Linuxgym: A sustainable and easy-to-use automated developmental assessment tool for computer scripting skills

Final project report **December 2008**

Project Team

Lead Institution

UNIVERSITY OF TECHNOLOGY, SYDNEY

Dr. Andrew Solomon, Prof. Jenny Edwards (Project Leaders)
Dr. Raymond Lister

Partner Institutions

UNIVERSITY OF SYDNEY

Assoc. Prof. Judy Kay

UNIVERSITY OF NEW SOUTH WALES

Dr. John Shepherd

Web sites

http://linuxgym.sourceforge.net/ http://linuxgym.com/

Acknowledgements

Support for this project has been provided by the Australian Learning and Teaching Council, an initiative of the Australian Government Department of Education, Employment and Workplace Relations. The views expressed in this report do not necessarily reflect the views of the Australian Learning and Teaching Council Ltd.

2008

Executive Summary

While 10% of all jobs advertised in the IT/telecommunications sector seek scripting as a skill, it is to the detriment of both industry and education that it has been very difficult to teach and learn. Developing scripting skills is like learning a language, or how to drive a car — it requires constant practice and feedback. However, with universities nowadays having high student:staff ratios, there is very little scope for providing the necessary feedback. As a result, within at least one of the participating universities (Solomon et al., 2006) and more generally (Bennedsen and Caspersen, 2007), coding courses have typically had a failure rate of over 35%.

On arriving at the University of Technology, Sydney in 2001, Andrew Solomon identified these problems. In response he developed the alpha version of Linuxgym, software which automates assessment and feedback of scripting skills. The use of this software has increased both the depth and breadth of coding skills learnt. The pass rate of the large student cohorts at UTS improved to 90% within a year, and over 95% thereafter. The reason for this improvement is that the software enables automated formative assessment of coding skills as often as weekly. Furthermore, because it reduces the university's cost (in work hours) of assessment through marking assignments, there is more teacher time for helpful interaction with students.

In 2006 this project funded by the Australian Learning and Teaching Council (then, the Carrick Institute) commenced for the purpose of redeveloping the Linuxgym software to be available to all educational institutions. The main goal was that work involved in installation, maintenance and exercise development would be minimal. Further goals were to create a broad set of exercises which could be drawn upon by academics around Australia, and a clear definition of industry standards with the long term aim of a certification exam to align Australian universities' curricula closely to industry's needs.

In collaboration with three universities, Linuxgym has been redeveloped as both an installable virtual machine, and a publicly accessible web-service upon which lecturers are able to create classes, provide students with Linuxgym accounts, allocate them a set of exercises and view their marks. With the assistance of the Linux Professional Institute, a survey has been conducted to clarify which scripting skills are needed in industry, and the results have influenced the focus of the Linuxgym library. The Linuxgym web-service has also been made available to the public as an individual learning tool (http://www.linuxgym.com).

This project has spanned three institutions and over 1000 students and individuals. Feedback through surveys and students' marks indicate that Linuxgym is a helpful and accurate tool for formative assessment.

1. Introduction

The Internet, and the computing world in general, increasingly require high level conglomeration of their systems and data to provide an efficient and knowledge-creating information flow between their units and to the outside world. Unix scripting is a key tool for making flexible use of the data streams. It is the "glue" which holds together a diverse range of applications in areas such as banking, telecommunications and genome sequencing to name a few. The Unix Command-Line Interface (CLI) enables operations which are arduous or even impossible with a point-and-click Graphical User Interface (GUI). For example, renaming a large set of files to include an extra extension could be achieved with a single line of code on the command line, as opposed to a click and edit for every file in the case of a GUI interface — as such, CLI skills are of major importance in system administration and networking.

Because learning scripting skills requires considerable practice and feedback, the typical university's high student:staff ratio offers very little scope for providing the necessary feedback, and university degrees have produced only a small percentage of graduates with the necessary scripting skills.

1.1 Project goals

Between 2002 and 2006 at UTS, Linuxgym automated the provision of formative feedback on student scripting activities. The goal of this project has been to make Linuxgym available to all universities and to ensure that it is well aligned with industry needs. In detail, it provides:

- **More practice and feedback for students**: Instead of having only one or two assignments per semester, Linuxgym should enable frequent practice and immediate feedback keeping the students informed of any topics into which they need to put more effort.
- **Lower marking workload for teachers**: With a class of 200 students, manually marking an assignment takes a whole week.
- **Improved student-teacher interaction**: The feedback enables students to be more aware if they've misunderstood the question, meaning that

when they interact with the teacher, it's on a similar wavelength. Moreover, with less time spent marking, teachers should have more time for interacting with students.

- **Alignment with industry standards:** The library should provide a growing set of topics and exercises which define the scripting skills required by the IT sector, enabling universities to direct their teaching to ensure the skills developed make the students more employable.

2. The approach

Although the alpha version of Linuxgym was clearly an improvement to previous methods of assessment, it came about from the single goal of reducing the lecturer's workload. This project involved obtaining a closer understanding of the set of problems being addressed, followed by a model of the software which should address these issues.

2.1 The difficulty of formatively assessing scripting skills

The majority of university students have great difficulty in going beyond the most basic Unix scripting skills. This is because there is insufficient formative assessment which enables the students to understand where they need to focus their efforts in learning.

To understand the reason for this we describe the two learning activities (apart from lectures) used at the University of Technology, Sydney — assignments and lab exercises. Although this refers to only two courses at one university, most university education of coding takes place using one or both of these approaches (Robins, A. et al., 2003).

Problems with assignments

The purpose of setting a practical assignment early in the semester is that when the student attempts some scripting, the tutor can read and test their code to identify errors and misunderstandings in order to provide formative feedback. An assignment later in the semester measures the extent to which the student has attained the necessary skills — a summative assessment.

Under the university's funding constraints, a tutor with thirty students is paid for four hours of marking in a semester. With two assignments in the semester, this results in four minutes of marking per assignment.

There are two reasons why this is an insufficient formative assessment. First, four minutes is not enough time to write explanatory feedback. Second, it takes a significant period before the students have been presented with sufficient material to do a whole assignment. Together with the time it takes the tutor to mark the assignments, this formative feedback arrives late in the semester — often too late for the students to understand and build upon this feedback.

Furthermore, an accurate assessment of coding — and scripting in particular — is difficult because there are many ways of performing the same task. It is sometimes the case that the tutor misunderstands the student's code, assuming errors where there is simply a different but valid approach.

Problems with lab exercises (combined with multiple choice tests)

In lead-up to tests, students are given a weekly set of exercises and they may seek advice from a tutor in an hour-long lab session. Throughout the semester there are tests to provide feedback on the extent to which the students have learnt what was intended.

In the typical class of thirty students, this leaves two minutes each week (on average) for any student to have the attention of the tutor — a total of half an hour throughout the semester. With this low level of interaction, it is often the case that both the tutor and the student are unaware of the knowledge gap. The tutor does not have time to locate the errors, and the student is unable to identify them.

To increase the amount of formative feedback, the method of testing is multiple-choice rather than the time consuming assessment of practical assignments. Unfortunately, a multiple-choice test can be a very inaccurate assessment of the student's actual skills and may instead be a representation of their ability to memorise facts through surface learning (McCracken M., et al., 2001). Therefore, a student might proceed through such a course believing they have the necessary skills when they do not.

2.2 The solution

Linuxgym is a tool for presenting problems to be solved by coding. Solutions are automatically assessed — providing feedback for the students to understand their errors, to the teachers on what topics need better coverage, and enabling an accurate assessment of a person's professional coding skill.

The underlying principle of Linuxgym is that tasks are neither multiple choice questions nor simulations, but coding as performed in practice — which, according to Newble and Clarke (Ramsden, 1992), is more likely to bring about a deeper learning. Linuxgym provokes the students to engage in authentic activities — in some sense a constructivist approach (Nunes, 2006).

The method of automatically generating feedback is by defining a list of attributes (derived from the overall task) which indicate the actual skills needed, and then showing the students which of the attributes they have attained.

For the remainder of this section, we will give an example of a Linuxgym exercise and explain its relationship with the skills being assessed.

Example: An exercise in data analysis

The students are setup with a Linux account on a server which has directories of data to which they can apply their scripts, such as the data in the file called femalenames.txt

/usr/local/linuxgym-data/census/femalenames.txt
Frequently Occurring First Names, US Census Bureau, 1990

MARY	2.629	1
PATRICIA	1.073	2
LINDA	1.035	3
…	…	…

Using this data as an example, the students are told to write a script whose behaviour is described as follows:

> **Search Prefix** Create a bash script called searchprefix.sh which takes two arguments, a string and a file (in that order). The output should be each line of the file which matches the string from the beginning of the line. For example, given a string "ANA" the line starting with "ANABEL" will be printed,

but the line starting with "SUSANA" will not. (Hint: use grep and ^ to indicate the beginning of the line).

Apart from the instructions there are “Links” to web-sites with relevant learning materials, as well as “Rules” — points which assist the student in avoiding silly errors, such as storing their script in the wrong directory.

To understand the function of Linuxgym, suppose the student’s solution is the following script:

```
student@linuxgym:~/ch6-data$ cat searchprefix.sh
#!/bin/bash

grep $1 $2
```

Then when pressing the “Mark Question” button, the following group of attributes is displayed:

✔ File exists

✔ File is executable

 The output of this script contains all necessary lines in some order

 The output contains some unnecessary lines

 Script behaviour is incorrect

From this the student can see that they are accidentally matching some lines which are not those intended, but otherwise, the script is in the right location and gets all of the necessary lines. The way this assessment takes place is by Linuxgym executing the student's searchprefix.sh on a hidden string and data file, and comparing its behaviour with that of a correctly implemented script.

3. Outcomes

3.1 Advancement in knowledge of industry requirements

Although Linuxgym exercises involve actual scripting, there are essential differences to the professional activities in this area, and it was our intention to identify these differences in order to be able to evaluate the outcomes of using Linuxgym as a formal assessment tool.

Firstly, the exercises presented are a small project. There is no technical barrier to presenting the students with an entire assignment, but the teaching methodology is aligned with providing more sounding-boards from which feedback can be provided. Another important difference is that, while teachers (and colleagues) are very concerned with qualitative aspects of the code, such as its readability and maintainability, Linuxgym generally pays little attention to these aspects unless they are objectively measurable — such as ensuring that the correct language is used for the implementation. Finally, Linuxgym provides a learning environment which displays links to relevant web pages, and "rules" which keep at the forefront the assumptions it makes about the student's code. Overall, Linuxgym closes the gap between information that is needed and locating it, as well as the gap between synthesizing and evaluating it (Webber and Johnson, 2000).

A similarity between Linuxgym and the real-world is that the machinery emulates a phase of the student's code being used, providing the feedback which would result from a line-manager noticing a bug and is therefore more accurate than a tutor simply reading the code or using it only once. *Unlike* the real-world, the student quickly gets a clear explanation of what is wrong with their code's behaviour.

Survey of the scripting skills in demand

Having understood the gaps between the Linuxgym technique of assessment and experience in the real world, we also needed to determine its coverage – the contents of the library.

Job advertisements indicate that the scripting skills mostly in demand are as follows:

- Data mining for the financial market trading institutions — scanning news services and modelling using rapid prototyping;
- Genome sequencing for the human genome project — sharing data from differently designed databases in different laboratories;
- Translation of protocols for mobile telecommunications;
- System monitoring in network security; and
- Data merging and manipulation for web services.

Due to the diversity of these applications, the top level conceptual constructs are not the same across the Unix scripting profession. It is therefore the technical foundation which defines these skills and on which the formal assessment must focus.

A survey of industry requirements for these technical skills was approached by producing a list of ninety commands and activities (“tools”) split into the following sections defining places where they are used:

- Command Line Interface;
- Managing files and directories;
- Information management;
- Bash scripting; and
- Data transfer.

Each tool was categorized by the participant as one of the following types:

- **Pocket**: The basic, “bread and butter” stuff. Tools used all the time which everyone must know in order to survive.
- **Toolkit**: Less fundamental but still very useful. Things you should be able to use without consulting the manual.
- **Van**: Tools in the van are ones you don't often use, but you should be aware that they exist, understand what they're for, and able to work out how to use them when the need arises.
- **Shed**: Other less important tools for which you would rarely find a use.

In addition, we had the participants indicate their application areas to see whether there would be any notable differences on the emphasis of different skills. These application areas were aligned with those found in the job advertisements, namely: system administration; networking; software development; web services; data analysis; teaching; or other. The survey was posted to discussion forums and mailing lists across the application areas.

There were 183 participants, most of them crossing several of the application areas listed above. Although 83.6% select system administration, only 9% work solely on this application. This application is followed by 64.5% performing software development and 48.6% doing data analysis. The largest number of participants (44.5%) has between 6 and 15 years experience as a scripting expert, followed by 31.3% who have more than 15 years experience.

Agreements: There are many shell scripting languages available in Unix, and there was some concern that the minor differences between these languages would be an obstacle to agreed upon standards to scripting skills, however 84.7% indicate they are using Bourne-Again Shell (Bash) followed by 34.4% using Bourne Shell (sh), and only 23.5% using Korn shell (which adheres to the POSIX 2 standard). Therefore we regard Bash as the industry standard in practice.

Disagreements: There were few topics on which there was great variance in the way they were categorized from “pocket” to “shed”.

- Simple functions for minimizing the number of keystrokes at the command-line;
- Whatis — a function which provides a summary of the man page;
- Emacs — a text editor which is more user-friendly than Vim.

These topics are very much based upon how the user performs various tasks, rather than the outcome, and therefore these skills should not be assessed as part of an accreditation. Further research is necessary to understand the contexts which bring about the differing responses on the following commands:

- Let and Printf — to perform and display the outcome of arithmetic operations.
- Trap — a method of preventing users from aborting scripts to ensure data integrity.

Summary: The skills which a scripting expert should have without having to read the manual pages are:

- Methods for transferring data between computers and working over a network (ssh, scp, sftp);
- Regular expressions for matching sets of strings;
- Vim — the main tool for text editing;
- Functions (such as grep and find) which search for data through file systems and data streams;
- Piping and redirection — managing interaction of processes; and
- Structured programming using functions such as if, while, for and case.

Data manipulation and transformation in preparation for use with a database or web system is a core activity of almost 50% of the respondents. Although functions such as awk, col, comm, join and paste are necessary, there was general agreement that the scripting expert needs access to the manual to use them each time. As such, Linuxgym should not demand the use of these functions without access to the manual.

3.2 Evaluation of Linuxgym

The formal evaluation of Linuxgym has been carried out by the Linux Professional Institute (LPI) at its Europe, Middle-East and Africa summit held in Utrecht, The Netherlands in November 2007. LPI is a not-for-profit organization established in 1999 to promote and certify essential skills on Linux and Open Source technologies through the global delivery of vendor-independent exams. As defined at the outset of this project, the criteria which have been verified by LPI are Linuxgym's accuracy as a testing tool, and the alignment of its content with industry standards. Aside from the formal evaluation, the LPI committee and its exam delivery partners are also considering Linuxgym as a tool to enable the expansion of their curriculum beyond system administration, and into the area of coding.

Strengths

The main strength of Linuxgym is its combination of being both a formative and summative assessment tool. At UTS Linuxgym has successfully been used to increase both coverage and pass rate of large student cohorts. Over three semesters, the undergraduate coverage — initially just copying and editing files — has expanded to include writing complex scripts without a noticeable change in the pass rate. In a postgraduate course where the failure rate of the four preceding semesters was 30–50%, the first semester using Linuxgym had a failure rate of almost zero with an average mark of over

75% for assignments — with very little change to the curriculum. In both cases the cost/work overload of marking has been reduced.
Linuxgym has also been used for formative assessment at the University of Sydney, and at the University of Portsmouth, UK where it has been adopted in a course on System Administration — more specialised than the courses at participating universities in Sydney:

> *Having used both the Redhat and LPI educational materials I have adopted Linuxgym for my class, as it fills serious gaps both for education, and assessment of these skills. Because learning scripting skills requires a great deal of practice Linuxgym provides much more helpful feedback than any staff member can offer. Moreover, the exercises are a far more accurate representation of real world activities than any of the assessment tools offered by those currently offering a certification*
> .
> Frank Margrave, Principal Lecturer, Department of Electronic and Computer Engineering, University of Portsmouth.

Below is a sample of the anonymous comments of students using Linuxgym in 2007, 2008. See Appendix A for the statistical summary of the student survey.

> *I found Linuxgym a very easy and motivating way to learn Linux.*
>
> *Really easy to use, and a very helpful learning aid.*
>
> *It was the best educational method I have ever seen in UTS.*
>
> *A creative way to learn by actually applying it as we would in the workforce.*
>
> *As one who never see Linux, before I think Linuxgym is a great learning tool, specially for basic!!!*
>
> *Linuxgym is a great pedagogical tool and complements any sound, computer-oriented subject. It has certainly enhanced my experience on Linux and has provided me with sufficient practical skills to perform elementary and advanced tasks on the Linux CLI. Thank you.*
>
> *Its the best way i've actually learnt how to use Linux compared to that of TAFE. Linuxgym rules hands down.*

Very enjoyable. I'm now considering recommissioning one of my older computers at home as a Linux machine to keep in practice. Some of the later exercises were very challenging, but I guess there has to be a discriminator between "good" and "great".

Great system, really encouraged me to learn several things I've been wanting to learn for a long time.

Weaknesses

The uptake in institutions outside of this project was certainly less than expected, and the reference group meeting in February 2008 came to a clear understanding of the problems which came about.

In technical terms the problems which came up were a consequence of making the software available without excessive work for the lecturer. Although running Linuxgym does not require a Linux based desktop computer, the size of the software and complexity of installing it was an impediment to students using it at home. The network requirements of the software require cooperation of the university's system administrators to modify various network security settings. Although the solution to these issues is not obvious, it is clear that the system administrators of the participating institutions should have been major participants on the project.

In terms of coverage, although the results of the industry survey were very clear, this was not influential enough to convince any of the partner universities to align their curricula with the Linuxgym library — in spite of this being a goal of the project. This reluctance is partly due to the speed at which the necessary hands-on skill set changes within the IT industry, but equally the need of universities to maintain links between its coursework curriculum and research. While there is no simple solution to this problem, a well established and stable Linuxgym certification supported by industry would make university curriculum alignment more worthwhile, which will happen if the Linuxgym library withstands the test of time.

4. Summary and future goals

Although the university uptake of Linuxgym was less than expected, the significant improvement in the students' learning experience, an increase in

the material covered, and a decrease in the teacher's workload indicates that a few changes to Linuxgym will rapidly increase its uptake.

In technical terms, the machinery is being redesigned to avoid the necessity of any software being installed on the student's computers without impacting on the server's performance.

It has become clear that the curriculum is the harder problem to solve and requires two independent approaches. The first approach is that Linuxgym's content — which has been validated and improved by a survey of industry needs — must be made available for a certification exam. This will be made possible by the technical improvements above, and will be accessible across the IT professional and education sector across the globe. This should suffice in convincing universities to align their curricula with industry's needs.

Another technical improvement — of disconnecting the exercise environment from the management of questions and feedback — will enable Linuxgym to address a broader set of learning goals.

REFERENCES

Book

Ramsden, P., 2003. Learning to Teach in Higher Education. Routledge London, UK.

Journals publications

Brennedsen, Jens and Michael E. Caspersen, 2007, Failure rates in introductory programming, ACM SIGCSE Bulletin, Volume 39 , Issue 2, pp.32—36.

Robins, A. et al., 2003. Learning and Teaching Programming: A Review and Discussion. Computer Science Education, Vol. 13, No. 2, pp 137–172.

Nunes, M and M. McPherson, 2006. Learning support in online constructivist environments in information systems. HEA-ICS Italics, Electronic Journal, Vol. 5, No. 2, pp 1–11.

McCracken, M. et al., 2001. A multi-national, multi-institutional study of assessment of programming skills of first-year CS students. ACM SIGCSE Bulletin, Vol. 22, No, 4, pp 125–180.

Webber, S and B. Johnston, 2000. Conceptions of Information Literacy: New Perspectives and Implications. Journal of Information Science, Vol. 26, No. 6, pp 381–397.

Conference paper

Solomon, A., D. Santamaria and R. Lister, Automated Testing of Unix Command-line and Scripting Skills (2006), 7th International Conference on Information Technology Based Higher Education and Training.

Appendix A: Student Survey

Assessment Statistics LinuxGym Student Questionnaire

Name: LinuxGym Student Questionnaire

Number of Attempts: 92

Instructions: This survey is part of a research project which is developing and evaluating the effectiveness of LinuxGym. The project is called LinuxGym: A sustainable and easy-to-use automated developmental assessment tool for computer scripting skills (UTS HREC approval reference number UTS HREC 2007-52). The project is being undertaken at UTS by Dr Andrew Solomon and Dr. Raymond Lister, Faculty of Information Technology (Tel: 9514 1850 email: raymond@it.uts.edu.au)

This survey is anonymous. For questions 1 to 9, Andrew Solomon (the subject coordinator) and any other teaching staff will only ever be shown statistical summaries of everyone's responses. They will never know the details of your individual response.

Please respond to the following statements about LinuxGym with one of the following responses: Strongly Agree, Agree, Neutral, Disagree, Strongly Disagree, Not Applicable.

Question 1 Opinion Scale/Likert

LinuxGym is more motivating to practice Linux than a written test.

Answers	Percent Answered
Strongly Agree	54.348%
Agree	40.217%
Neither Agree nor Disagree	3.261%
Disagree	1.087%
Strongly Disagree	1.087%
Not Applicable	0%
Unanswered	0%

Question 2 Opinion Scale/Likert

LinuxGym is more motivating to practice Linux than a written assignment.

Answers	Percent Answered
Strongly Agree	56.522%
Agree	36.957%
Neither Agree nor Disagree	3.261%
Disagree	1.087%
Strongly Disagree	2.174%
Not Applicable	0%
Unanswered	0%

Question 3 Opinion Scale/Likert

LinuxGym helped me to improve my Linux skills.

Answers	Percent Answered
Strongly Agree	53.261%

Agree	38.043%
Neither Agree nor Disagree	3.261%
Disagree	2.174%
Strongly Disagree	1.087%
Not Applicable	0%
Unanswered	2.174%

Question 4 Opinion Scale/Likert

I prefer LinuxGym to a written test.

Answers	Percent Answered
Strongly Agree	59.783%
Agree	27.174%
Neither Agree nor Disagree	2.174%
Disagree	4.348%
Strongly Disagree	6.522%
Not Applicable	0%
Unanswered	0%

Question 5 Opinion Scale/Likert

I prefer LinuxGym to a written assignment.

Answers	Percent Answered
Strongly Agree	59.783%
Agree	25%
Neither Agree nor Disagree	4.348%
Disagree	5.435%
Strongly Disagree	5.435%
Not Applicable	0%
Unanswered	0%

Question 6 Opinion Scale/Likert

The feedback I received from LinuxGym when I made errors helped me to subsequently improve my Unix skills.

Answers	Percent Answered
Strongly Agree	20.652%
Agree	42.391%
Neither Agree nor Disagree	18.478%
Disagree	11.957%
Strongly Disagree	6.522%
Not Applicable	0%
Unanswered	0%

Question 7 Opinion Scale/Likert

LinuxGym has given me an accurate idea of my Linux skills

Answers	Percent Answered
Strongly Agree	18.478%
Agree	58.696%
Neither Agree nor Disagree	16.304%
Disagree	4.348%
Strongly Disagree	1.087%
Not Applicable	0%
Unanswered	1.087%

Question 8 Opinion Scale/Likert

LinuxGym marking was consistent and fair.

Answers	Percent Answered
Strongly Agree	28.261%
Agree	52.174%
Neither Agree nor Disagree	11.957%
Disagree	5.435%
Strongly Disagree	1.087%
Not Applicable	1.087%
Unanswered	0%

Question 9 Opinion Scale/Likert

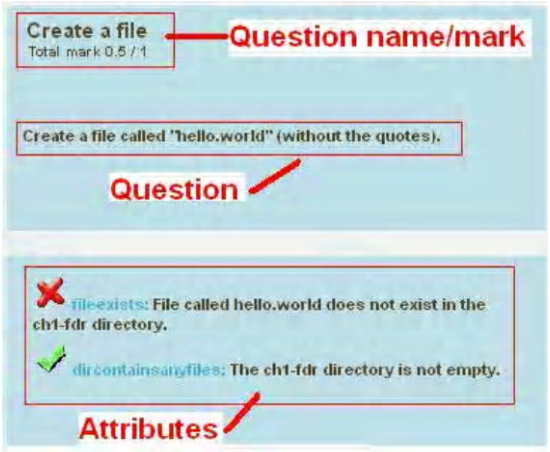
My learning experiences with LinuxGym were interesting and thought provoking.

Answers	Percent Answered
Strongly Agree	30.435%
Agree	53.261%
Neither Agree nor Disagree	11.957%
Disagree	4.348%
Strongly Disagree	0%
Not Applicable	0%
Unanswered	0%

Question 10 Multiple Answer

To understand what we mean by "Questions" and "Attributes" see the picture below.

Please select the features of the diagram which you find useful **Linuxgym Online**(http://linux
the website where you can view your results.



Answers	Percent Answered
Question name and mark	71.739%
Question	73.913%
Attributes	85.87%

Question 11 True/False

When you do not get a question completely correct, you get **partial marks** for the attributes which were satisfied (as shown in the diagram above). We would like to know your opinion of whether:

- partial marks are an accurate measure of skills, or;

- a binary choice of 0 or 1 would be a more accurate measure of skills.

Please answer True or False to the following statement:

"Partial marks are a more accurate measure of my skills than simply 0 or 1".

Answers	Percent Answered
True	90.217%
False	9.783%
Unanswered	0%

Question 12 Essay

If there's anything else you'd like to say about LinuxGym, please enter it in the box below.

Any comments you make are anonymous. Your name will not be associated with your response.

Unanswered Responses

48

Given Answers

The constant assessment each week had me more focussed on getting full marks for each question rather than learning.