "practice" for the on-line tests. Tutorials were liked least by respondents with reasons such as topics already covered in lectures" and "I find lecture notes sufficient" common. Worked Examples received some worthwhile comments that they added depth to further other than those done laboratory sessions.

## 4. Conclusions and Future Work

Awareness of the great diversity of backgrounds, ages, abilities, motivation etc. within a first year, semester one programming class led to the design, implementation and presentation of online support materials using WebCT. Structured examples and exercises were presented in what was hoped to be a meaningful and effective way in order to encourage students to *think* about the concepts included in the module content. The aim of this study was to gather preliminary information about how students rate the WebCT resources offered and how their results compared with the previous year's cohort of students. The authors feared that the use of on-line assessment might have an adverse effect on the students' end-of-semester examination results. However, not only were the evaluation findings very encouraging but fears of a detrimental effect on end-of-semester results proved unfounded. Since a combination of on-line and traditional assessments were used for the coursework element of the module, and the traditional end-of-semester examination (used to assess the 'skill' of programme writing) produced slightly better results than those of the previous year, we can also claim that the new coursework regime did not have a negative effect on the cohort's programme writing skills.

The design of the support materials has been presented here along with evaluation results derived from the students, and a comparison of both the coursework and examination results with the

2001/2 cohort's results. The fact that only 77 of the 150 students participated in the evaluation is acknowledged and it is recognised that this detracts from the strength of the findings; however, the evaluation has shown that the vast majority of students who did participate found the on-line resources helpful and were in favour of on-line assessment. It also highlighted the fact that methods (the call to methods and parameter passing) remain topics of difficulty and the intention is that this material will be further enriched and expanded for the next academic year. The statistical comparison of results has revealed that WebCT has proved to be a successful means of coursework assessment which has, in this comparison, slightly reduced the average coursework marks achieved by students but has had no adverse effects on their examination performances. Further work on the development and refinement of the WebCT support materials and further comparisons with other cohorts (assessed on-line) are planned for the next academic year.

## 5. References

- Beyer, B.K. (1997) *Improving Student Thinking: a Comprehensive Approach*. Allyn and Bacon, Boston
- Bhalerao, A & Ward, A. (2001) Towards Electronically Assisted Peer Assessment: A Case Study, *ALT-J*, Vol.9, No.1, 26-37
- Bloom, B.S. & Krathwohl, D.R. (1984) *Taxonomy of Educational Objectives, Handbook 1: Cognitive Domain*. Addison-Wesley, Massachusetts
- Booch, G., Jacobson, I. and Rumbaugh, J. (1999) *The Unified Modeling Language User Guide*, Reading, MA, Addison-Wesley
- Carbone, A. & Schendzielorz, P (1997) Developing and Integrating a Web-based Quiz into the Curriculum, *Proceedings of WebNet 97, World Conference of the WWW, Internet & Intranet*, Toronto, Canada; November 1-5, 1997
- Daly, C (1999) RoboProf and an introductory computer programming course, Annual Joint Conference Integrating Technology into Computer Science Education, *Proceedings of the 4<sup>th</sup> annual SIGCSE/SIGCUE ITiCSE conference on Innovation and Technology in Computer Science Education*, Poland, 155-158
- Daly, C. and Waldron, J., (2002) Introductory Programming, Problem Solving and Computer Assisted Assessment, *6th Annual CAA Conference*
- Davis, H.C., Carr, L., Cooke, E., & White, S., (2001) Managing Diversity: Experiences Teaching Programming Principles, *LTSN-ICS 2<sup>nd</sup> Annual Conference 2001*, University of North London

- Grey, D. and Miles, R. (2002) Teaching Java Objectively - Reflections on a Web-based Java Course, *Proceedings of the Sixth Java and the Internet in the Computing Curriculum (JICC) Conference*, University of North London, Jan. 2002
- Jenkins, T. (1998) A Participative Approach to Teaching Programming, *Integrating Technology into Computer Science Education*, Dublin City University, 1998, ACM Press, 125-129
- Jenkins, T. (2001) Teaching Programming - A Journey from Teacher to Motivator, *LTSN-ICS 2<sup>nd</sup> Annual Conference 2001*, University of North London
- Jenkins, T., & Davy, J., (2000) Dealing With Diversity in Introductory Programming, *LTSN-ICS 1<sup>st</sup> Annual Conference 2000*, Heriot-Watt University, Edinburgh
- nic Gearailt, A. (2002) Using Java to increase Active Learning in Programming Courses, *Principles and Practice of Programming in Java*, Trinity College Dublin, June, 2002
- Pilgrim, M-J. (2000) Using Web Technology to Enhance the Traditional Classroom at Trent: Developing a Support Framework to Facilitate the Implementation of WebCT, <http://www.trentu.ca/~mpilgrim/oise1606>
- Sayers, H.M., Nicell, M.A., and Hagan, S.J. (2003) Teaching Java Programming: Determining the Needs of First Year Students, to be presented at *4<sup>th</sup> Annual LTSN-ICS Conference, NUI, Galway*
- Woit, D.Dr. and Mason, D.Dr., (2003) Effectiveness of Online Assessment, *SIGCSE'03*, February 19-23, 2003, Reno, Nevada, USA
- Wolfman, S., (2002) Making Lemonade: Exploring the Bright Side of Large Lecture Classes, *SIGCSE'02*