**National Key Centre for School Science and Mathematics**

# A Case Study of Portfolio Assessment in an Introductory Computer Programming Course

**Beryl Elizabeth Plimmer**

**This thesis is presented as part of the requirements for the award of the Degree Master of Science of the Curtin University of Technology**

**December 1999**

***Abstract***

This thesis is a case study on the use of portfolio assessment in an introductory computer programming course. The literature review examines the research on programming and learning to program, general learning theories, assessment and the role of assessment in learning. The methodology used was a triangulation of academic results, student and teacher opinions. The results and analysis showed portfolios to be a fair measure of student ability and a positive assessment and learning tool. However portfolios cannot totally replace controlled assessment, as a controlled assessment is necessary to prevent plagiarism. The study concluded that student portfolios were a valuable assessment and learning tool for computer programming courses.

***Table of Contents***

# 1. Introduction

My initial goal when undertaking this study was to explore more effective ways to teach computer programming to novice programmers in a tertiary environment. My research into teaching and learning in this environment has lead me to believe that the most likely way to effect learning was by the use of active assessment methods (Berson et al., 1998; Biggs, 1998; Popham, 1995). When examining the range of possible assessment methods, I was struck by the obvious relationship between the skills needed to be a commercial programmer and the theoretical advantages of portfolio assessment.

This thesis is a case study of the use of portfolio assessment in an introductory computer programming course. The first chapter reviews the literature for the different components of learning to program: the skills required to program, learning and the role of assessment in learning, and assessment. There is little evidence of people either examining assessment in relationship to programming courses or using student portfolios in programming courses.

I will then set out the theoretical basis for the case study and the methodology employed. The case study will be evaluated using triangulation techniques where the opinions of the students and teachers together with the students' results are analysed. A summary of student surveys, discussions with teachers and student grades and an analysis of these findings will follow.

Finally, in the conclusion section, I will convey my understanding of the strengths and weaknesses of portfolio assessment as it was implemented in the case study and indicate areas where future investigation could be undertaken.

## 2. Literature Review

This section will firstly examine the literature on computer programming from the perspective of both expert programmers and programming teachers. It will then explore theories of learning. These two fields will then be drawn together in an analysis of the literature on learning to program. The final section will examine the role of assessment in learning and teaching in a tertiary environment, with specific reference to computer programming courses.

### 2.1. Programming

First I will review computer programming as a cognitive activity. Programming is the task of analysing and solving problems using a computer programming language (Booth, 1990). This definition however, over simplifies the task. Booth goes on to describe programming as a complex phenomenon composed of only slightly less complex phenomena.

Lammers (1996) interviewed many of the forerunners of modern programming. One of the questions she asked was “Is programming an art or a science?” Simonyi, one of the principals of Microsoft Corporation and leader to the team who developed MS Word and MS Excel responded “Some people call it a science, some people call it an art, some call it a trade or skill. I think it has aspects of all three.” (p.11). Kildall, who wrote the first microcomputer operating system, CPM and one of the early high level programming languages PL1 said “There is certainly some art in it. But a lot of programming is invention and engineering. It’s much like a carpenter who has a mental picture of a cabinet he is going to build”. He went on to say, “Programming has some science in it as well” (p.65). Bricklin, the author of VisiCalc, the first spreadsheet program said, “Parts of it are science and parts are skill. There is a craft component” (p.149). Consistently, they described programming as a skill, a science and an art. Lammers questioned many of the nineteen programmers she interviewed as to how they created a program. Most replied that the first step was visualising the finished product. From there they progressively broke the problem down into smaller and smaller parts and then wrote the code. Shneiderman (1980) and

Blum (1992) have made similar observations about the way expert programmers undertake program construction.

Educators generally describe programming in the reverse order to the practitioners. To create a program the programmer needs detailed knowledge of: the programming language syntax (skill) and problem decomposition and algorithm construction (science) (Linn, 1985; Maheshwari, 1997; Shneiderman, 1980). Thomas and Upah (1996) added to this list the isolation and removal of syntax and logic errors and a review of the solution. Blum (1992) described programming as knowing "What to do and how to do it". While Shneiderman (1980) described programming in terms of syntax and semantic knowledge.

No direct link was seen between practitioners' 'art' and educators' descriptions of programming. It could be that the practitioners whom Lammers (1996) interviewed were the visionaries of the modern software era. They were the people who first created operating systems, programming languages and applications software. In the past most commercial programmers have not needed this level of creativity. Yet at the same time these inventors saw creativity as an important aspect of all programming and talked of the beauty and elegance of programs. The visual interfaces of recent years require another aspect of creativity, visual design skills, from all programmers.

Each of these definitions of programming is a different interpretation of the same tasks (McGill & Volet, 1997). The programmer needs detailed knowledge of the programming language and associated host software. This may include the integrated development environment (IDE), the operating system, network software, database management system (DBMS) and internet protocols. He or she will also need to be able to interpret the problem specifications and create algorithms to solve the problem. Lastly the programmer needs the vision to be able to 'see' the completed software.

Programming languages and host software with their complex syntax and rules take some time for beginners to master but they become automatic for the experienced programmer. Algorithm construction and problem solving are more

challenging activities. The common approach to problem solving is to decompose the problem in small parts and then solve each part. However the depth of the hierarchy that this creates even for relatively simple programs confronts the programmer with complexity ratios in the region of $10^9$ (Dijkstra, 1998). Dijkstra claims that the conceptual hierarchies are deeper than humans have ever had to handle before and present a radically new intellectual challenge.

The nature of digital representation imposes another challenge for the programmer. The alteration of one bit of data or one character of program code can mean the difference between success and total failure (Dijkstra, 1998).

The final part of programming, the art of creating really great software, was the least discussed by educators. Expert programmers work by decomposing a problem into parts that they can fit to pre-existing templates or structures (Soloway & Iyengar, 1986). Linn (1985) describes these as stereotypical patterns of code that use more than a single language feature. However in order to do this the programmer must be able to mentally formulate a solution. This is the creative, artistic part of programming that is the most difficult to master (Lammers, 1996).

The challenge for the student was to become proficient at a programming language while at the same time developing problem solving and problem decomposition skills. Alongside this, their creativity needs to be stimulated so that they can mentally compose a problem solution.

Bill Gates, Microsoft Corporation, described it thus, “You have to simulate in your mind how the programs going to work and you have to have a complete grasp of how the various pieces of the program work together” (Lammers, 1996, p.73).

## 2.2. Learning

Educational psychology has long debated the mechanisms by which people learn and has attempted to define levels or stages of knowing and understanding. Piaget described the stages children pass through as sensori-motor, pre-operational, concrete operational and formal reasoning (Gardener, 1992). Carpenter (1980) suggested that Piaget's final stage is a prerequisite for any mathematics beyond arithmetic. As programming has similar requirements for abstraction it is reasonable to assume that this is also true for programming. While Piaget believed that these steps where a natural progression through which all 'normal' children would pass independent of instruction, others have suggested that this is not the case. Vygotsky (Carpenter) and his associates claimed that intellectual development is inextricably linked to school learning. In the following section I will look at the learning of problem solving or cognitive skills and the intellectual development of young adults and adults.

### *Problem Solving*

Plato argued strongly that problem solving could be taught and learnt as an independent skill (Mann, 1979). However endeavours in artificial intelligence (AI) over the last thirty years have not been successful in defining any general problem solving techniques (Taylor, 1991). AI has moved to the creation of domain specific systems where the knowledge engineer elicits knowledge from the expert, this consists of both descriptive and heuristic knowledge (Gonzalez & Dankel, 1993; Taylor, 1991). AI's categories of descriptive and heuristic knowledge can be linked directly to educational psychology categories of knowing what and knowing how. It seems likely then that learners will be most successful learning and problem solving in a specific domain where knowledge and skills are taught in conjunction with each other (Edwards, 1991). Failure to solve problems can often be linked directly to insufficient domain specific knowledge (Glaser, 1984).

Edwards (1991 p. 90) summarised the findings of other researchers on knowledge acquisition

- Any educational program needs specific cognitive objectives (these are the required knowledge structures and processes to operate them successfully).

- Metacognitive knowledge (conscious awareness of oneself as a problem-solver plays an important role in problem-solving success.
- Students must understand both the conditions under which the components of their knowledge are applicable and the relationship among their various pieces of knowledge. That is they need some sort of high-order control schema or executive knowledge.
- No skill can be learnt without sufficient well-structured practice.

The passage from novice to expert is one that requires both knowledge and thought. A person needs to think in order to obtain knowledge. On the other hand knowledge is essential to thinking (Evans, 1991a, p.1). Hence we see a spiral of epistemological growth; learn skills, practise, apply, learn skills, ….. As skills are practised they become increasingly automatic until they become autonomous with little need for deliberate control. At each stage the skills of previous stages need to become more automatic to reduce cognitive load (Edwards, 1991).

Knowledge and thinking have been variously described as ‘knowing that’ (Nickerson, 1985; Ryle, 1949) or ‘declarative knowledge’ (Anderson, 1982) and ‘knowing how’ (Nickerson, 1985; Ryle, 1949) or ‘procedural knowledge’ (Anderson, 1982). Evans (1991b) goes further identifying four specific perspectives. The first is procedural knowledge, knowing how. This is, specific tasks that an individual can perform routinely in response to certain environmental cues. The next, propositional or declarative knowledge (Anderson), is knowledge of the concepts. It may be used in two ways, firstly as the initial stage of learning procedural knowledge (before it becomes automatic) or may explicitly or implicitly control skills. The third, higher order procedures, are representative knowledge that can be used to modify and extend specific procedures and propositional knowledge. This includes selecting, modifying and combining specific procedures. The last, executive procedures, refers to the individual’s ability to manage and monitor their pursuit of goals, otherwise cited as the metacognitive process.

In order to learn a new concept the student must create a mental model that corresponds to the structure of the task (Boulton-Lewis & Halford, 1991). The

processing load this requires depends on the structural complexity of the task. This would suggest that learning discipline specific skills will allow the student to chunk knowledge and thus reduce cognitive load (Edwards, 1991).

***Intellectual Development***

Perry's (1970) studies into the intellectual and ethical development of people through the college years have resulted in him proposing nine 'positions' that are commonly held by people during early adulthood. His first position is 'basic duality' where matter is viewed as right versus wrong or good versus bad. Perry observed that over time students become aware of variety of opinions and outlooks that people hold, so move to a position he describes as 'multiplicity'. At this point anybody's opinion is valid and one opinion is as good as another .At the early stages of multiplicity, the student does not see the need for their position to be justified. Perry suggests that intellectual challenge encourages the students to move from 'multiplicity' to 'relativism' where he or she understands that the truth of a position is relative and the meaning of an event depends on the understanding of the event. This is not to say that any meaning is correct, but that a position must be supported by valid arguments.

More recent studies by Belenky and colleagues (1986) have supported Vygotsky's (Carpenter, 1980) and Perry's (1970) findings. They describe five levels of cognitive development. The first is 'silence', people who have not developed the capacities for representational thinking. Their research (exclusively on women) found that a few adults were operating at this level. This is followed by 'received knowledge', the ability to hear, understand and remember (Perry's basic dualism). Their third level is 'subjective knowledge', an internal voice and a quest for self replace the external voice of received knowledge. This is the 'normal' level for mid to late teenagers and common in first year tertiary students. The fourth level is 'procedural knowledge'. At this level people are consciously, deliberately, and systematically analysing the acquisition and application of procedures for obtaining and communicating knowledge (Belenky et al., 1986, pp 93-95). They suggest that movement to this level is most often triggered by the demands of academic study. The final level

that Belenky and colleagues define is ‘constructed knowledge’ (Perry’s relativism). At this level people are able to integrate the internal and external voices. “Ultimately constructivists understand that answers to all questions vary depending on the context in which they are asked and on the frame of reference of the person doing the asking” (Belenky et al.). They align this level to Piaget’s term “horizontal décalage”.

Belenky and colleagues (1986) followed a number of women over a period of years. Women were observed at each of the five levels. They noted that many developed as a response to demands placed on them either by life-changes such as motherhood or divorce, or academic study. They concluded that development was not necessarily linear or tightly linked to age, but rather closer to Vygotsky’s (Carpenter, 1980) model of response to environment.

What is clear, through the varying description of the ways in which people learn, is that there is a continuum of epistemological positions that people will pass through. Some will not progress past the most basic level and it is probable that only a small percentage will get to the highest levels. It is also likely that academic study will accelerate this developmental process.

***Learning to Program***

Three component parts of programming have been identified. Programming requires the programmer to follow language syntactic rules exactly. A programmer also needs to be able to systematically deconstruct and reconstruct problems. Finally a programmer needs to be able to imagine the complete product at the inception. Each of these parts can be placed at a different position on the developmental frameworks.

The first requirement, to be able to apply the program language syntax, is the simplest. The rules of language syntax are well defined and the compiler will clearly direct the programmer to any transgression of these rules. Language syntax can be categorised as “knowing what” (Nickerson, 1985; Ryle, 1949) and

can be placed at the level of Perry's (1970) 'basic dualism' or Belenky and colleagues' (1986) 'received knowledge'.

The second requirement, to be able to deconstruct and reconstruct problems is more difficult (Linn, 1985; Shneiderman, 1980). To be able to successfully achieve this a programmer needs to consider the complexity of the problem and the various logic paths that may be required in a program to successfully complete a task. The requirement here is for the programmer to be able to identify the range of values that the program may be required to deal with. This can be categorised as "knowing how" (Nickerson, 1985; Ryle, 1949), however as a programmer becomes more experienced, they build up a set of templates of standard algorithms automating some of this process (Linn; Shneiderman). The identification of the different logical pathways moves the cognitive requirements to Perry's (1970) 'multiplicity' or Belenky and colleagues' (1986) 'procedural knowledge'.

The last requirement of programming is to be able to mentally 'see' a solution. Simple programs that solve problems where the algorithm and solution is well known are at the 'received knowledge' level. More complex programs that are similar to other problems that the programmer has solved previously may be solved by template matching (Linn, 1985; Shneiderman, 1980). Many business problems are at this level in that they fit general business algorithms and can be categorised at the 'high order procedures' (Evans, 1991b) level. However the creators of truly novel programs are clearly working at a higher level. These programmers have the ability to imagine new ways to use a computer either to solve problems or to entertain the user. The creative or artistic aspect of programming can lie anywhere along the epistemological continuum depending on the problem, however most useful programs would require the programmer to be operating at 'multiplicity' or 'procedural knowledge' level.

The traditional approach to the teaching and learning of programming has been to explore the programming language in small discrete sections and for the students to complete a number of well understood exercises to practise a particular skill. This model concentrates on the lower cognitive requirements of

programming. Under this model of tuition the normal assessment regime is a number of assignments and an examination. Most argue that this results in shallow rather than deep learning (Dijkstra, 1998; Linn, 1985; Linn & Clancy, 1992; Milbrandt, 1995) that will not equip the student well either for further study or real programming.

Linn and her colleagues (Linn, 1985; Linn & Clancy, 1992) have made extensive use of case studies to teach programming. They provide models of how experts solve programming problems for students to analyse. The students then solve problems following the experts' methods that they explored. Milbrandt (1995), describes a similar method of focusing on problems and teaching to the solution of these problems.

Dijkstra (1998) advocates the extensive use of formal methods of algorithm description before any instruction in program language syntax. The belief is that formal methods will result in better algorithm definition and more robust code.

No studies have been identified where the assessment tools were examined as to their contribution to the students' learning.

### 2.3. Assessment

Assessment in an educational setting is some judgement about a student's learning or knowledge (Knight, 1995). There are two major reasons to assess, to improve learning and for accountability (Angelo, 1994; Renwick & Renwick, 1992). The first is close to teachers' hearts, the second is generally imposed on us, and therefore in the 'must do' category. It is perhaps for this reason that teachers and students often resent assessment rather than see it as a part of learning. Renwick and Renwick (1992) also include institutional management and monitoring. There is an alternative view that an integrated approach to assessment can be at the heart of student learning (Knight, 1995; Popham, 1995).

The focus of this study is the tertiary environment. The tertiary classroom of the 90's is more diverse than ever before. There is a wide range of engagement,

abilities and cultures. It cannot be assumed that the student will engage in the learning process regardless of the delivery and assessment method. “Good teaching is getting most students to use the higher cognitive levels that the more academic students use spontaneously” (Biggs, 1998). The teaching focus must be clearly on what the student does and there must be a clear alignment between objectives, teaching and assessment (Biggs). In an environment where attendance at class and completion of exercises and homework is optional the most powerful tool that at teacher has to encourage student participation is the part, which in the student’s view, really counts – assessment (Popham, 1995).

The continuum of possible assessment tasks is huge; clearly each will affect the learning experience in a different way (Hager & Butler, 1996). At one end of the continuum are three hour finals as the only assessment, at the other workplace assessment and informal evaluation by student or teacher that is not reflected in a official result. In between there are other typical task such as essays, assignments, case studies, projects, tests, attendance and portfolios. Regardless of the purpose of assessment it must be seen as a fair and valid measure of the skills and abilities it is intended to measure (Herman, 1992). A review of the main categories of assessment tools will describe the advantages and disadvantages of each.

### *Tests and Examinations*

Tests and examinations are perhaps the most frequently used forms of assessment. They are generally conducted under controlled conditions and the student has a number of set questions to answer (Woolfolk, 1993). There are a number of advantages of this form of assessment. Firstly the examiner can be more certain that the work is that of the individual. A test also concentrates that time spent on assessment, leaving the majority of course time available for learning. There are also some disadvantages of limited time tests. Tests are effectively a snapshot of a selection of a student’s knowledge. It is not possible in a three hour final to assess all of a semester’s work. Tests are, of their very nature, a sampling procedure. The other major disadvantage is that it is difficult

to assess higher order cognitive skills with small problems that can be solved in a limited time.

Controlled assessments present a dilemma when one evaluates their fairness and validity. Many would claim that they are more fair and valid than uncontrolled assessment as one can be sure that the work is that of the individual and all participants have equal time and resources. However it is also claimed they often have little affinity with real tasks that they purport to assess (Gardener, 1992).

### *Assignments & Case Studies*

Typical programming assignments require students to write a program that solves a problem that has been defined by the teacher. Assignments have some advantages over tests in that they allow the students to undertake larger pieces of work over a longer time period (Woolfolk, 1993). Carefully designed assignments can cover the language syntax requirements for programming by specifying problems that can best be solved by use of a wide range of language elements. It is also possible with assignments to challenge students to decompose significant problems and then construct an algorithm to solve the problem; this gives the student practice at the second component of programming, problem decomposition and reconstruction.

However assignment work also has some drawbacks. It is often not possible to be certain that the work submitted is that of the student submitting it. The other major drawback is that if a class is all working on the same problem, generally one or two of the more able members of the class will solve the difficult aspects of the problem and share the solution with the group as a whole. An example of Vygotsky's (Carpenter, 1980) theory of learning from peers in practice, but in this case to the detriment of most students. Often the tasks set as assignments have little relationship to 'real' programming problems. Case studies are in many ways similar to assignments (Linn & Clancy, 1992). Generally case studies resemble a 'real' problem, although the problem had often been 'sanitised' by the

teacher and therefore lacks the complexity and messiness of the real world (Plimmer, 1999).

Whenever the teacher specifies the problem it is difficult to engage the student in the more creative aspects of programming. A problem specification will of its very nature direct the solution space (Gonzalez & Dankel, 1993). This means that it is difficult with assignments and case studies to fully engage the student in the third component of programming, creativity. Clearly assignments and case studies are more fair and valid in allowing the student to complete work in an environment that is close to the normal work environment, although, as will be described later, programmers are often required to work in conditions that more closely resemble examination. However some students do not submit their own work. The origin of work is often difficult to prove and validity and fairness become a real issue if there is no controlled assessment.

***Portfolio Assessment***

Recently there has been a shift to authentic assessments, assessment that mirrors the real-world and is integrated with learning. Assessment where the learners are active participants and the criteria are open and negotiable (McDowell, 1999). The goal is to engage learners in the assessment as well as the learning. This engagement should assist the learner to develop better learning and self-evaluation skills that are so vital as we move to a society where life-long learning must be the norm. Portfolio assessment is perhaps the most common form of authentic assessment.

In the broadest terms portfolio assessment is a purposeful collection of a person's work The critical factors in successful portfolios are: the portfolio is a limited selection of the student's work, the selection process is the responsibility of the student, the process and expectations are explicit, clear criteria against which the portfolio will be judged are provided (Arter & Spandel,1992 p.36; Gardener, 1992, p.36; Lester & Kroll, 1991).).

With portfolio assessment the teacher does not specify what the student should do, but rather what skills he or she should demonstrate. Portfolio assessment has traditionally been used in creative fields such as fine art and music. More recently it has become common in a much wider range of educational settings (Gardener, 1992). A well-designed portfolio will meet the teachings goals of engaging the students in the higher order cognitive of activities of reflection and creativity (Arter & Spandel, 1992; Biggs, 1998; Gardener, 1992).

Gardner (1992) suggests that portfolios act as a silent mentor to students; they are instruments of learning rather than a repository. The student becomes a responsible partner in documenting their learning (Wolf, LeMahieu & Eresh, 1992, p. 10). Arter and Spandel (1992) list a number of issues that must be addressed when designing portfolios. Firstly they suggest design responsibilities should rest with the teachers and students to preserve ownership. Secondly the purpose to which the portfolio is to be put must be defined, as this will affect the latitude that teachers and students have in their creation. Thirdly the portfolio must link naturally to instruction. Arter and Spandel's (1992) final point was the content, what degree of standardisation will be required, will the portfolio include draft as well as final work? Kerka (1995), stressed that the tasks presented must be meaningful and have some real-life application.

When evaluating portfolios teachers can look at a number of things. They can evaluate the knowledge and skills, the reflection that is demonstrated, the response to feedback, the number and range of entries (Arter & Spandel, 1992; Gardener, 1992). Some teachers believe that it is contradictory to use portfolios to expand our view of student achievement and then contract it again to a grade (Arter & Spandel, 1992; Boughey & Searle, 1998). This would limit its use to formative assessment. However in the tertiary sector it is likely the majority of students will value only what is graded (Lester & Kroll, 1991; Biggs, 1998).

To use portfolios effectively in a programming course the three components of programming need to be addressed. For the first of these, program syntax, the teacher can specify the syntax the student must demonstrate. The second problem decomposition and algorithm construction will be required to solve any non-

trivial problems. Again these criteria will need explicit specification. The third component of creative programming is the most difficult to address. Portfolio assessment gives the student the opportunity to explore the creative side of programming by not requiring the completion of predefined tasks. It is the student's responsibility to think of problems and their solutions.

Portfolio assessment does have some disadvantages. As with assignments and case studies the teacher cannot be sure that the work submitted is the student's. This is potentially more of a problem than with predefined tasks as the student can simply submit work copied from a book or the internet and it would be difficult to prove that it is not the student's original work. The other major difficulty with portfolio assessment is assigning grades. When students undertake a wide variety of tasks it could be difficult for a teacher to grade in a fair and consistent manner.

To summarise, the two primary goals of assessment are to aid learning and evaluate students. It seems likely that portfolios are an effective method of assessment for the promotion of learning. Many have assumed that course work assessment is 'fairer' and a more accurate evaluation of a student's ability. However, Supovitz and Brennan (1997) carried out a large study comparing portfolio assessment to standardised tests and found that ethnic, gender, and socio-economic inequities persisted with portfolio assessment. While some groups scored better in one regime or the other there seemed to be little overall difference as a measure of achievement.

# 3. Theoretical Base

## 3.1. Learning

The purpose of the course that this case study focuses on was for the students to learn the principles of computer programming. This was a first-semester undergraduate degree course. There were three distinct demographic groups of students. About half of the students entered the course directly from school. A quarter were new immigrants from Asia, most of whom have a degree and work experience but are unable to practice their profession in New Zealand because of professional registration requirements and poor English. The balance were mature students retraining.

It can be assumed that many of the students enter at Perry's (1970) position of 'basic duality'. The philosophy of the degree is to graduate students who are at least at Perry's position of 'multiplicity' and are considering the shift to 'relativism'. It is very easy for students in a subject such as programming to look for 'the right answer'; where in reality there are many right answers. This course should encourage students to explore programming and apply programming techniques to a range of problems.

There are three major elements in programming: the language syntax, problem decomposition and algorithm construction, and creativity (Lammers, 1996). This course introduced the students to each of these elements. The starting point for learning was language syntax as without solid basic syntax it is difficult concentrate on problem solving (Edwards, 1991; Glaser, 1984; Mann, 1979; Evans, 1991b). The first three weeks focused almost entirely on language syntax. This was followed by an introduction to problem decomposition and algorithm construction. Over the next two weeks language syntax was expanded so that the students had covered the basic data types, assignment, selection and iteration instructions.

At this point the students were encourage to put several of their small programs together to build some bigger programs to solve simple problems. These programs formed the basis for the first portfolio submission. By leaving the

problem choice to the students they could work from their own expertise while they engage in the third element of programming – creativity, without excessive cognitive load (Edwards, 1991; Evans, 1991b). These tasks facilitated the construction of templates that will reduce the cognitive load as students progressed to the next step (Linn, 1985; Shneiderman, 1980).

While the students were consolidating the basic syntax and preparing their portfolios, the course moved to a discussion on software development life cycles and the history of programming languages. The focus then returned to language syntax with the introduction of functions and arrays. This expanded syntax allowed the students to solve a far wider range of problems. They were given many sample problems and solutions. The second portfolio submission had three elements. The first was to write an essay on one of the theoretical aspects that had been discussed (program development life cycle or history of programming languages). Secondly, they had to find a program in a similar language to C and translate that program to C. Lastly they designed and wrote a more substantial program in C demonstrating the new syntax they had learned.

With this portfolio submission the three elements of programming were addressed, and program syntax was expanded and demonstrated in the portfolio. Problem decomposition was more formally addressed with the students describing their program in formal terms. Requiring the students to define their own problem fostered creativity. The task of program translation from one language to another had the dual effect of exposing them to a different language and requiring them to translate a problem solution from one syntax to another. This exercise demonstrated to them that language syntax is merely a tool that they must employ in order to solve problems and move their thoughts to the more important task of problem decomposition.

In the final portion of the course the language syntax was expanded to include data structures, basic pointers and data files. There was also time spent on examining a range of standard business algorithms. For the final portfolio submission the students were provided with data files and program code to read and write these files. The students were expected to apply some of the standard

business algorithms to a data file to build a mini system. The data files were nominally named ‘members’, ‘music’, and ‘products’. Each file contained a set of records; each record had at least one field of each of the fundamental data types. The students built a program to add, change and delete records, sort and report the information. The scenarios that the students constructed and business rules they appled were at their discretion.

There was an emphasis for the final program on well-designed and structured algorithms and elegant code. In this submission the three elements of programming were integrated with the students creating the scenario and then designing and writing a program to fulfil the scenario. The existence of standard algorithms can be a point of revelation for some students. They realise that there are ‘template’ approaches and they can apply the syntax to the pre-existing algorithms. Generally the algorithms need some adaptation, this adaptation requires a high level of understanding.

There was also an extension section for the final submission. More able students were encouraged to write programs completely outside the course work and submit them in this section. The problem domain was open. The student simply described what they had done and filed their programs.

### 3.2. Assessment

Assessment was required as part of the degree requirements. The challenge was to design an assessment plan that would meet the need to evaluate students and to foster learning (Angelo, 1994). The review of assessment techniques showed advantages and disadvantages of differing types of assessment. For this reason a two-fold approach was taken. First the students prepared a portfolio of course work. This was complemented by a final examination.

#### *Portfolio Assessment*

Students were required to submit their portfolios three times over the semester. On the first and second submission the portfolio was graded by discussion between tutor and student. The tutor wrote a summary sheet noting good points,

points that needed improving, and omissions. On the final submission the student could submit 'rework' of any work from earlier submissions. This was limited to rework, entirely new work was not permitted.

There were a number of specific elements built into the design of the portfolio. First, the submission of the portfolio three times. This was a deliberate decision because we were aware that many students will only complete assessed work (Biggs, 1998; Popham, 1995) and we believed that it was important they complete some work to the best of their ability early in the course and receive feedback. Secondly we decided to mark the first and second submissions with the students. Clearly it would not be possible to hand out model answers so we believed that a discussion would be a more educative experience and we would probably disseminate more of our thinking than if we wrote comments. We also believed that the students would be more likely to improve their self-evaluation skills if they understood how we assessed their work. Thirdly the facility to rework early submissions was a direct response to the principles of a portfolio being a collection of the student's work at the completion of the course, in contrast to assignments where the marks are locked in earlier.

We envisaged that discussion amongst the students would engender more learning as there would be a wider variety of experiences happening within the group (Carpenter, 1980). While reflection is a critical and valuable part of portfolio assessment we decided not to grade it as recommended by Sumsion and Fleet (1996).

***Examinations***

Examinations have lost favour in recent years, however two short examinations were included at the conclusion of the course. The theory examination concentrated on the background computer knowledge required for programming. The practical examination required the students to write small segments of program code on the computer.

One could argue that programmers are often put in situations similar to examinations: limited time, high stakes and high stress. I asked one of our recent graduates how the stress of examinations compared with that of his job as a programmer. His reply was "I was at one of my client sites last week and the whole automatic bakery came to a grinding halt because of *my* program bug. Everyone was asking me how long it was going to be before the plant was up and running again because there was dough and bread stuck at every stage of production – that's stress! If you can't program in an exam you can't be a programmer."

***Pros***

In summary the theoretical advantages of portfolio assessment that we were endeavouring to capture were:

- Give the students the opportunity to be more creative.
- Require the students to actively assess their own work.
- Broaden the group experience with the variety of work that the group as a whole completed.
- Provide assessment closely integrated with course work.
- Delay final marking until the end of the course thus allowing people new to programming a little more time to get to grips with it.

While there are many detractors of examinations, the examination does, in many ways emulate the real world of programming, with pressure and time restrictions and was likely to be as accurate a measure of a student's ability as the portfolio (Supovitz & Brennan, 1997). An examination also gives the assurance that the work is that of the individuals.

***Cons***

The portfolio may be difficult for less experienced students in that they have to make up their own problems. The variety of problems also means the students do not have as consistent a set of course knowledge as if they had all tackled the same problems. This variety may also make marking more difficult and time consuming.

An examination is a snap shot of student knowledge and artificial in that they have no access to outside assistance. It could be argued that examinations with set questions are less fair when the students have had a wide choice of problems that they have tackled during the coursework.

### 3.3. Research Goals

This research evaluated portfolio assessment as a method of coursework assessment for a first programming course. It looked at portfolios as a learning and assessment tool, evaluated the student and faculty attitudes to portfolio assessment.

The specific questions the study focused on were:

1) To evaluate portfolios as course work assessment tools.

   Research Questions:

   a) How well do the students meet the learning outcomes described in the syllabus?
   b) Does portfolio assessment effect the completion rate of the course?

2) To evaluate student attitudes to portfolio assessment.

   Research Questions:

   a) What value do students believe portfolio assessment has?
   b) Does portfolio assessment increase the level of student enjoyment in the course?
   c) Does portfolio assessment increase or decrease student workload?

3) To evaluate faculty attitudes to portfolio assessment.

   Research Questions:

   a) From a faculty perspective what are the merits and deficiencies of using portfolio assessment with respect to the following:

- Effect on students' meeting learning outcomes
- Student attitudes
- Teacher attitudes
- Teacher workload

This research is significant in three ways. Firstly, it will enrich the learning environment for the students by encouraging them to more actively control their own learning and to reflect on their progress. Secondly, it will provide information to other computer programming teachers on an alternative assessment strategy. Lastly, it will add to the theoretical knowledge on the application of portfolio assessment.

# 4. Methodology

This investigation best fits a case study approach (Merriam, 1988). There was a mix of qualitative and quantitative data. As befits this type of research, a triangulation of information was sort (Denzin, 1988). The success or other wise, of the portfolios was judged on the basis of the academic results, students' feedback, course teachers' opinions and a review of the work by a colleagues.

## 4.1. Academic Results

The students' course-work versus examination marks were statistically analysed and compared with the results of the previous year (Appendix A). The examination was set at the same level to the previous year so the mark correlation could be a basis of comparison. Course completion rates were compared with the previous year. I am, however, aware that the numbers were small and that student body can vary wildly from one year to another so these comparisons may be of limited validity.

## 4.2. Student Survey

All the students (approximately twenty-five) were asked to participate by completing questionnaires. The students were surveyed using a written questionnaire about two-thirds of the way through the teaching semester and again after the completion of the course. The questionnaire used was a modified section of a Hoyt and Owens (1973) questionnaire cited in Doyle (1975, p.110) (Appendix B). The first questionnaire was conducted during a class and all of the students present that day completed it. The second time it was distributed either in another class, or by mail and the students were asked to return them to the office. This time there was one additional question, the students were asked to indicate the grade they received. The questionnaires were summarised and the following analysis was carried out on each set of questionnaires.

For each of the eleven questions the students rated their agreement and the importance of the question to their learning. Spearman correlation coefficients were calculated for the relationship between the 'agreement' and 'importance' scales for each question. All of the responses were compared by age using likelihood ratios based on contingency tables and Kruskal-Wallis 1-way anova.

The second questionnaire responses were also compared to grade. Gender was not used in any comparative analysis because of the low number of women students. A factor analysis with varimax rotation was used on each set of portfolio and importance questions separately to draw out the central factors.

The results of this analysis are presented in the next section and summary statistics are included in Appendix B. The questionnaire also included a comments section; a summary of these comments is also presented.

### 4.3. Teachers' Opinions

As the course co-ordinator and I took all the lectures. I kept a reflective diary that will be summarised in the following section. There was one other teacher involved in course who took half of tutorials. We held discussions at critical points during the semester. Detailed notes were taken during the discussions and these will be reviewed.

### 4.4. Academic Review

A colleague teaching a higher-level programming course has also reviewed the students' work. His comments are incorporated.

# 5. Results

## 5.1. Academic Results

The comparison of course work marks to examination marks showed that the portfolio-to-examination marks were slightly more closely correlated than the assignment-to-examination marks, however given the sample size and difference this was not statistically significant. The completion rate for the course improved from 69% to 80% and the pass rate from 64% to 71%, but because of the small numbers it is difficult to draw any conclusions from this.

## 5.2. Student Survey

A summary of the student questionnaires is attached in Appendix B. The questionnaire was distributed to students twice. The first time two-thirds of the way through the course just after the students had their second portfolio submission returned. For most questions about 75% of the students rated portfolios the same or more useful than other forms of course work assessment. The exception was development of self-expression skills. One response surprised me, course workload. Two-thirds of the students indicated that they thought that portfolio was the same or less work than the course work they were doing in other courses.

The Spearman correlation coefficient of agreement and importance for each of the eleven questions showed very close correlations between the responses with the exception of the questions on course enjoyment and course workload, which were respectively .43 and .53. It is of interest that these two questions are less related to academic matters than the other questions.

All of the responses were compared to age using likelihood ratios based on contingency tables and Kruskal-Wallis 1-way anova. The contingency tables showed an age related response in the questions on learning fundamental principles and the importance of fundamental principles with the older age groups rating portfolios better and learning fundamental principles more important. The Kruskal-Wallis test indicated the older students rated the learning

of factual knowledge more important than the younger groups. In contrast likelihood ratios and Kruskal-Wallis test indicated that the younger students felt that portfolio assessment encouraged personal responsibility and were more concerned with fairness of marking than their older colleagues. The question on development of self expression skills showed an interesting pattern with the under 20's being spread evenly across the scale from "more than most other courses" to "much less than other courses". All other age groups were concentrated on about average.

The factor analysis with varimax rotation of the portfolio responses drew out four significant factors. The first rated portfolios highly for the application of course material and development of creative capacities. The second factor linked course enjoyment to the lower workload. The third was the development of more accurate self-appraisal, while the fourth factor was the perceived fairness of the interactive marking.

In the factor analysis of the importance of each aspect, the most important factor was personal responsibility and the application of course material. The second factor was marking usefulness and fairness, the third gaining factual knowledge and last workload.

The second survey that was conducted after the course was completed and the results had been published. The raw data indicated that those students who responded (less than the initial survey) were even more positive about portfolios than the initial survey indicated. With the exception of workload and marking fairness, ninety percent or greater of the participating students, rated portfolios equal or better than other assessment methods. (Clearly if those who did not enjoy the course portfolios did not respond to the survey this result could be skewed.)

As with the first survey, the Spearman correlation coefficient of agreement and importance showed close correlations between responses for most questions, however the exceptions this time were in learning of fundamental principles and application of course material. Little can be drawn from these; particularly with

learning of fundamental principles as the overall counts were the same, but many of the respondents rated one higher than the other.

The likelihood ratios based on contingency tables indicated no significant responses by age. The Kruskal-Wallis test again indicated that younger students saw portfolios as assisting them develop personal responsibility.

The factor analysis showed quite a shift in perceptions between the surveys. The most significant factor for portfolios in the second survey was gaining factual knowledge and personal responsibility, neither of which ranked at all in the first survey. These were linked to course enjoyment. The second factor was creativity and application of course material; this was the first factor in the previous survey. Third was learning fundamental principles, not ranked previously, and negatively associated with this was workload, which had previously been a positive part of the second factor. The last factor was self-appraisal skills, which had ranked third earlier.

The importance factors also showed a shift. The most important factor after the course was marking. This was followed by fundamental principles with course enjoyment, self-appraisal skills and lastly gaining factual knowledge.

The comments the students made were generally very positive with them liking the flexibility and several commenting that ‘second chance’ took the pressure off. In the first survey three students commented that it was difficult to come up with ideas, however in the second set of surveys only one made this comment. In both surveys a number of students commented that they liked the interactive marking systems as they felt it had helped clarify their thoughts. Although in the first survey one student commented that she found the marking intimidating, the survey was anonymous so we responded to this comment to the whole group offering ‘private’ marking to anyone who wished. This offer was not taken up and the comment was not repeated in the second survey.

## 5.3. Teachers' Opinion

This section will summarise the teachers' perspective of the portfolio case study. It is based on the journal notes that I took as the course coordinator and the opinion of my fellow teacher (Peter) as we discussed the course and portfolios during and after the course. I will present this information as a time sequence.

The first major undertaking was to prepare the course plan and portfolio package for the students. I undertook this in December 1998. The course plan was a reiteration of the previous offering of the course with information on the portfolio requirements replacing assignments. Preparing the portfolio package was a significant task. It is clear from research that for portfolios to be successful the guidelines and criteria need to be well defined (Arter & Spandel, 1992). It took a lot of time to firstly structure the package, deciding on exactly what major outcomes we wanted the students to achieve and then to fill in the detailed criteria.

The first lecture described the portfolio requirements to the students. There were few questions and the students seemed quite relaxed. Peter had some questions about the requirements in his tutorial and he assured the students they would get plenty of guidance. The second week we covered input, output and arithmetic and the basic data types and I pointed out to students that this was the syntax required in the first section of the first portfolio submission.

I had anticipated that based on the textbook examples and problems the students would be able to make up their own problems in the tutorial sessions. Peter and I quickly found that only a handful of the more able students could do this. We supplemented the textbook with more exercises. In week five we set an exercise that was about the right size and complexity for the 50-line program required for the first submission and then used the model answer from this to demonstrate how to complete the portfolio forms and describe what we would be looking for when marking.

The first submission was marked in week seven. The standard of the work was in the range that we had expected. Peter and I each marked our own tutorial groups and then reviewed them together. We found it very time consuming to mark such a variety of programs. It also took us some time to allocate marks. The marks tended to be clustered around seven with a couple at nine – ten (the maximum) and a couple below five.

At week nine we felt the course was hitting a critical point, the students were preparing their second portfolio submission and had assignments due for other courses. Many appealed for extensions, which we turned down and reminded them that work could be resubmitted at the end of the course. Some of the students were still having more difficulty than we anticipated thinking up their own problems.

The second submission showed some pleasing improvements. Most of the students were able to accurately judge their own performance. Peter and I felt more relaxed about the marking process, although we still tended to cluster the grades. At this point of the semester, Peter had some reservations about the use of portfolios for the weaker students. He felt that they would probably manage better if they were more directed.

The final part of the course was a consolidation with the emphasis on writing ‘good’ code. For the third submission the students were more directed; we had provided three data files from which they selected one to use. We found the better students used more imagination and incorporated more varied routines. This submission also included extension work. We actively encouraged the more able students to write a program for this section. They all wrote games, noughts and crosses or connect four were popular. Two students decided to work together, something we had not anticipated but we negotiated a marking system with them.

In the final submission about half of the students chose to resubmit earlier work. Most students improved their earlier work substantially. Students who had got more than 80% on the first attempt tended to spend their time on extension work,

rather than rework, which seemed to us an appropriate allocation of time. We were suspicious that a couple of students had someone else do some of their work. One less able student, copied a program from a book and put it in as extension work. Peter simply did not mark it, as the student could not explain it. A couple of other less able students chose to do extension work, rather than rework that we believed would be of more benefit to them.

We set the examination at the same level as the previous offering of the course and the students performed at the expected level.

# 6. Analysis of Results

## 6.1. Academic Results

In earlier sections, I discussed the two-fold purpose of assessment; as an aid to learning and for measuring the knowledge of students. With reviewing academic results we are evaluating how well the assessment measured the students' knowledge. The close correlation between examination marks and portfolio indicates a consistency. It seems reasonable to assume that both are fair measures of student knowledge. This is consistent with the findings of Supovitz and Brennan (1997).

We believe that some of the weaker students probably passed the course with portfolios where previously they would have either dropped-out or failed. The minimum mark requirement in the examination leaves us confident that they have reached a level of basic competence. While the higher completion rate and pass rate is pleasing on one level we have noticed that the weaker students who just managed to pass are struggling with the follow-on course and at the time of writing seem unlikely to pass.

There are contradictory arguments with this. One could argue that this course is compulsory where the subsequent course is not, so one could argue that students have learnt enough programming to complete the compulsory requirements of the degree and can avoid all further programming. Or, one could argue that a strong knowledge of programming if fundamental to an IS degree and therefore the students who just managed to scrape through should repeat the course. If the latter were the case the obvious solution would be to impose a higher standard in this course. At this point the programme committee that is responsible for the academic management of the degree has taken the former approach.

For the average and better students, the course has produced similar learning outcomes to previous courses. Most of these students are enrolled in the follow-on course and their teacher has not observed any significant difference.

## 6.2. Students' Perceptions

The research questions stated earlier on student attitudes were.

- What value do students believe portfolio assessment has?
- Does portfolio assessment increase the level of student enjoyment in the course?
- Does portfolio assessment increase or decrease student workload?

It seems clear from the survey results that the students rated portfolios highly as a learning tool. They clearly found them more useful than assignments or case studies for the application of course material. A typical comment was

> *Portfolios are a good approach to learning because it gives you more flexibility in learning skills and principles and also allows more creativity*

However many commented that it wasn't easy, especially thinking up their own tasks. This was another typical comment

> *Probably more interesting than having the topic set for you, but also quite hard to think of a topic you can do with the skills learned so far.*

The variations in response due to age in the question about learning underlying principles in the first questionnaire could be explained in reference to Perry's (1970) or Belenky and colleagues (1986) intellectual development. The older students, all of whom held tertiary qualifications, were probably operating at a 'procedural knowledge level', consciously, deliberately and systematically obtaining and communicating knowledge and appreciated the need to learn the principles underlying a discipline new to them. While the less mature students are more likely to be operating at the 'subjective knowledge level', simply accepting what is laid before them.

As portfolios affected their general learning, they were again very positive, however it is fair to comment that the students did not perceive portfolios as

giving much opportunity for oral or written self-expression – which given the tasks they were required to do, is not surprising (no presentations or written reflections were required). Generally their programming skills were not at a level where they were able to create new interactive games or other programs that they would classify as 'self-expression'. Developing personal responsibility and self-appraisal skills came through as a big plus for portfolios. This was particularly evident with the younger students. It is likely that personal responsibility is an issue for them operating at Perry's (1970) or Belenky's (1986) lower levels, where personal responsibility is the norm for the more mature student.

The students' rating of course enjoyment showed interesting developments. At the mid point the mean score for enjoyment was below the 'average' for a course and enjoyment did not rate as one of the important indicators in the factor analysis. After the course, the mean score was between *much more than most* and *more than most*, with no one scoring the enjoyment as less than average and enjoyment was part of the second factor in the importance scale. As noted earlier this result could be skewed if only those who enjoyed the course returned the questionnaire.

The response to the workload question in the first questionnaire surprised me, with a lot of students rating the workload as less than other courses, however this was reversed in the second questionnaire with many rating the workload as more than other courses.

One of the issues that had concerned us as teachers was being able to mark fairly over a wide range of work and at the time of the first questionnaire this was also a concern for the younger students. The older students were less worried with the fairness of marking. Possibly they no longer viewed the world from Perry's (1970) position of basic duality where there is a need for 'right' and 'wrong' answers. These concerns were not evident in the second questionnaire. However marking is clearly a very important issue for students as the two questions on marking rated as the first factor in the *after course* questionnaire and the second factor in the *during course* questionnaire. One could conclude, at the end of the course the only evidence of success or failure a student has is the knowledge in

his or her head, and a grade. These responses would clearly support the hypotheses of Popham (1995) and Biggs (1998) that the way to better learning is better assessment, as this is where the students will concentrate their effort.

### 6.3. Teachers' Experience

From our perspective the students did meet the learning outcomes. The first portfolio submission we judged to be a generally better effort than the previous first assignments and we had a higher student retention rate than previous courses. There were two things we decided needed changing. Firstly, we asked them to number the lines in their programs. I had thought the compiler had this facility, it didn't. They used MS Word to number the lines and therefore lost the indenting that is crucial to the readability of program code. Secondly, we did not set an upper limit on the number of programs that could be submitted. The poorest submissions had the most programs. The students had done no selection at all, and simply put in most of the programs they had written. These students had not engaged in the selection process that is clearly so important with portfolio assessment (Arter & Spandel, 1992; Lester & Kroll, 1991). We decided a limit of three – four programs would be appropriate.

With the second submission we decided we could have given more suggestions and guidance on possible programming problems and would do so for further offerings of the course. From the final submission where some students inappropriately opted to do extension work we concluded that it would be a good idea to put in a requirement of at least 60% for submissions one and two, to be allowed to submit extension work.

Peter and I spent sometime reviewing the course. We discussed how we felt the portfolios had contributed to the students' learning. We both felt that it had been very successful for the more mature or able members of the group. They could clearly see wider application for their newly acquired skills either from their experience or better comprehension of the material. Peter was particularly concerned with the inability of the some of the less competent students to create

their own problems. I had also found this a problem, but was less concerned with it. I believe that several of these students would have withdrawn from the course if it had been offered in the normal way and they had received low marks that were “locked in” for the first assignments.

One aspect that had always concerned us with new programming students was that they did not master the basic syntax quickly enough. This was evident right up to the conclusion of the course, with some students still making basic mistakes and unable to correct them themselves. Portfolio assessment had done nothing to help this problem. These students were experiencing cognitive overload (Edwards, 1991; Evans, 1991b) that severely impacted on their ability to cope with the course.

We both felt that the portfolio had a positive effect on student attitudes. This was clearest with the most able students and the extension work that they submitted. There was a feeling of ownership of their portfolio work which was apparent when they discussed their plans and work with us. Our feelings on their positive attitude were clearly supported by the comments some of them made in the questionnaires. One student commented in the second questionnaire

> *I learnt a lot from this course and think the most was learned during the portfolio because it helped my understanding. Thank you for being great teachers.*

Peter and I both found it quite difficult to take this course with such a radical change in the assessment, especially as it is part of a relatively new degree programme. I worried that the students would have the appropriate skill to pass the final examination where the questions were fixed, when they had completed different problems in the coursework. These fears proved to be groundless. It was great to see the ideas that the students came up with. This was one of the highlights for us.

This first offering of a portfolio course was a lot more work for us. The preparation of the portfolio package for students took a lot of research and time. I

hadn't written a portfolio package before and so it is reasonable to assume that it would be much quicker another time. The course was offered again in the second semester of 1999 and I found that the initial time had been well invested. I made a few alterations based on our observations above, but other than that it did not need the rework and rethinking of the new assignment questions of the past.

Secondly, we both found the marking difficult. We wanted to be fair and consistent and this we found difficult with such a variety of work. Traditionally we have been able to quantify our marking quite accurately – x marks for doing y correctly. While we used the criteria specified as a guide we had to take a more holistic view of the student's work. We both developed more confidence in our intuitive feel towards the end of the semester as we found we were consistently awarding the same marks and a colleague who reviewed some of the work concurred with our marking.

At the time of writing I am teaching the next offering of the course, again using portfolio assessment. I have found the marking easier and quicker this time. It required a conscious effort to spread the grades more evenly, but this too becomes easier with practice. I am now confident of my ability to mark consistently and am confident that the students will have gained appropriate knowledge to pass the final.

## 7. Conclusions

This case study set out to examine the effectiveness of portfolio assessment in an introductory computer programming course. I examined the cognitive requirements of programming and the literature on learning to program. From this literature it is clear that programming is a cognitively demanding task and that learning to program is difficult. The novice programmer must master the programming language, algorithm construction and problem decomposition. At the same time as teaching students to program as part of an undergraduate degree in information systems the course is endeavouring to develop the student's thinking and learning skills.

Portfolio assessment is intended to promote learning at the higher cognitive levels and encourage students to develop as learners. The evidence collected from the student questionnaires and the teacher observations would support these claims of portfolio assessment. The students commented favourably about being able to select their assessable work themselves and many of the younger students noted that personal responsibility was an important learning experience in this course. The teachers found that the portfolio created a positive learning environment.

The learning outcomes for the average students were comparable with earlier offerings of the course using assignments. We believed that some of the weaker students who completed and passed the course would have previously dropped-out. The course also had a final examination and examination–to–portfolio marks were similar to previous examination–to–assignment marks. This would suggest that portfolios are as effective a measure of student knowledge as examinations or assignments. In the tertiary environment that this study was carried out in, controlled assessments are necessary to verify the individual's work as there is the same difficulty with portfolios as with assignments with students not handing in their own work.

The teachers found the transition to portfolio assessment quite heavy in preparation the first time the course was offered, however a subsequent offering had lower than normal preparation. We also found the marking difficult to start with, but this also became easier with experience. One area that we would like to encourage more student effort is in the rote learning of basic syntax. Portfolios did not assist with this.

The students' response to the questionnaires, particularly the second questionnaire reinforced my belief that one of the keys to better learning in our environment is better assessment. Portfolio assessment provides some unique benefits as a learning and assessment strategy that will have benefits in our programmes.

## 8. Recommendations and Further Directions

Clearly portfolios are a useful assessment and learning tool for programming courses. They may be even more successful in higher level programming courses when the students are able to tackle a larger and more diverse range of problems.

An ongoing challenge in this particular course is to encourage the students to learn the basic programming language well so that they can then concentrate on the more difficult tasks without cognitive overload. Methods to accomplish promote this are a further challenge worth investigation.

## 9. References


Anderson, J. (1982). Aquisition of cognitive skills. *Psychological Review, 89,* 359-406.

Angelo, T. A. (1994). Classroom Assessment: Involving Faculty and Students Where It Matters Most. *Progress, Trends and Practices in Higher Education, 6*(4), 1-2,5,10.

Arter, J. A., & Spandel, V. (1992). Using portfolios of student work in instruction and assessment. *Educational Measurement: Issues and Practice, 11,* 36-11.

Belenky, M. F., Clinchy, B. M., Goldberger, N. R., & Tarule, J. M. (1986). *Women's ways of knowing: The development of self, voice and mind*. New York: Basic Books.

Berson, J., Engelkermeyer, S., Oliaro, P. M., Potter, D. L., Terenzini, P. T., & Walker-Johnson, G. M. (1998). *Powerful partnerships: A shared responsibility for learning* (Joint Report ): American Association for Higher Education.

Biggs, J. (1998). *What the student does: Teaching for enhanced learning in the '90s.* Paper presented at the Higher Education Research and Development Society of Australasia, Auckland.

Blum, B. (1992). *Software engineering: A holistic view*. New York: Oxford University Press.

Booth, S. (1990). *Concepts of programming: A study into learning to program*. Available: http://ericae.net/ericdb/ED338227.htm.

Boughey, J., & Searle, R. (1998). *Reflecting on and sharing practice: The design and implementation of a regional master's degree course in tertiary studies.* Paper presented at the Higher Education Research and Development Society of Australasia, Auckland.

Boulton-Lewis, G. M., & Halford, G. S. (1991). Processing capacity and school learning. In G. Evans (Ed.), *Learning and teaching cognitive skills* . Hawthorne: The Australian Council for Educational Research.

Carpenter, T. P. (1980). Research in cognitive development,. In S. RJ (Ed.), *Research in Mathematics Education.* . Reston, VA: National Council of Teachers of Mathematics.

Denzin, N. K. (1988). Triangulation. In K. J.P (Ed.), *Educational Research, methodology, and measurement: An international handbook* (pp. 511-513). Sydney: Peramon Press.

Dijkstra, E. W. (1998). On the cruelty of really teaching computing science. *Communications of the ACM, 32*(12), 1398-1413.

Doyle, K. O. (1975). *Student evaluation of instruction*. Lexington: Lexington Books.

Edwards, J. (1991). The direct teaching of thinking skills. In G. Evans (Ed.), *Learning and teaching cognitive skills* . Hawthorne: The Australian Council for Educational Research.

Evans, G. (1991a). *Learning and teaching cognitive skills*. Hawthorne: The Australian council for educational research.

Evans, G. (1991b). Lesson cognitive demand and student processing in upper secondary mathematics. In G. Evans (Ed.), *Learning and teaching cognitive skills* . Hawthorne: The Australian council for educational research.

Gardener, H. (1992). Assessment in context: The alternative to standardized testing. In B. R. Gifford & M. C. O'Connor (Eds.), *Changing Assessment: Alternative views of aptitude, achievement and instruction* (pp. 77-119). Boston: Klower.

Glaser, R. (1984). Education and thinking: The role of knowledge. *American Psychologist, 39*, 93-104.

Gonzalez, A. J., & Dankel, D. D. (1993). *The engineering of knowledge based systems.* New Jersey: Prentice Hall.

Hager, P., & Butler, J. (1996). Two models of educational assessment. *Assessment and Evaluation in Higher Education, 21*(4), 367-378.

Herman, J. L. (1992). What research tells us about good assessment. *Educational Leadership, 49*, 74-78.

Kerka, S. (1995). *Techniques for authentic assessment.* Available: http:www.wa-wbl.com/resources_educators/eric.

Knight, P. (1995). *Assessment of learning*. London: Kogan Page Ltd.

Lammers, S. (1996). *Programmers at work: Interviews with 19 programmers who shaped the computer industry*. Redmond: Microsoft Press.

Lester, F. K., & Kroll, D. L. (1991). Evaluation: A new vision. *Mathematics teacher, 84*, 276-283.

Linn, M. C. (1985). The cognitive consequences of programming instruction in classrooms. *Educational Researcher, 14*(5), 14-16, 25-29.

Linn, M. C., & Clancy, M. J. (1992). Case for case studies of programming problems. *Communications of the ACM .Mar 1992; v35n3, pp. 121-132, 35*(3), 121-132.

Maheshwari, P. (1997). Improving the learning environment in first-year programming: Integrating lectures, tutorials and laboratories. *Journal of Computers in Mathematics and Science Teaching, 16*(1), 111-131.

Mann, L. (1979). *On the trail of process*. New York: Grune and Stratton.

McDowell, E. (1999). Editorial. *Assessment & Evaluation in Higher Education, 23*(4), 335-338.

McGill, T. J., & Volet, S. E. (1997). A conceptual framework for analyzing students' knowledge of programming. *Journal of Research on Computing in Education, 29*(3), 276-289.

Merriam, S. B. (1988). *Case study research: A qualitative approach*. San Fransisco: Jossey Bass.

Milbrandt, G. (1995). Using problem solving to teach a programming language. *Learning and Leading with Technology, October*, 27-31.

Nickerson, R. S. (1985). Understanding understanding. *American Journal of Education, 93*(2), 201-239.

Perry, W. J. (1970). *Forms of intellectual and ethical development in the college years: A scheme*. New York: Holt, Rinehart and Winston, Inc.

Plimmer, B. E. (1999). *But is it a new environment out there?* Paper presented at the National Advisory Committee on Computing Qualifications, Dunedin.

Popham, W. J. (1995). Farewell, curriculum: Confessions of an assessment convert. *Phi Delta Kappan, 79*, 380-384.

Renwick, L. B., & Renwick, D. P. (1992). Assessing the thinking curriculum: New tools for educational reform. In B. R. Gifford, O'Connor, M.C. (Ed.), *Changing assessments: Alternative views of aptitude, achievement and instruction* (pp. 37-75).

Ryle, G. (1949). *The concept of the mind*. London: Hutchinson.

Shneiderman, B. (1980). *Software psychology*. Cambridge, Massachusetts: Winthrop Publishers.

Soloway, E., & Iyengar, S. (Eds.). (1986). *Workshop on empirical studies of programmers*. New Jersey: Ablex Publishing Corporation.

Sumsion, J., & Fleet, A. (1996). Reflection: can we assess it? Should we assess it? *Assessment & Evaluation in Higher Education, 21*(2), 121-130.

Supovitz, J. A., & Brennan, R. T. (1997). Mirror, mirror on the wall, which is the fairest test of all? An examination of the equitability of portfolio assessment relative to standardized test. *Harvard Educational Review, 67*(3), 472-502.

Taylor, J. (1991). Designing instruction to generate expert cognitive skill performance: empirical evidence. In G. Evans (Ed.), *Learning and teaching cognitive skills* . Hawthorne: The Australian council for educational research.

Thomas, R. A., & Upah, S. C. (1996). Give programming instruction a chance. *Journal of Research on Computing in Education, 29*(1), 96-108.

Wolf, D. P., LeMahieu, P. G., & Eresh, J. (1992). Good measure: Assessment as a tool for educational reform. *Educational Leaders, 49*, 8-13.

Woolfolk, A. E. (1993). *Educational psychology (5th Ed)*, Boston: Allyn and Bacon.

# 10. Appendixes

## 10.1. Appendix A

The two sets of marks were entered into a spreadsheet. Pearson correlation coefficient was calculated for each set of marks. Fisher's transformation was then applied to each coefficient to produce a new variable that has an approximately normal distribution.

Fisher's transformation is

$$z = \tfrac{1}{2}\log\frac{1+r}{1-r}$$

This produces z that is approximately normal with a mean of

$$\tfrac{1}{2}\log\frac{1+\rho}{1-\rho}$$

And standard deviation of $\dfrac{1}{\sqrt{n-1}}$

The null hypothesis is that $H_0 = \rho_1 = \rho_2$

and the test statistic is

$$\frac{z_1 - z_2}{\dfrac{1}{\sqrt{n_1 - 3}} + \dfrac{1}{\sqrt{n_2 - 3}}}$$

The resulted was .3088 of a standard deviation – implying that there was no significant difference between the two correlations. The data was also tested for outliers using minitab, these were removed and the calculations redone, the results were still not statistically significant.

| Raw Data | | | | | Data with outliers removed | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 1998 Assignment Assessment | | | 1999 Portfolio Assessment | | 1998 Assignment Assessment | | | 1999 Portfolio Assessment | |
| Examination | Course Work | | Examination | Course Work | Examination | Course Work | | Examination | Course Work |
| 37.8 | 58.5 | | 44.8 | 43.7 | 29.9 | 46.5 | | 44.8 | 43.7 |
| 29.9 | 46.5 | | 41.6 | 45.3 | 32.6 | 42.5 | | 41.6 | 45.3 |
| 32.6 | 42.5 | | 36.8 | 42.4 | 30.0 | 45.0 | | 36.8 | 42.4 |
| 30.0 | 45.0 | | 34.1 | 44.2 | 26.3 | 48.5 | | 34.1 | 44.2 |
| 26.3 | 48.5 | | 35.8 | 37.2 | 28.6 | 39.0 | | 35.8 | 37.2 |
| 28.6 | 39.0 | | 36.3 | 36.5 | 26.6 | 39.5 | | 36.3 | 36.5 |
| 26.6 | 39.5 | | 39.3 | 33.3 | 27.0 | 39.0 | | 39.3 | 33.3 |
| 27.0 | 39.0 | | 34.3 | 37.5 | 28.5 | 37.5 | | 34.3 | 37.5 |
| 28.5 | 37.5 | | 35.3 | 36.5 | 22.1 | 43.5 | | 35.3 | 36.5 |
| 22.1 | 43.5 | | 31.8 | 39.7 | 24.5 | 40.5 | | 31.8 | 39.7 |
| 24.5 | 40.5 | | 32.7 | 38.3 | 26.0 | 36.0 | | 32.7 | 38.3 |
| 26.0 | 36.0 | | 32.8 | 37.1 | 26.6 | 34.5 | | 32.8 | 37.1 |
| 26.6 | 34.5 | | 35.8 | 33.8 | 23.8 | 37.0 | | 35.8 | 33.8 |
| 23.8 | 37.0 | | 31.3 | 37.8 | 17.8 | 42.5 | | 31.3 | 37.8 |
| 17.8 | 42.5 | | 36.4 | 32.6 | 20.0 | 40.0 | | 36.4 | 32.6 |
| 20.0 | 40.0 | | 31.4 | 36.6 | 23.2 | 32.5 | | 31.4 | 36.6 |
| 23.2 | 32.5 | | 28.9 | 37.9 | 16.5 | 35.5 | | 28.9 | 37.9 |
| 13.6 | 39.0 | | 34.4 | 31.0 | 17.8 | 32.5 | | 34.4 | 31.0 |
| 16.5 | 35.5 | | 26.2 | 37.2 | 17.8 | 32.5 | | 26.2 | 37.2 |
| 17.8 | 32.5 | | 27.6 | 35.7 | 23.7 | 26.5 | | 27.6 | 35.7 |
| 17.8 | 32.5 | | 30.5 | 32.4 | 26.9 | 23.0 | | 30.5 | 32.4 |
| 23.7 | 26.5 | | 25.0 | 34.5 | | | | 25.0 | 34.5 |
| 26.9 | 23.0 | | 20.8 | 36.5 | | | | 20.8 | 36.5 |
| 19.2 | 15.5 | | 22.7 | 32.0 | | | | 22.7 | 32.0 |
| 8.8 | 25.5 | | 16.6 | 38.0 | | | | | |
| | | | 29.1 | 23.9 | | | | | |
| | | | 15.9 | 23.9 | | | | | |
| | | | | | | | | | |
| Contribution to grade | | | | | | | | | |
| 40.0 | 60 | | 50 | 50 | 40.0 | 60 | | 50 | 50 |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Pearson Co-relation Co-efficient | | | | | | | | |
| | 0.5392 | | 0.4714 | | | 0.2800 | | 0.4156 |
| Fisher's Transformation | | | | | | | | |
| | 0.6030 | | 0.5119 | | | 0.2877 | | 0.4424 |
| Hypothesis Test | | | | | | | | |
| | 0.3089 | | | | | -.4818 | | |

## 10.2. Appendix B

Student Questionnaire – Portfolio Assessment

The software development principles course that you are enrolled in is using a portfolio approach to course work assessment. Over the semester you are being required to compile a collection of your work as evidence of your learning.

This questionnaire asks for your opinion of this approach. Specifically your beliefs on how the compilation of your portfolio has effected your learning of software development and learning in general. You are also asked to comment on your enjoyment of the course, the course workload and portfolio marking. Please use as comparison other courses of a similar level in which you are currently enrolled or have completed in the past.

This questionnaire is anonymous and only summary information will be reported. Participation is voluntary. You may withdraw at any time. Your participation or non-participation in the completion of this questionnaire will have no effect on your course or results.

For each statement please rate:
your agreement or disagreement in the first scale and then
how **important** you believe this factor to be to your learning in the **second scale**.

| | Importance | | | | | Agreement | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Much more than | More than | About average | Less than | Much less than | Very important | Important | Undecided | Unimportant | Very unimportant |
| Effect of Portfolios As a Learning Experience | | | | | | | | | | |
| a) Programming Specific | | | | | | | | | | |
| Usefulness to learning factual knowledge | | | | | | | | | | |
| Usefulness to learning fundamental principles, generalisations and theories | | | | | | | | | | |
| Usefulness to learning how to apply course material to improve rational thinking, problem solving and decision making | | | | | | | | | | |
| b) General development | | | | | | | | | | |
| Usefulness to developing creative capacities | | | | | | | | | | |
| Usefulness to developing skills in expressing my self orally and or in writing | | | | | | | | | | |
| Usefulness to developing a sense of personal responsibility (self reliance, self discipline) | | | | | | | | | | |
| Usefulness to developing more accurate self- appraisal skills | | | | | | | | | | |
| Effect of Portfolio on Course Enjoyment | | | | | | | | | | |
| Preparing my portfolio was enjoyable | | | | | | | | | | |
| Effect of Portfolio on Course Workload | | | | | | | | | | |
| The work involved in preparing my portfolio was roughly equivalent to course work in other courses | | | | | | | | | | |
| Interactive Marking | | | | | | | | | | |
| The interactive marking was useful | | | | | | | | | | |
| The interactive marking was fair | | | | | | | | | | |

Demographics
Please circle Gender M / F
Age Group <20 20-29 30-39 40+

**General Comments**

_____
_____
_____
_____
_____

Thank you

## 10.3. Student Questionnaires

Summary of results Questionnaire 1

| Effect of Portfolios As a Learning Experience | Portfolio | | | | | | | Importance | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Much more than | More than | About average | Less than | Much less than | No response | | Very important | Important | Undecided | Unimportant | Very unimportant | No response |
| **a) Programming Specific** | | | | | | | | | | | | | |
| Aid to gaining factual knowledge | 3 | 11 | 8 | 1 | | 1 | | 3 | 18 | 2 | 1 | | |
| Aid to learning fundamental principles, generalisations and theories | 3 | 10 | 6 | 4 | | 1 | | 6 | 13 | 4 | | | 1 |
| Aid to learning how to apply course material to improve rational thinking, problem solving and decision making | 7 | 10 | 3 | 3 | 1 | | | 10 | 8 | 2 | 2 | | 1 |
| **b) General development** | | | | | | | | | | | | | |
| Aid to developing creative capacities | 5 | 14 | 2 | 1 | | 2 | | 11 | 8 | 3 | | | 2 |
| Aid to developing skills in expressing my self orally and or in writing | | 3 | 12 | 4 | 3 | 2 | | 1 | 9 | 7 | 3 | 2 | 2 |
| Aid to developing a sense of personal responsibility (self reliance, self discipline) | 4 | 11 | 4 | 1 | 2 | 2 | | 5 | 9 | 4 | 1 | 2 | 3 |
| Aid to developing more accurate self- appraisal skills | 3 | 8 | 10 | | 2 | 1 | | 3 | 14 | 5 | | 1 | 1 |
| **Effect of Portfolio on Course Enjoyment** | | | | | | | | | | | | | |
| Preparing my portfolio enhanced the course enjoyment | | 4 | 12 | 6 | 2 | | | 3 | 12 | 7 | 1 | 1 | |
| **Effect of Portfolio on Course Workload** | | | | | | | | | | | | | |
| The work involved in preparing my portfolio was | | 6 | 6 | 7 | 3 | 2 | | 2 | 11 | 5 | 3 | | 3 |
| **Interactive Marking** | | | | | | | | | | | | | |
| Perceived Usefulness | 7 | 9 | 5 | 2 | 1 | | | 7 | 9 | 3 | 4 | 1 | |
| Perceived Fairness | 4 | 6 | 10 | 3 | 1 | | | 8 | 9 | 3 | 3 | | 1 |

| **Demographics Please circle** | **Gender** | **M / F** | **Age Group** | **<20** | **20-29** | **30-39** | **40+** |
|---|---|---|---|---|---|---|---|
| | | 21/1/1 | | 9 | 6 | 4 | 4 |

Summary Statistics Survey 1

| | Correlation between portfolios and usefulness Significance | Portfolios | | | | | | Importance | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Kruskal-Wallis 1-Way Anova by Age | | Rotated Factor Analysis | | | | Kruskal-Wallis 1-Way Anova by Age | | Rotated Factor Analysis | | | |
| | | Chi-Square | Significance | Factor 1 | Factor 2 | Factor 3 | Factor 4 | Chi-Square | Significance | Factor 1 | Factor 2 | Factor 3 | Factor 4 |
| Aid to gaining factual knowledge | .011 | 2.23 | .52 | .56 | -.00 | .61 | -.23 | 6.63 | .084 | .02 | -.04 | .88 | -.08 |
| Aid to learning fundamental principles, generalisations and theories | ..086 | 1.84 | .60 | .58 | .49 | -.13 | -.48 | 5.30 | .151 | .52 | .03 | .55 | .38 |
| Aid to learning how to apply course material to improve rational thinking, problem solving and decision making | .001 | 2.35 | .50 | .80 | .18 | .29 | .14 | 3.13 | .371 | .82 | .23 | .16 | -.06 |
| **b) General development** | | | | | | | | | | | | | |
| Aid to developing creative capacities | .110 | 4.27 | .23 | .69 | -.64 | -.01 | .15 | .777 | .854 | .32 | .66 | -.15 | .10 |
| Aid to developing skills in expressing my self orally and or in writing | .0 | .514 | .91 | .40 | .70 | .27 | .17 | 1.07 | .782 | .75 | .02 | -.12 | .01 |
| Aid to developing a sense of personal responsibility (self reliance, self discipline) | .0 | 7.82 | .04 | .35 | -.15 | .67 | .24 | 4.31 | .229 | .86 | .08 | .04 | .07 |
| Aid to developing more accurate self- appraisal skills | .008 | .674 | .87 | .00 | .20 | .94 | -.09 | 1.69 | .637 | .63 | -.06 | -.62 | .19 |
| Effect of Portfolio on Course Enjoyment | | | | | | | | | | | | | |
| Preparing my portfolio enhanced the course enjoyment | .428 | 4.62 | .20 | .03 | .81 | -.36 | -.07 | 5.37 | .146 | -.00 | .15 | -.43 | .76 |
| Effect of Portfolio on Course Workload | | | | | | | | | | | | | |
| The work involved in preparing my portfolio was | .53 | 1.70 | .63 | -.14 | .79 | .34 | .13 | 4.38 | .223 | -.05 | -.00 | .11 | .88 |
| Interactive Marking | | | | | | | | | | | | | |
| Perceived Usefulness | .0 | .799 | .84 | .54 | .12 | -.11 | .62 | 1.62 | .654 | .08 | .80 | .05 | .26 |
| Perceived Fairness | .0 | 1.82 | .60 | .09 | .07 | .16 | .92 | 2.16 | .539 | -.05 | .81 | .03 | -.18 |

Summary of results Questionnaire 2

| Effect of Portfolios As a Learning Experience<br>a) Programming Specific | Portfolio<br>Much more than | More than | About average | Less than | Much less than | No response | | Importance<br>Very important | Important | Undecided | Unimportant | Very unimportant | No response |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Aid to gaining factual knowledge | 5 | 6 | 8 | | | | | 9 | 7 | 1 | 2 | | |
| Aid to learning fundamental principles, generalisations and theories | 2 | 9 | 7 | 1 | | | | 2 | 9 | 7 | 1 | | |
| Aid to learning how to apply course material to improve rational thinking, problem solving and decision making | 2 | 11 | 4 | 2 | | | | 5 | 11 | 2 | 1 | | |
| **b) General development** | | | | | | | | | | | | | |
| Aid to developing creative capacities | 5 | 7 | 7 | | | | | 6 | 7 | 6 | | | |
| Aid to developing skills in expressing my self orally and or in writing | 1 | 2 | 14 | 2 | | | | 3 | 5 | 7 | 3 | 1 | |
| Aid to developing a sense of personal responsibility (self reliance, self discipline) | 6 | 8 | 4 | 1 | | | | 4 | 11 | 3 | 1 | | |
| Aid to developing more accurate self- appraisal skills | 2 | 12 | 4 | 1 | | | | 4 | 10 | 5 | | | |
| Effect of Portfolio on Course Enjoyment | | | | | | | | | | | | | |
| Preparing my portfolio enhanced the course enjoyment | 6 | 9 | 4 | | | | | 4 | 11 | 2 | 2 | | |
| Effect of Portfolio on Course Workload | | | | | | | | | | | | | |
| The work involved in preparing my portfolio was | 3 | 6 | 7 | 3 | | | | 3 | 12 | 4 | | | |
| Interactive Marking | | | | | | | | | | | | | |
| Perceived Usefulness | 5 | 9 | 3 | 1 | 1 | | | 5 | 7 | 5 | 1 | 1 | |
| Perceived Fairness | 4 | 9 | 2 | 4 | | | | 3 | 9 | 5 | 2 | | |

Demographics

**Please circle Gender M / F** **Age Group <20 20-29 30-39 40+**

18/1 8 4 6 1

Grades

| A | B | C | D | E |
|---|---|---|---|---|
| 1 | 10 | 6 | 1 | 1 |

Summary Statistics Survey 2

| | Correlation between portfolios and usefulness Significance | Portfolios: Kruskal-Wallis 1-Way Anova by Age Chi-Square | Portfolios: Kruskal-Wallis 1-Way Anova by Age Significance | Portfolios: Rotated Factor Analysis Factor 1 | Portfolios: Rotated Factor Analysis Factor 2 | Portfolios: Rotated Factor Analysis Factor 3 | Portfolios: Rotated Factor Analysis Factor 4 | Importance: Kruskal-Wallis 1-Way Anova by Age Chi-Square | Importance: Kruskal-Wallis 1-Way Anova by Age Significance | Importance: Rotated Factor Analysis Factor 1 | Importance: Rotated Factor Analysis Factor 2 | Importance: Rotated Factor Analysis Factor 3 | Importance: Rotated Factor Analysis Factor 4 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Aid to gaining factual knowledge | .107 | 3.53 | .31 | .89 | .10 | .01 | -.23 | 2.53 | .46 | -.02 | .01 | -.16 | .88 |
| Aid to learning fundamental principles, generalisations and theories | .330 | 2.54 | .46 | .29 | -.00 | .80 | .24 | 2.78 | .42 | .03 | .77 | -.05 | .18 |
| Aid to learning how to apply course material to improve rational thinking, problem solving and decision making | .201 | 1.57 | .66 | -.01 | .75 | .02 | .08 | .79 | .85 | .16 | .48 | .16 | .65 |
| **b) General development** | | | | | | | | | | | | | |
| Aid to developing creative capacities | .013 | 1.06 | .78 | .27 | .82 | .05 | -.10 | 1.75 | .62 | .55 | .59 | -.08 | .16 |
| Aid to developing skills in expressing my self orally and or in writing | .006 | 7.9 | .04 | -.09 | .66 | .12 | .14 | 1.10 | .77 | .46 | .25 | .04 | .39 |
| Aid to developing a sense of personal responsibility (self reliance, self discipline) | .000 | 2.25 | .52 | .80 | -.13 | .00 | .05 | 3.4 | .32 | -.17 | .60 | -.60 | -.08 |
| Aid to developing more accurate self- appraisal skills | .037 | 5.4 | .14 | -.17 | .11 | .06 | .92 | 3.3 | .33 | -.16 | -.15 | .90 | -.05 |
| Effect of Portfolio on Course Enjoyment | | | | | | | | | | | | | |
| Preparing my portfolio enhanced the course enjoyment | .007 | 1.91 | .59 | .75 | .38 | .15 | .03 | 5.54 | .13 | .69 | -.01 | .00 | .25 |
| Effect of Portfolio on Course Workload | | | | | | | | | | | | | |
| The work involved in preparing my portfolio was | .032 | 1.7 | .63 | .12 | -.11 | -.85 | .12 | .19 | .97 | .50 | .38 | .67 | -.12 |
| Interactive Marking | | | | | | | | | | | | | |
| Perceived Usefulness | .003 | 4.2 | .23 | .35 | .52 | -.13 | .29 | 3.08 | .37 | .88 | -.16 | -.06 | .13 |
| Perceived Fairness | .039 | .84 | .83 | .50 | .53 | -.02 | .55 | 2.9 | .39 | .83 | .33 | .22 | -.02 |