Programming your success

Needs Assessment

VANIER
CÉGEP/COLLEGE

Haritos Kavallos

Produced on October 25, 2012

Revised on November 20, 2012

Disclaimer

Unless stated otherwise, all materials, textual, visual, or other in this document are the intellectual property of Vanier College and may not be copied, reproduced, distributed or displayed without Vanier College's express written permission.

Revisions

Version	Editor	Date
1.0a	Haritos Kavallos	Oct 25, 2012
2.0a	Haritos Kavallos	Nov 20, 2012

Table of Contents

Introduction

Background Information

The Mathematics & Science Centre (henceforth: MSC) at Vanier College was established with the primary mission of offering a peer tutoring service that has traditionally aimed to benefit students studying in pre-university science programs. The service is drop-in based where students can sit with tutors any time during operating hours and ask for explanations on concepts or seek assistance with understanding assignments. The service is also offered privately where a student can request for a tutor to meet with them individually and get help on a continual basis throughout the semester.

The MSC has been looking to expand its tutoring service to accommodate three-year career programs – specifically, the Computer Science and Technology program which will be run as a pilot test.

As of August 2012, the Centre is coordinated by Harry Kavallos.

Overview of the Project

The purpose of this project is to assess the need for a training program (to be developed by the MSC) for students studying in the Computer Science and Technology program who are struggling and require out of the classroom help with their first Java Programming course. To that extent, this project aims to analyze, design, develop, implement, and evaluate a tutorial that not only covers the fundamentals of Java Programming but also introduces novice programmers to concepts that help them think more programmatically. As per the sponsor's request, an e-learning solution is expected.

This report will outline the problem by stating the sponsor's request and business need, examine a current fictional performance scenario as well as an ideal performance scenario, and go through a breakdown of associated tasks and project constraints. Furthermore, the learner will be described through several facets before concluding with a list of content objectives as well as a list of tools that will assist in evaluating the effectiveness of the completed training project.

Data Collection

The data collected for this report was gathered via an interview conducted with Safiya Simon, former MSC Coordinator from 2010-2012 and Louise Gauthier, Professor and Coordinator of the Computer Science and Technology program. An additional interview will be scheduled with a professional from Vanier College's Professional Development Office which, among other things, has a mandate to explore and help advise on implementing new training programs for both students and faculty members.

The Problem

Sponsor's Request

The project was initiated by Safiya Simon – former MSC Coordinator – who sought to expand the MSC's peer tutor offerings beyond the pre-university science programs. Specifically, she expressed the need to produce an online training program that could assist students studying in Computer Science and Technology with understanding the fundamentals of basic Java programming.

Ms. Simon identified several issues that helped prompt her to see a need for this initiative:

- The MSC does not employ enough peer tutors with the knowledge required to tutor programming. [The MSC budget cannot afford to hire several peer tutors that focus specifically on helping students with programming].

- As career programs are growing within the College, there is an increase in students requesting help with specialized topics such as: programming, computer science mathematics, etc.

- Most students that pass through the MSC looking for help in career program courses are turned away and often do not have any other means of obtaining extra help from any of the services within the College.

Business Need

The business need that this project will aim to achieve is increasing revenue. Career programs at Vanier College are known for seeing large attrition rates in comparison to the pre-university programs. Specifically in the Computer Science and Technology program, it is common for half the students transitioning from their first semester to their second semester to switch programs or to leave the College altogether. Like all Cégeps, Vanier's governmental endowment is directly related to student enrollment numbers. This means that the attrition in career programs affects Vanier negatively since fewer students' means a smaller annual budget.

Furthermore, this project is in-line with Vanier's 2008-2013 Strategic Plan on two main points. First, it states that Vanier will ensure that academic support services will *have the needed resources* to handle the changing needs of students (Vanier, 2008). Second, the plan states that Vanier is obligated to ensure that all incoming students *have support services to turn to* and that such support services will be modified to help students excel in their programs of study (Vanier, 2008).

Current Performance Situation

Steven is a first semester student enrolled in Vanier College's three year Computer Science and Technology program. He is fresh out of High School and decided he would pursue an education and eventually a career in the computer science field. He is motivated by the fact that he is an avid gamer and figures it would be interesting to be involved in making games in the future.

Unfortunately, three weeks in and Steven is struggling to get through his first Java programming assignment and as the deadline approaches, he is becoming increasingly frustrated. His professor informs him that the MSC is a great place to seek peer tutoring. Taking his professors advice, Steven makes his way to the Centre.

At the Centre, Steven approaches the Coordinator and explains that he needs help with Java programming. The Coordinator mentions how there is only one tutor who has the know-how to tutor programming, however, that tutor's area of expertise is mostly C++ programming but might be able to offer assistance regardless.

Eager to sign up to be paired with this student, Steven responds, "Alright! Can you sign me up with him? I'd like to start as soon as possible." Unfortunately for Steven once more, the tutor only does drop-in tutoring twice a week and to make matters worse, the times that the tutor is available, Steven is in the smack middle of his Java programming class. Discouraged and frustrated, Steven sighs and leaves the Centre feeling like he has been let down.

On his own, Steven does his best to complete the first assignment and ends up coming out with a poor grade. Having struggled to understand the fundamentals of thinking like a programmer, his struggles continue to the point where Steven strongly considers the idea of switching into a different program.

Ideal Performance Situation

Steven is a first semester student enrolled in Vanier College's three year Computer Science and

Technology program. He is fresh out of High School and decided he would pursue an education and eventually a career in the computer science field. He is motivated by the fact that he is an avid gamer and figures it would be interesting to be involved in making games in the future.

Two weeks in and Steven is struggling to get through his first Java programming assignment. He decides it is time he seeks some help well before the assignment deadline so that he can fall back on track along with the rest of his friends in the program. His professor informs him that the MSC is a great place to seek peer tutoring and has recently become more involved in assisting students from career programs. Taking his professors advice, Steven makes his way to the Centre.

At the Centre, Steven approaches the Coordinator and explains that he needs help with Java programming. He sits with Steven and explains how there is one drop-in tutor available twice a week that can always be consulted for assistance with theoretical programming concepts and practical assignments. The Coordinator also hands Steven a pamphlet titled: “Programming Your Success”.

The pamphlet explains how thinking programmatically is a difficult process for most people who have not worked with programming languages before. The Coordinator points to a hyperlink in the pamphlet and explains to Steven how a new online tutorial was created that is specifically geared towards tutoring and evaluating novice programmers in basic and intermediate concepts that are fundamental in helping students see a higher success rate in their first semester. Feeling relieved that he was directed to a potentially powerful resource, Stevens only question is, “is it free?”, to which the Coordinator chuckles at followed by a nod.

Using the tutorial as a way of complimenting his class notes and textbook readings, Steven is not only able to successfully complete his first assignment, but he is now thinking about programming in a new light and tackling problems in more effective ways.

Tasks in Ideal Performance

End Result: Computer Science and Technology students *understand the fundamentals of Java programming and successfully transition to their second semester*.

Main and Supporting Tasks

1. Describe how to think like a programmer.
    - Describe different methods for organizing thoughts before tackling a problem.
    - Describe how to split a problem into smaller problems so that they can be solved individually and more efficiently.

2. Demonstrate how to set up the Java programming environment as used by students in the Computer Science and Technology program.
    - Setup the environment by completing the following:
        - Download Java 2 Software Development Kit.
        - Download the free Textpad application.
        - Using Textpad, open a sample program provided by the teacher or the textbook.
        - Run the compiler built into Textpad to ensure that it is operational.
3. Explain basic programming topics.
    - Define what variables are and how their scopes behave.
    - Describe the different primitive data types that exist in Java.
    - Define what different types of conditional statements exist in Java.
    - Explain the advantages of using a SWITCH over an IF, ELSE statement.
    - Explain the difference between a FOR and WHILE loop.
    - Explain the logic behind how arrays function.
    - Explain why you would want to consider using methods in your program.
4. Explain Object Oriented Programming (OOP).
    - Describe the advantages of OOP over sequential programming.
    - Define what a constructor is.
    - Describe the purpose of GETTER and SETTER methods.
    - Explain the process of how to instantiate an object of a class in Java.
5. Create a Java program that covers the main topics taught.
    - Write code to produce a simple class structure that includes a main method.
    - Write code to define variables with global and local scope.
    - Write code that displays output text onto the computer screen.
    - Write code that captures keyboard data entered by a user.
    - Write code to produce the following constructs:
        - Create a basic conditional statement using IF and ELSE.
            - Select the Boolean variable or expression that will be tested as being true or false.
            - Write the generic syntax to produce the conditional structure.
            - Between the IF brackets, write the statement (or sets of) that will execute when the condition is tested as TRUE.
            - Between the ELSE brackets, write the statement (or sets of) that will execute when the condition is tested as FALSE.
        - Create a complex conditional statement using SWITCH.
            - Select the variable that will be tested to match against one of the possible cases.

        - Write the generic syntax to produce the conditional case structure with an inclusive default catch-all case.
        - Within the case structure, write the match condition for each case.
        - Between each case, write the statement (or sets of) that will execute when the variable matches the case's condition.
    - Create a WHILE loop.
        - Select the variable that will act as a counter for the loop.
        - Write the generic syntax to produce a WHILE loop structure.
        - Within the loop brackets, write the statement (or sets of) that will be executed during each iteration of the loop.
        - Within the loop brackets, write the statement that will change the counter value.
    - Create a FOR loop.
        - Write the generic syntax to produce a FOR loop structure.
        - Within the loop structure, write the counter variable's initial value, the loop stop condition, and the counter variable changer.
        - Within the loop brackets, write the statement (or sets of) that will be executed during each of the iterations of the loop.
- Write code that defines methods with and without formal parameters.
- Make sure the program compiles without any errors.

Note: In order to limit the scope to what is feasible in 30 minutes of instruction as required for building the learning materials, only task points 2 and 3 will be expanded on in the report(s) that follow this assessment.

Foreseeable Constraints

- **Budget:** The MSC has a yearly operational budget of $13,000 and an additional $20,000 specifically reserved for the peer tutoring service. Historically, every year the Centre ends up over budget, thus, there is no opportunity to have the MSC fund any new training programs. [Funding opportunities might exist through the Dean of Technology/Career Programs however – this remains to be explored].

- **Staff:** Since the expectation is that the MSC will produce the entire training program, it will become necessary to extend the current assistant's hours from part-time (14 hours a week) to full-time (35 hours a week). This will make it possible for the assistant to administer the MSC while the Coordinator focuses most of his efforts on producing the training program within a reasonable timeframe [which remains to be discussed]. The constraint however is that budget restrictions might prevent the assistant from being given the additional hours.

- **Viewing Guidelines:** The online training will need to be viewable on all major browsers (Chrome, Explorer, Firefox, Safari, and Opera) and compatible with at least the latest versions of each. Adobe Flash is a requirement as well as sound; though, closed captioning will be made available. A stable internet connection is required.

Learning Environment

The request is to develop this training program online so that it may be widely accessible by learners. As such, the surroundings of the learning environment can vary greatly; though, the learning environment itself, a computer or tablet, will be common for all learners.

Due to the training's viewing guidelines, it is possible that some learners might have issues accessing the training through their own devices. If that is the case, the MSC offers 21 computer stations that any student can use for academic purposes. These stations would also be capable of displaying the training. The limitation however is that these computers are often occupied quickly and no time restrictions are in place for students who are using them. Normally, another limitation would be that learners would have a hard time finding the MSC since it is located in an area that receives little traffic; however, learners looking for the help will be directed to the center by their teachers or the program coordinator.

A link to the online training will be added to the Computer Science and Technology program website, the MSC website under the resources section, and promoted by the center's coordinator. The training will be designed to be accommodative to students with certain physical disabilities: closed captioning for learners with limited hearing, and zooming capabilities for learners with vision problems will be incorporated into the development.

The Learner

Learner Profile

Students entering the Computer Science and Technology program are in most cases transitioning directly from high school – though, this is not always the case (see Outlier Learner Type II below for instance). These students normally struggle academically because they are not used to the changes between high school and Cégep environments. In high school, these students would have been used to being monitored and offered feedback more often – feedback that would be less frequently given in Cégep. Furthermore, some students have a difficult time adjusting to the autonomy granted in Cégep coupled with the fact that the workload of a career program is much higher than what students would have been used to in high school.

The following learner characteristics are worth noting:

- The majority of students are between 17 and 19 years old. A second group of students, typically in the process of completing their second DEC, is between 20 and 25 years old. The smallest group of students, often in the process of re-training themselves or switching careers, is over 35 years old.

- As part of its Strategic Plan, Vanier is aiming to grow its international profile (Vanier, 2008). As such, approximately 3% of students in the program are from other countries.

- Although this demographic is shifting, currently the program is split at 85% male and 15% female (2012).

The following describes three examples of possible learner types, where the typical learner is representative of the majority of students within the program.

Outlier Learner (Type I): Jenna is 17 years old, fresh out of high school and has just started Cégep. In high school, she was very involved with an annual robotics competition and with the help of mentors got a lot of experience in C programming. She also took a computer class where she learnt web programming. Having found a passion for computers at a young age, Jenna has been autonomously learning as much programming as she can fit into her spare time.

Outlier Learner (Type II): Carl is 51 years old and already holds a DEC from a trade school he attended during his teenage years. He had been working as a developer for Mechtronix for a little over twenty years up until he was laid off a few months ago. Since Carl never self-trained himself, he’s come to realize that his skill-set is obsolete and no longer marketable. Carl hopes to speed track through the program and get back to work as soon as possible.

Typical Learner: Cynthia is 18 years old, fresh out of high school and has just started Cégep. She uses a computer daily but aside from knowing how to troubleshoot problems and use common applications, Cynthia has never programmed before though, she is fascinated with the idea of building software.

Previous Knowledge

Students entering the Computer Science and Technology program at times bring with them basic knowledge of web programming, particularly HTML and CSS knowledge which they would have typically acquired through a high school computer class. A small percentage of students have self-taught themselves some form of application programming language, typically C++ or Java. A large majority of students enroll in the program with no prior experience in programming – web, application, or otherwise.

Challenges Faced

Students in the Computer Science and Technology program who are struggling with their courses are facing a disadvantage. Whereas students in pre-university programs have access to academic support resources offered through: Student Services, The Learning Centre, The Mathematics & Science Centre, and program specific resource centers; computer science students are only able to seek support through their professors as no additional support services are currently in place for them. The prior is also true for all other career programs at Vanier College.

Requirements

Content Objectives

1. Given that the online training unit on problem solving is completed, describe the problem solving tactics behind how to think like a programmer.
    - Describe at least two techniques/tools that are useful for organizing thoughts before tackling a problem.
    - After having explored a few cases provided through the online training, describe how to split a problem into smaller problems so that they can be solved individually and more efficiently.
2. With the aid of any online resource, set-up the Java programming environment as used by students in the Computer Science and Technology program within 20 minutes.
    - Given any computer or laptop, setup the programming environment by completing the following:
        - Download Java 2 Software Development Kit.
        - Download the free Textpad application.
        - Using Textpad, open a sample program provided by the teacher or the textbook.

        - o Run the compiler built into Textpad to ensure that it is operational.

3. Given that the online training unit(s) on Java fundamentals is completed, explain basic programming topics with 70% - 90% accuracy.
    - Define what variables are and how their scopes behave.
    - Describe all the different primitive data types that exist in Java.
    - Given an understanding of choice constructs, define what different types of conditional statements exist in Java.
    - Explain the advantages of using a SWITCH over an IF, ELSE statement.
    - Given an understanding of loop constructs in Java, explain the difference between a FOR and WHILE loop.
    - Explain the logic behind how arrays function.
    - Explain why you would want to consider using methods in your program.
4. Given that the online training unit on classes is completed, explain Object Oriented Programming (OOP) with 70% - 90% accuracy.
    - Given an understanding of both OOP and sequential programs, describe the advantages of OOP over sequential programming.
    - Given prior knowledge of how to create methods in Java, define what a constructor is and the different forms it can take.
    - Given prior knowledge of how to create methods in Java, describe the purpose of GETTER and SETTER methods.
    - Explain the process of how to instantiate an object of a class in Java by making a call to an existing constructor.
5. Upon completing the online training, create a Java program that covers the main topics taught that includes at least 90% of the following and is error free:
    - Given a basic understanding of OOP, write code to produce a simple class structure that includes a main method.
    - Write code to define variables with global and local scope.
    - Write code that displays output text onto the computer screen.
    - Write code that captures keyboard data entered by a user.
    - Given an understanding of the basic fundamentals of Java programming, write code to produce the following constructs without any syntactic errors:
        - o Create a basic conditional statement using IF and ELSE.
            - When a condition needs to be evaluated as true or false, select the Boolean variable or expression that will be tested.
            - Write the generic syntax to produce the conditional structure.
            - Between the IF brackets, write the statement (or sets of) that will execute when the condition is tested as TRUE.

        - Between the ELSE brackets, write the statement (or sets of) that will execute when the condition is tested as FALSE.
    - Create a complex conditional statement using SWITCH.
        - When a condition needs to be evaluated against multiple values, select the variable that will be tested to match against one of the possible cases.
        - Write the generic syntax to produce the conditional case structure with an inclusive default catch-all case.
        - Within the case structure, write the match condition for each case.
        - Between each case, write the statement (or sets of) that will execute when the variable matches the case's condition.
    - Create a WHILE loop.
        - Select the variable that will act as a counter for the loop.
        - Write the generic syntax to produce a WHILE loop structure.
        - Within the loop brackets, write the statement (or sets of) that will be executed during each iteration of the loop.
        - Within the loop brackets, write the statement that will change the counter value.
    - Create a FOR loop.
        - Write the generic syntax to produce a FOR loop structure.
        - Within the loop structure, write the counter variable's initial value, the loop stop condition, and the counter variable changer.
        - Within the loop brackets, write the statement (or sets of) that will be executed during each of the iterations of the loop.
- Given prior knowledge of how to create methods in Java, write code that defines methods with and without formal parameters.
- Given a correctly setup programming environment, ensure that the program compiles without any errors.

Evaluation

Level 1

This level of evaluation has been adapted from *Training design basics*, Carliner (2003).

Tells us what you thought:

1. In one word, how would you describe the e-training? ____________________________

2. Using a number, how would you describe this e-training?

1 Horrible	2	3 Average	4	5 Excellent

3. How much did you know about Java Programming before taking this e-training?

1 Nothing	2	3 Some	4	5 A lot

4. How much did you know about Java Programming after taking this e-training?

1 Nothing	2	3 Some	4	5 A lot

5. How likely are you to use some or all of the skills taught to you through this e-training?

1 Not at all	2	3 Uncertain	4	5 Highly likely

6. What did you like most about the e-training?

   ____

   ____

7. What do you think could be improved with the e-training?

   ____

   ____

8. How do you feel about e-training having been used as the mode of delivery?

   ____

Level 2

The summative evaluation that follows will look to assess the level at which the learners succeeded with their objectives. The main objectives specific to this training are outlined below.

1. Given that the online training unit on problem solving is completed, describe the problem solving tactics behind how to think like a programmer.
2. With the aid of any online resource, set-up the Java programming environment as used by students in the Computer Science and Technology program within 20 minutes.
3. Given that the online training unit(s) on Java fundamentals is completed, explain basic programming topics with 70% - 90% accuracy.
4. Given that the online training unit on classes is completed, explain Object Oriented Programming (OOP) with 70% - 90% accuracy.
5. Upon completing the online training, create a Java program that covers the main topics taught.

This level of evaluation will focus on testing whether the learner has acquired the expected required knowledge.

Format of Evaluation

One possible option is that this level of evaluation will be integrated within the online training as a core component that learners will be encouraged to complete. If a question is answered incorrectly, the program would not only provide the correct answer but explain step by step how such an answer was derived.

It is however worth noting that the format of the questions that follow might have to be altered so that a computer can have the ability to validate the answers provided a lot more easily.

Note: Possible answers are shown in red.

1. Describe two problem-solving techniques that can be used to help you more easily tackle a programming question.

    1. **Divide and Conquer:** Do not begin by writing code. Gain an understanding of what the goal of the program is and break it down into logical modular components. Each component can then be thought of as its own smaller and easier to manage problem.
    2. **Abstract to models:** Develop a UML (or equivalent) diagram representation of how you plan to organize the program you are producing. This technique allows for better thought organization and helps to define how different modular components will behave.

2. Which of the following is not a valid primitive data type in Java? **[select one that applies]**

    a. int
    b. double
    c. float
    d. letter
    e. char
    f. All are valid

3. Check off all the statements below that are **true**.

□ A variable can be declared to begin with either a digit or a number.
■ If `x = 2`, when the statement `y = x++;` executes, y will be 2 and x will be 3.
□ A Java program does not have to necessarily be part of a class structure.
■ In the statement, `x = keyboard.readLine();` x is considered a variable.
■ A logical (boolean) expression cannot be assigned to a variable of type `double`.
■ It is possible that the statements within the body of a loop will never execute.

4. List all the steps involved in setting up a Java programming environment using Textpad.

1. Download Textpad for free from: http://textpad.com/download/index.html (latest stable version is recommended). **Do not install yet!**
2. If you have not already done so, download and install the latest Java SE Development Kit (JDK) found here: http://www.oracle.com/technetwork/java/javase/downloads/index.html
3. After the JDK has been installed, proceed to install Textpad.
4. Start the Textpad application.
5. In the top menu, select "Preferences > Configure".
6. A popup menu will appear from which you must select "Compile Java".
7. On the right side, you'll notice "parameters"; change the text in the box to: -classpath . $File
8. Finish by clicking OK to apply the changes.

5. Describe <u>at least two advantages</u> of Object Oriented Programming over sequential programming:

A few possible answers include:

It improves re-usability:
The same code can be easily reused in another program.
Reduced maintenance:
Detecting errors and making fixes is simplified as everything is much more modular.
Real-World modeling:
Complexity is minimized and overall, the program is more clear.

6. **Programming Question**

Write a Java program that asks the user to input an individual's full name (first and last name) followed by the individual's annual gross salary (salary before taxes). Calculate the net salary (salary after taxes) and output the following: first name, last name, gross salary, net salary, and the amount taken in taxes.

Your program should include the following:

- Tax rate is determined by salary bracket:
  - Over $150,000 (50% annual tax rate)
  - Over $100,000 to $150,000 (46% annual tax rate)

  - Over $71,000 to $100,000 (40% annual tax rate)
  - Over $43,000 to $71,000 (35% annual tax rate)
  - Over $32,000 to $43,000 (32% annual tax rate)
  - Less than $32,000 (28% annual tax rate)

- Include validation:
  - If the salary provided is not a number, prompt the user with an error and ask them to re-input a correct salary.

- **calculateTaxes(double salary, double taxRate) method –** Given the gross salary and the correct tax rate, this method should calculate and return the amount taken in taxes.

- **toString() method –** This method will concatenate the values and output the result to the screen.

- Create any number of private methods that you require to accomplish the tasks outlined in this question.

```
public class calculate {

  // Global variables
  double taxRate;
  double grossSalary;
  double netSalary;
  double taxAmountTaken;
  String fullName = “”;

  public static void main(String[] args) {

           this.getUserInformation();
           taxRate = this.getTaxRate(grossSalary);
           taxAmountTaken = this.calculateTaxes(grossSalary,taxRate);
           netSalary = this.calculateNetSalary(grossSalary,taxRate);
           this.toString();
           System.out.println(str);
           System.exit(0);
  }

  public double calculateTaxes(double salary, double taxRate) {
           return (salary * taxRate);
  }

  public string toString() {
           String str = “”;
           str += “Fullname: ” + fullName + “\n”;
           str += “Gross Salary: ” + grossSalary + “\n”;
           str += “Net Salary: ” + netSalary + “\n”;
```

```
        str += “Tax Rate: ” + taxRate + “\n”;
        str += “Amount Taken: ” + taxAmountTaken;
        return str;
    }

    private getUserInformation() {
        Console console = System.console();
        fullName = console.readLine("Please enter your first and last name: ");
        grossSalary = console.readLine("Please enter your annual salary (gross): ");
    }

    private double getTaxRate(double salary) {
        double rate;
        if (salary > 150,000) {
            rate = 0.5;
        } else if (salary > 100,000 && salary <= 150,000) {
            rate = 0.46;
        } else if (salary > 71,000 && salary <= 100,000) {
            rate = 0.40;
        } else if (salary > 43,000 && salary <= 71,000) {
            rate = 0.35;
        } else if (salary > 32,000 && salary <= 43,000) {
            rate = 0.32;
        } else {
            rate = 0.28;
        }
        return rate;
    }

    private double calculateNetSalary(double salary, double taxRate) {
        return salary – (salary * taxRate);
    }

} // End class
```

Level 3

The challenge in evaluating at this level is that students taking this training are in their first of six semesters which implies that they would only be entering the workforce in a little under three years. Even when they do enter the workforce, chances are a lot of the students will have accepted a position that does not involve Java programming.

Since there are too many logistical difficulties that lie with assessing these learners once they have started working professionally, a proposed alternative is to observe the learners rate of improvement on the knowledge of the subject within the context of their course.

The proposed evaluation would take place 5 to 6 weeks after the student has completed the online training or at the end of the semester – depending on what comes first. **The learners Java programming teacher would be asked to fill out a survey based on the observations they made after marking the learner's most recently submitted assignment. The goal of the survey will be to determine if the learner has retained and applied the knowledge gained from the online training.** [It is however worth noting that it is impossible to determine whether the learner's long-term success is based predominately on them having completed the training.]

Student Name: ___________________________ Student ID: __________________

Answer the following questions based on the student's most recently submitted assignment.

Question	Y/N	Comments/Notes
Was the student referred to the Mathematics and Science Centre?		
Did the student seek help during your office hours on more than one occasion?		
Has the student's grades on assignments improved throughout the course of the semester?		
Has the student demonstrated an understanding of conditional statements?		
Has the student demonstrated an understanding of loop statements?		
Has the student demonstrated an understanding of methods/functions?		
Does the student understand how Java classes work and how objects can be instantiated?		
Has the student submitted an assignment that compiles without any error messages?		
Is the student's assignment solution approach logical and well structured?		
Additional Comments		

References

Carliner, Saul. (2003). *Training design basics*. Alexandria, VA: ASTD Press.

College, Vanier. (2008). Strategic Plan 2008-2013. Retrieved October 18, 2012, from http://www.vaniercollege.qc.ca/pub/smp.pdf