Engaging Undergraduate Programming Students: Experiences using Lego Mindstorms NXT

Collette Gavan
Edge Hill University
St Helens Road
Ormskirk, Lancashire, L39 4QP, UK.
(+44)1695 657633
gavanc@edgehill.ac.uk

Dr Mark Anderson
Edge Hill University
St Helens Road
Ormskirk, Lancashire, L39 4QP, UK.
(+44)1695 657634
mark.anderson@edgehill.ac.uk

ABSTRACT
In this paper, the experiences gained in adopting an approach to first year undergraduate programming classes which attempts to engage the students in problem-solving and teamworking activities are discussed. Both have a significant role to contribute in the development of employability skills. The approach taken makes use of Alice to introduce programming concepts and Lego Mindstorms NXT kits to develop Java programming skills. The module assessments combine individual and team-based components, encouraging the students to engage with their peers in order to solve the challenges they are set. This paper reports on the results achieved by the students on the module over a four year period which correlates to the introduction of the changes to the module design. The paper also considers the views of the students gathered from anonymous module evaluation forms.

Categories and Subject Descriptors
K.3.2 [**Computer and Information Science Education**]: Computer science education

General Terms
Design, Experimentation, Human Factors, Theory.

Keywords
Programming, student engagement, teaching.

1. INTRODUCTION
The teaching of programming to first year undergraduate students has, for a long time, been a problematic area [8]. Students often struggle with both the conceptual ideas along with the syntax of the language being taught [1,20,19]. In many respects, the move towards teaching using an object-oriented language, such as Java or C++, has served to amplify these issues [19,24]. Unlike procedural programming, in which the students would concentrate on the development of algorithms and syntactic correctness of the code that they are being asked to develop, the object-oriented paradigm introduces further complexity in its level of abstraction. Following this, students are required to model the relevant features of a "world" which their program is to implement. Whilst it is possible to teach procedural programming to students in the first instance and then move on to the object-oriented paradigm at a later point in a programme [19], the students may then face confusion as the methods they follow to design and implement their code will change.

In an attempt to address this barrier to learning, there have been a number of approaches which have been adopted to assist student understanding of programming concepts. Some approaches have focused on the development of specialized programming environments in which students develop their code [3,10,16]. Alongside assisting students to develop code, often by providing visualisation which demonstrates the structure of the program which has been developed (an example of this is BlueJ to teach Java programming), the creation of a specialized teaching environment removes some of the complexity of a standard Integrated Development Environment (IDE) from novice programmers. The outcome of this is that students still need to encounter these IDEs at a later point in their academic studies.

However, in conjunction with the development of the practical skills associated with the teaching of programming there is also an increased focus to develop employability skills amongst students [6,11]. Whilst the employment figures for graduates may be high, there is no indication whether graduates are fully benefitting from the abilities that they have developed through their degree programmes. Indeed, there have been predictions that graduates may face periods of unemployment, or of employment in positions which are not relevant to their chosen degree paths [21]. However, the issues involved when developing work-related practices within degree programmes are highly complex [13,25].

This paper presents an approach taken to teaching introductory programming to first year undergraduate students and reports on the experiences from the module. The module has been designed not only to engage the students with the underpinning concepts of programming, but also to develop both an understanding of the object-oriented paradigm and assist the students in developing their employability skills in terms of problem-solving and team working. The module design has evolved over the previous four years, and the paper reflects upon the changes that have been introduced and the impact which these have had upon the student experience.

2. EMPLOYABILITY SKILLS
There is a well held view that, alongside the teaching of academic subjects, the "focus for university is to...[prepare] students for life beyond the disciplines" [12]. A traditional view presented by

SIGITE'12, October 11–13, 2012, Calgary, Alberta, Canada.

Barrie (2006) identifies the key attributes a graduate would possess [2]. The focus of these abilities relates to the academic values and concepts that have been long-established within the higher education sector and include academic inquiry, academic rigour, the generation of ideas, and the development of learning skills. It could be argued, however, that these attributes would define a highly generic skill set. However, it has also been argued that these skills form the basis of the skill set sought by employers [22]. A question would then arise concerning the relevance of the skills developed through a degree programme to the careers of those who graduate from those programme. A degree may deliver the high level skills to satisfy these attributes, but may not meet the more specific requirements of the potential employers.

A key distinction of experiential learning as opposed to other learning theories is that it is based on an ongoing process of learning [9]; the forming and reforming of concepts through practice and experience. In this respect there is a close alignment with the HE sector reforms which also focus on education [22]. The importance of experiential learning was echoed by Winter and Maisch (1996) when reporting upon the experiences from the Accreditation and Support for Specified Expertise and Training (ASSET) programme [23]. This programme made use of a competence-based model of education for students in two subject areas, and found that the students were able to develop levels of professional understanding through the experience which they gained whilst on the programme. Furthermore experiential learning has been advocated as an appropriate means of teaching problem-solving to students [17]. It has been argued that experiential learning, through the implementation of Problem Based Learning (PBL), should be a pedagogy which is adopted centrally to the higher education curricula [18].

Instilling experiential learning will commence the ongoing process for graduates to enter the workplace and continue their development through further learning [15]. This notion is supported by the work of Teichler and Kehm (1995) who identify that there is an ongoing rise in the development of lifelong learning processes [21], with the suggested implication that the role a university should adopt is that of preparing the graduates for self development during their careers and not to have a focus of attempting to produce the finished article. This is strongly supported by the findings of Mason and Williams (2003) who found, when surveyed, that the vast majority of employers did not expect a graduate employee to be "work-ready"[13]. In this regard, new employees were expected to undertake additional training to meet the needs of their job. However, the findings also revealed that the role of most training programmes the graduates had completed was to teach the technicalities of the roles, for example how to use software packages or equipment, and not the transferrable skills, such as communication skills. The implication here is that employers would expect graduates to already be in possession of those skills on joining a company.

3. MODULE DESIGN

In designing the first year module to introduce programming concepts to first year undergraduate students, the course needed to address the issues relating to both the employability requirements and also the "fear factor" which is often faced by students when presented with modules in such subjects. The module would make use of the Java programming language. The module runs over two semesters (26 weeks) and is taken by all students who enter the undergraduate Computing and Web Systems Development programmes on offer at Edge Hill University in the United Kingdom. The result is that the module has a cohort size of 50-90 students every year, and that there can be no assumption made about the level of programming experience that the students have on joining the module. The students are typically in classes of 20-25 students and will attend two 2 hour laboratory sessions each week. Additional support is offered to the students through a mentoring scheme in which second year students are available to assist the first years once a week in a laboratory with their programming problems to offer guidance and advice.

The structure of assessment within the module enables module tutors to quickly identify any areas of the subject which may require reinforcement, although the module generally adopts a Problem-Based Learning approach. As such, the students are required to complete two sets of individual portfolio work to develop skills in specific topics within the subject, and also complete two team-based projects within the module to supplement and extend the skills developed through the portfolios. The portfolio elements of assessment are compulsory elements which the students must pass in order to pass the module, regardless of whether a passing grade has been achieved in the other components of the module assessment.

3.1 Developing Employability Skills

The development of employability skills within the curriculum is focused on the team assessments which the students are required to perform. In these assessments, the students are guided in developing their problem-solving skills and in forming effective teams. In doing so, the students develop team norms by which the members of the team will operate, which are essential for effective team operation [14].

Students working within teams will experience a structured environment with a clearly defined role in a project [4,7]. The structure offers immediate access to a support network which may provide support at a number of different levels, both academic and otherwise. Indeed, most teams will also consist of students who possess different levels of experience or knowledge, therefore providing an invaluable resource to the network for members of the team. The teams also nominate a team leader, and assign roles to the team members (although it is important to note that all team members must perform programming tasks on the project). To aid team interaction, the teams are provided with online tools through the institutional Virtual Learning Environment (VLE) which enables the team to communicate and delegate tasks when they are away from the campus.

4. LANGUAGES AND ENVIRONMENTS

To introduce students to the concepts of programming and build both their skills and confidence in terms of problem solving, the modules makes use of two programming environments. The Alice programming environment[1] is used to introduce students to the key constructs of programming, such as sequence, selection and iteration. Given that there can be no assumption made about the previous programming experience of the students, Alice offers two significant benefits for teaching introductory programming. Firstly it makes use of a drag-and-drop interface with code

[1] http://www.alice.org/

Table 1: Module results 2008/09 to 2011/12

	A	*B*	*C*	*D*	*Fail*	*Pass rate (%)*	*No Students*	*Mean*	*SD*
2008/09	10%	28%	44%	16%	2%	94%	50	57.34%	11.27
2009/10	11%	21%	27%	36%	5%	90.32%	62	53.61%	12.61
2010/11	20%	43%	12%	21%	1%	96%	81	59.83%	12.56
2011/12	48%	21%	14%	1%	13%	84%	86	61.72%	20.64

presented in a Java style. This enables students to focus on enhancing their problem solving skills without the necessity of learning language syntax simultaneously. Students are also able to view their code in a style which looks like Java code, and so become aware of the syntax implicitly such as the use of blocks and semi-colons related to the development of Java code. The second key advantage of using Alice is based in its adoption of an object-oriented paradigm. Within Alice, students place objects into a virtual world and interact with those objects by sending messages to the object concerned. Students create objects by selecting the appropriate class from the Alice Class Library. The visualization of these concepts has proven extremely useful when introducing object-orientation to students. One set of portfolio exercises, one formative team exercise and one summative team exercise are undertaken by the students in this section of the module.

In the second section of the module the students move towards programming in Java using the Eclipse IDE. Within this section of the module, the students also complete a set of portfolio exercises along with a team exercise. As an initial component of the portfolio, the students are required to develop their own tutorial guide for Eclipse to help them explore the IDE prior to performing the Java-based tasks.

Following the portfolio, the team task which the students are required to complete has introduced the use of Lego Mindstorms NXT robotics kits into the programming curriculum. In order to program the robots, the students use leJOS NXJ[2] with the relevant plugin for Eclipse. This enables the robots to be programmed using Java running under a customized Java Virtual Machine on the NXT control brick. In the coursework groups of 3 or 4 students are required to solve a number of problems and, in order to do so, the students must design their robot, design their coding solution for each problem, build the robot and program the device for each problem, or level, that has been set. Prior to the introduction of the Lego Mindstorms robots, students were required to develop a game using Java. This remains as an option to students although, as shall be discussed in this paper, the students have been enthused and fully engaged in developing their programming skills through the final team-based project.

5. MODULE RESULTS ANALYSIS

Table 1, shown above, contains the overall module results from the introductory programming module for four iterations. The letter grades relate to percentage scores achieved by the students where

- A = 70-100%
- B = 60-69%
- C = 50-59%
- D = 40-49%

These grade boundaries correlate to degree classifications in the UK. A student is required to attain 40% or greater to pass the module. The Mean column shows the mean student mark achieved within the module in that academic year.

The results which are shown for 2011/12 are those which have been produced prior to the reassessment period, and hence the module pass rate appears lower in this year than in previous years.

It should also be noted that the 2008/09 module results represent the structure of the module prior to the introduction of Lego Mindstorms into the module curriculum. In that academic year, the students were required to implement a game in Java using the skills which they had acquired throughout the earlier stages of the module. The structure of the module, however, remained the same and the students completed the final assessment in small teams. These results are included to represent a baseline from which the effect of introducing robotics as a mechanism for engagement can be judged. However, it is acknowledged that the results achieved by the students in that academic year were impressive, with a high pass rate and a normal distribution of student grades on the module.

The use of the NXT kits was first introduced into the curriculum in 2009/10. In this academic year, all students were required to make use of the robots for their final programming coursework. However, the department only had a limited number of kits for the students to use. Difficulties were also experienced in the laboratory environment when installing and using the software to program the robots. The teaching team addressed these problems by providing the students with caddies with the Eclipse and leJOS software installed so the students could attempt the programming tasks. The students were also provided with pre-built robots, and so selected the robot which they wished to use to attempt the challenges. The results for this year represent the issues which the students faced, with little change in the overall results profile of the cohort.

In the following academic year 2010/11, the Module Leader amended the module to continue the experiment to evaluate the impact of using the Mindstorms NXTs to teach programming. In this year, the use of the robots was optional. The student teams were able to select between a coursework to program the NXT robots, or alternatively to write a game in Java using Swing. The significant difference in experience which the students gained when using the NXTs was that the teams had to design and build

[2] http://lejos.sourceforge.net/index.php

their own robots, and so had greater access to kits which they could use. The teaching team observed a difference in behavior between the students using the NXTs and the students writing the game. Those teams using the robots were on campus and in the laboratories far more frequently, staying until late in the evening and demonstrating high levels of engagement. The module results for this academic year also demonstrate a changed profile of results. The number of students gaining an 'A' or 'B' grade increased significantly, and the overall pass rate and average module result also improved. Furthermore when examining the individual coursework results, it is clear that those students who worked with the Mindstorms for their final coursework achieved improved grades over their previous coursework submissions in the module. In comparison, those students who undertook the game development task achieved results that were broadly in line, or slightly lower, than their previous grades in the module.

In the current academic year 2011/12 the use of the Mindstorms has been extended so that the majority of students would work in teams to design, build and program their own devices to resolve the challenges they were set. A small number of students (8) who had not engaged in earlier teamwork exercises, either out of choice or for personal reasons, were set an alternative programming coursework. Each team was allocated their own kit and was given a brief introduction to programming using leJOS. Again, the teaching team noted high levels of engagement with the teams using the NXT devices. This has been reflected in the results profile for the module. The innovation and creativity which the students have demonstrated has led to over two-thirds of the students achieving an 'A' or 'B' grade. It is also noted that the average grade on the module has increased significantly. If the results for the courseworks in 2011/12 are examined in further detail (Figure 1 below), then there can be seen to a significant change in the pattern of results gained by the students across each of the assessments. The portfolio exercises (shown in gold) and the Alice team exercise (shown in blue) both represent a skewed normal distribution. However, the assessment which engages the students using the NXTs inverts these results. A high proportion of the students achieved grades of 'A' or above (which represents marks of 70% or higher). It is also interesting to note that there is only one team of three students (out of 77 students) who used the NXTs for their coursework who are currently classified as not passing the module. The remaining 8 students who have currently not passed the module consist of those who attempted the game writing coursework. As stated earlier, the reassessment period for this module has not concluded, so it is expected by the authors that the number of students passing the module should increase.

6. STUDENT EXPERIENCE

Whilst it is important to acknowledge the statistical evidence of the improvements that have been achieved by adopting the NXTs in the teaching of programming, the student experience is also a major factor in judging the success of this approach. Each module at Edge Hill University concludes with the students completing an evaluation form which is submitted anonymously and enables the students to express their thoughts regarding the module which they have completed. Each student in the cohort completes an evaluation form which gathers both qualitative and quantitative data relating to the student experience on the module. The results presented in this paper relate to the 2010/11 and 2011/12 academic years which reflect the experiences of the students from the broader introduction of the Lego Mindstorms NXTs into the curriculum.

Broadly, the students in 2010/11 reported that they enjoyed the module and benefitted from the design of the course, describing the course as "stimulating" and "fun and interesting". The use of Alice to introduce programming concepts was identified by a number of students as having a positive impact upon their experiences, with one student stating that they

"Enjoyed using Alice, especially the game coursework."

In many respects, this comment demonstrates not only the benefit of using Alice, but also the value felt by the students in relation to the teamwork elements. Indeed, this is supported by further student comments, such as

"Lots of variety in coursework, and individual/group exercises were good."

In terms of building employability skills, the development of teamworking skills are significant so the identification of these elements of assessment as having a positive impact on the student experience were particularly pleasing.

When referring to the use of the NXT kits, students who made use of the kits were unanimous in their response "Programming using Eclipse and the robots"

Or more simply,

"ROBOTS!!!"

This demonstrated the beneficial effect on their engagement in determining the good aspects of the module. Within this cohort, the issues that were generally raised concerning the module were that the students felt they wanted more lectures and tutorials in Java, and that they wanted to spend more time in the module programming in Java.

The issues raised in 2010/11 were addressed for the subsequent academic year by introducing lecture classes to introduce fundamental concepts to the students, whilst retaining the core PBL elements of the assessments, and moving the transition point from Alice to Java to two weeks earlier in the module.

In comparison to the previous cohort, the 2011/12 evaluations reveal that the students appreciated the wider adoption of the NXT kits as a means of developing their programming skills. They report that "getting to work with the robots" was "interesting" and "fun". The students again identified the introduction to programming using Alice, with comments such as

"I like the way that I came here with very little knowledge in programming. Alice was a great way to start learning."

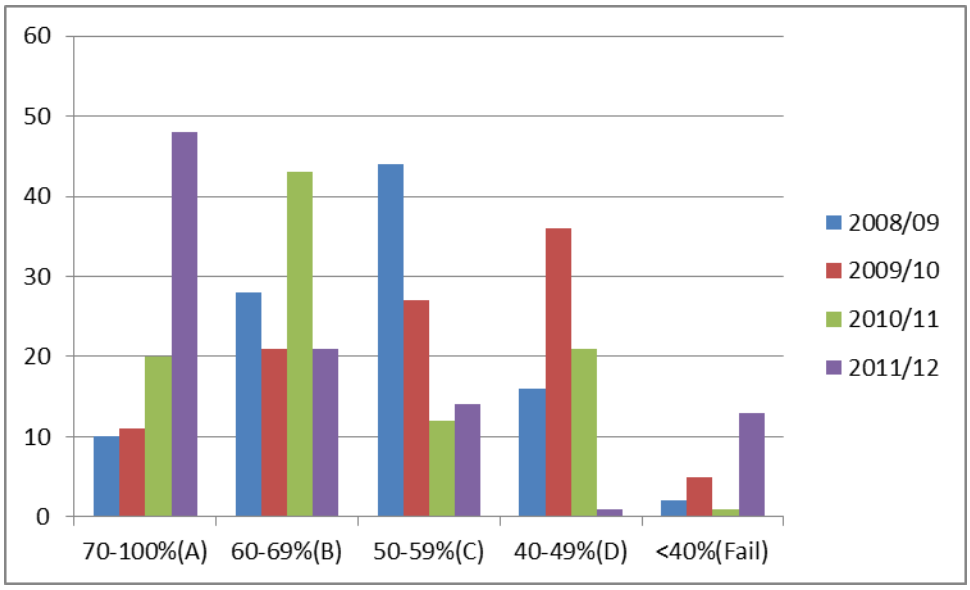
However, of particular interest were the comments which indicated that the students felt benefit from the activities which assist them to develop broader skills to assist them in the workplace, such as teamworking

"Enjoyed working in groups. Helped me to see problems from different perspectives and tackle them differently",

and enhanced problem-solving skills

"Changed my opinion of tackling unfamiliar computer problems and programming overall".

Figure 1: 2011/12 Module coursework results profile



The improvements suggested by this cohort centered around the inclusion of reflective blogs to enhance academic skills, and the provision of longer tutorial sessions to provide further opportunity for team building. These amendments to the module shall be considered for implementation in the 2012/13 academic year.

7. CONCLUSIONS

The introduction of the Lego Mindstorms robots into the first year programming course, accompanied by an introduction to the fundamental concepts through the use of Alice, have had a demonstrable effect on the engagement and results of the students on the module. Where students would be fearful of programming exercises and maintain an attitude of "just doing enough" to be able to pass the module, both the Alice and Mindstorms team assessments have led to students wishing to extend their skill sets beyond those that are taught in the course and push themselves to produce work which far exceeds the requirements of the course. This is reflected in both the assessment results and the student evaluations. The module results show a marked improvement in the assignments where the students collaborate and develop their skills using the NXT kits. The module evaluations provide feedback which demonstrates that the activities within the module resulted in a stimulating and engaging challenge for the students, but also that the students felt their teamworking and problem-solving skills had developed through the courseworks. Indeed, the interest and excitement which has been generated through these changes to the module have led to further changes in the curriculum.

Modules have been introduced in the second and third years which explore the development of robots, maintaining a focus on programming, for students to further enhance their problem-solving skills. These modules have proven to be extremely popular, with module registrations increasing five-fold over the three years that the modules have been running. The second change to the curriculum has been the introduction of a new route within the Computing suite of programmes which specializes in the development of programming skills. The route, Application Development, followed a significant number of requests from students for further modules and opportunities to develop their programming knowledge. This is in stark contrast to the Computing programmes of four or more years ago at Edge Hill University when students would try to avoid programming modules if they could. Both of these developments have resulted in increased creativity and inspiration by the students to develop innovative software using new and emerging technologies, such as the Microsoft Kinect camera, where use of the SDKs are not currently taught within the curriculum.

In terms of addressing employability skills, then the changes in the curriculum that have been driven by the model of teaching programming introduced in the first year programming module and discussed in this paper have brought significant and tangible changes to the behavior and interaction of the student cohort. This is reflected in the students' ability to work within teams, to negotiate during teamwork, to engage with external clients during projects, and in designing solutions to problems which may lie outside of the topics covered within the modules that they have studied. Students are actively interested in developments across the broad spectrum of Computing subjects, and are bringing ideas for external visitors, workshops and the introduction of technologies to the departmental staff.

8. ACKNOWLEDGMENTS

Our thanks to the Computing students at Edge Hill University for participating in this study.

9. REFERENCES

[1] Azemi, A. 1995. Teaching Computer Programming Courses in a Computer Laboratory Environment. FIE '95 Proceedings of the Frontiers in Education Conference on 1995. Volume 1.

[2] Barrie, S.C. 2006. Understanding what we mean by the generic attributes of graduates. Higher Education51: 215-241

[3] Cavus, N, Uzunboylu, H. and Ibrahim, D. 2006. Using the Moodle Learning Management System and GREWPtool Collaborative Tool for Teaching a Programming Language. Presented at Creating the Future 4th FAE International Symposium, University of Lefke, 30 Nov – 1 Dec, Lefke, Cyprus.

[4] Delamont, S. (2001). 'The marriage analogy: personal and positional relationships in PhD supervision'. Presented at the Higher Education Close Up Conference 2, Lancaster University, 16-18 July 2001. Available at http://www.leeds.ac.uk/educol/documents/00001732.doc (accessed 8 May 2009)

[5] Enzai, N.I.M., Kar, S.A.C., Husni, H. and Ahmed, N. 2011. Harnessing Creativity in Teaching and Learning Introductory Programming. In Proceedings of 2011 IEEE Student Conference on Research and Development. Putrajaya, Selangor, Malaysia

[6] Fincher, S. 1999. What are we doing when we teach programming? 29th ASEE/IEEE Frontiers in Education Conference, 1999.

[7] Finlay, I. (2002) 'Online support and learning: a study of part-time students on a doctoral programme'. Available at http://www.leeds.ac.uk/educol/documents/00002314.htm (accessed on 8 May 2009).

[8] Kelleher, C. and Pausch, R. (2005) Lowering the Barriers to Programming::A Taxonomy of Programming Environments and Languages For Novice Programmers. ACM Computer Survey, 37 (2), pp. 83–137.

[9] Kolb, D.A. 1984. Experiential Learning: experience as the source of learning and development. New Jersey: Prentice Hall

[10] Kuljis, J. 2004. Orienting the Teaching of an Introductory Object-Oriented Programming to Meet the Learning Objective. Proceedings of the 26th International Conference on Information Technology Interfaces

[11] Lin, C.H. and Liu, E.Z.F. 2012. The Effect of Reflective Strategies on Students' Problem Solving in Robotics Learning. In Proceedings of Fourth IEEE International Conference on Digital Game and Intelligent Toy Enhanced Learning, Takamatsu, Kagawa, Japan.

[12] Matthew, R.G.S. and Pritchard, J. 2009. Hard and Soft – A Useful Way of Thinking about Disciplines? In Kreber, C. (ed) The university and its disciplines: teaching and learning within and beyond disciplinary boundaries. Taylor and Francis.

[13] Mason, G. And Williams, G. 2003. How much does higher education enhance the employability of graduates?. Summary report to HEFCE. Available online http://www.hefce.ac.uk/pubs/rdreports/2003/rd13_03/rd13_03sum.pdf [Accessed 21/12/11]

[14] Mayo, E. 1945. The social problems of an industrial civilisation. Cambridge, MA: Harvard University Press.

[15] Raelin, J.A. 2008 Work-based learning: bridging knowledge and action in the workplace. San Francisco: Wiley

[16] Satratzemi, M., Xinogalos, S. and Dagdilelis, V. (2003) An Environment for Teaching Object-Oriented Programming: ObjectKarel. Proceedings of the Third IEEE International Conference on Advanced Learning Technologies (ICALT '03). Athens, Greece.

[17] Saunders, M. And Machell, J. 2000. Understanding emerging trends in higher education curricula and work connections. Higher Education Policy. Pergamon. Pp. 1-16

[18] Savin-Baden, M. 2000. Problem-based Learning in Higher Education: untold stories. Open University Press.

[19] Sheard, J. and Hagan, D. 1997. Experiences with Teaching Object-oriented Concepts to Introductory Programming Students using C++. Proceedings of the Technology of Object-Oriented Languages and Systems (TOOLS'97).

[20] Sobrinho, W.T. and Goncalves, R.S. 2011. Study of Robotics Singularities using LEGO Mindstorms Kit.. Proceedings of 2011 IEEE IX Latin American and IEEE Colombian Conference on Automatic Control and Industry Applications (LARC), 1-4 Oct, Bogota, Columbia.

[21] Teichler, U. And Kehm, B.M. 1995. Towards a New Understanding of the Relationships between Higher Education and Employment. European Journal of Education 30(2). Pp 115-132.

[22] Universities UK Response to the Higher Education White Paper (online). Available at http://www.universitiesuk.ac.uk/Publications/Documents/ResponseToTheHigherEducationWhitePaper.pdf [Accessed 21/12/11]

[23] Winter, R. and Maisch, M. 1996. Professional Competence and Higher Education: The ASSET Programme. London: The Falmer Press.

[24] Xiaohui, G. and Yaguan, Q. 2010. The Exploration of Teaching about Java Programming. Proceedings of The International Conference on E-Business and E-Government, ICEE 2010, 7-9 May 2010, Guangzhou, China.

[25] Yorke, M. and Knight, P.T. 2006. Embedding employability into the curriculum. Learning and employability series 1. The Higher Education Academy. Avaliable online http://www.heacademy.ac.uk/resources/detail/employability/Embedding_employability_into_the_curriculum [Accessed 22/12/11]

.