US-China Foreign Language, March 2019, Vol. 17, No. 3, 97-106
doi:10.17265/1539-8080/2019.03.001

A Roadmap for Metacognition and Computational Science

Erin M. Hodgess
Western Governors University, Salt Lake City, United States

We consider some of the aspects of metacognition as potential teaching tools for computational science. We present definitions within metacognition, along with SMART goals and the concept of a growth mindset. A project is produced for consideration. This project involves learning such varied computational tools as R, Fortran, the Message Passing Interface (MPI), and OpenACC. A test over the material is produced, along with a website for the students' use for reflection. Using the test and the website as a guideline, we look at a simulation study to determine possible recall. We find that the highest grades show significant decline in recall, while the remaining grades stay fairly consistent. Finally, we discuss the implications of the study results and future research.

Keywords: metacognition, programming languages, simulation

Introduction

Metacognition is being used through all topics of learning. The concept of knowing about knowing helps students process their strengths and weaknesses for any subject. While there is a wealth of knowledge in artificial intelligence, there are not typically many applications for teaching in the STEM disciplines. An interesting exception to this for teaching physics is found in Mazur (1997). We will present a potential project for use in computational science. This would typically represent a half or an entire semester of material for an undergraduate program. If students have NVIDIA cards in their laptops, it will be possible to complete the entire set of work on those. If not, instructors could obtain educational accounts through XSEDE.

In Section 2, we will present some basic definitions of metacognition, along with SMART goals, and a growth mindset. Section 3 contains the construction of the potential project, with its tools, testing, and websites. The simulation study is described in the following section. The results of the study are discussed in Section 5, and we provide a brief conclusion in Section 6. We present the exam and the website in the Appendix.

Overall Definitions of Learning

Metacognition

The definition of metacognition is "the set of capacities through which an operative cognitive subsystem is evaluated or represented by another subsystem in a context-sensitive way" (Proust, 2013, p. 32). A system will operate when there is a task which appears wrong, or a task which has great meaning. Metacognition is part of mental agency, by being aware of one's actions through self-probing and post-evaluation (Proust, 2013). We can think of mental agency as being the processor or controller of one's actions. Part of these processes involves a mental feedback loop, known as "test-operate-test-exit" or TOTE loops (Miller, Galanter, & Pribham, 1960).

Erin M. Hodgess, Ph.D., Course Instructor, General Education Mathematics, Western Governors University, Salt Lake City, United States.

The first test indicates the presence of an error, or a possible difficult test. We can think of this as viewing a difficult exam problem. The operate step involves the actions necessary to stabilize the situation. The next test step will be to review the progress made. If the overall goal has been met, then the user exits. Otherwise the process is repeated. This has been demonstrated with programmers and programming languages with read-eval-print loops (REPL) (Larson, 2019). Larson states that “the quality of feedback loops decreases inversely with the amount of effort or time required to receive feedback”. This can be said of the entire programming process as well. The faster and more abundant the feedback from the program itself is, the more efficient the programmer becomes. Obviously, we strive for correctness during this process, particularly in programming. There are two types of measurable correctness: resolution and calibration (Proust, 2013). Resolution is a condition in which a person uses his or her judgement of knowledge to identify an item. Calibration reflects a prediction of average correctness in an overall test. Resolution is important in such items as multiple choice or matching questions. Calibration tends to be more of the big picture approach. Both impact study habits. However, resolution may propel a student to sheer memorization, rather than the full understanding of concepts.

A discussion is presented on the feeling of knowledge (FOK), when a target is not recalled from memory, but a monitoring of the basic knowledge takes place (Koriat-Laughlin). The two main concepts here are monitoring and retrieval. The recall process is quite similar to the search for a file on a computer (see Figure 1). There is an abstraction called the tip of the tongue state, in which the name or idea is on the tip of the tongue. In terms of computation, this may be searching for a file using patterns rather than an exact name. This knowledge is useful because it gives the agent a frame of reference to locate the missing image. For instance, a person may not remember the exact form of a theorem, but he or she may know that the theorem is found in a certain textbook. The trace access model is considered thoroughly as well. The trace is the memory segment, while the access is the memory retrieval process. This model provides an excellent structure for testing the validity and accuracy of the FOK concept. There is a correlation between the information available at time 1 and the accuracy of recognition at time 2. Naturally, the time lag is critical in this process. These can be somewhat difficult to record, as are most longitudinal studies. There are references to several empirical studies that indicate that both ease of access and the amount of available knowledge support the FOK validity (Koriat, 1994).

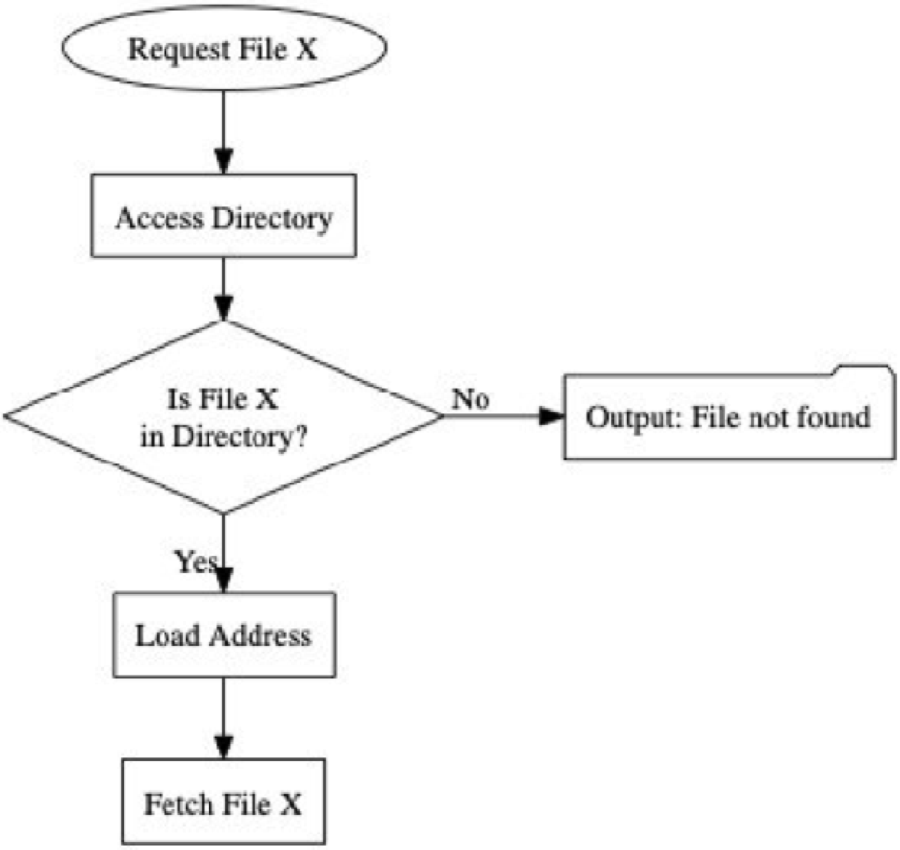


Figure 1. Computing Process.

SMART Goals

SMART goals have become quite popular in current literature, in business, learning, and personal growth. The SMART stands for specific, measurable, achievable, relevant, and time-bound. We will consider one of the examples given in Bardowell (2018), which is designed specifically for college students. The goal is that of "I want to get better grades in my courses".

- Specific: I want to get an A- in each course.
- Measurable: Over the next week, I will make appointments with each of my instructors to get study strategies.
- Achievable: On the day after tomorrow, I will create a time schedule for this semester.
- Relevant: I want to realize my full potential as a student.
- Time-Bound: By the next major exams, I will be scoring A- or higher grades in my courses.

We can approach this in terms of the goal of learning R for our course.

- Specific: I want to have a thorough knowledge of the R programming language, both theoretical and practical.
- Measurable: I will join some of the online R help lists in order to learn what tasks people are performing. I will spend at least an hour per day either reading or programming.
- Achievable: I will create a list of tasks and homework problems to complete before the exam.
- Relevant: I want to learn this as part of my student career, as well as potential future employment.

- Time-Bound: I will be performing at a high level on my course and next exam.

Growth Mindset

Mindset, in particular growth mindset, is another popular concept. Essentially, there are two particular mindsets: fixed and growth. According to Dweck (2010), a person with a fixed mindset believes that talent alone defines success, while a person with a growth mindset believes that hard work can supplement talent in order to improve a situation. Fixed mindset people are often afraid to answer questions in class and are worried about making mistakes. Growth mindset people know that mistakes are part of the learning process and can lead to greater success while gaining more knowledge. Some people embrace the hard work associated with difficult projects, while others tend to avoid it. In an interview with Carol Dweck (Gross-Loh, 2016), she noted that while praise for process is admirable, it cannot be given at every turn. Strategies must be in place to help students in order to avoid ineffective efforts. However, learners of any age and discipline can benefit from the effective use of the growth mindset. Computational science is ideal for the growth mindset, due to the intense and iterative process involved. Mistakes in programming, problem definition, and application are quite common. But obtaining the solutions can be quite exciting and lead to further growth.

Naturally, even typically growth mindset people cannot always be “on”. There is a new strategy of “yet”, as seen in Back (2018). When learners say that they don’t understand an idea, the response is, “you don’t understand it YET”. This is a much more supportive process for the learners. The yet concept can be particularly effective for adult learners, since many of them have a time gap in their educational attempts. For computational science, mathematics, and indeed any sort of STEM discipline, the yet idea is revolutionary. Not every program compiles on the first attempt, nor the first proof does not always support the support the empirical results. But it may yet.

A Description of the Project

In this experiment, students will learn such topics as the statistical language R, the Fortran programming language, along with the directives of the Message Passing Interface (MPI) and OpenACC (R Core Team, 2018; Markus, 2012; Pacheco, 1997; Chandrasek & Jukeland, 2018). As previously mentioned, students may either use their own computers or supercomputers. Actually, many gaming computers have NVIDIA graphics cards, so many students can produce the work on their computers. We have created an R package to supplement the MPI and OpenACC directives.

These packages, language, and directives are useful both in the classroom and in the work force. R is one of the most popular statistics packages, not only because it is free, but because of the incredible number of contributed packages. As of January 2019, there are 13,683 contributed packages on all kinds of statistical ideas. Some people have wondered why we are using Fortran, rather than C or C++. In a 2018 paper, Hodgess and Mhoon discovered the amazing speed of Fortran when combined with the other tools. They initially started with C but found that Fortran was significantly faster. We use the Portland Group Inc. (PGI) Fortran compiler. That is free and easy to download. Next, we obtain the MPICH version of MPI. There are two versions of MPI: OpenMPI and MPICH. We found that the MPICH works better with the Windows laptop. Both versions are free and can be easily installed. Finally, the OpenACC directives are used with the PGI compiler. If a user would prefer the GNU tools, the latest versions will support OpenACC directives, for C, C++, and Fortran.

Students will be encouraged to work together on projects, as well as explaining concepts to each other, as described in Mazur (1997). Explanations to others indicate how well a person actually understands the concept. The effectiveness of peer learning significantly increases the mean scores of exams, as noted in Mazur (1997).

We developed an eight question, 60 point exam to be used as the midterm, or possibly the final, depending on the pace of the course. There are programming questions, a pseudocode questions, and a few multiple choice questions. We modified the Exam Wrappers mathematics found at Lovett (2018) to construct our website for the students to enter their post-test reflections. There are various sections for basic test issues and software issues. The responses are recorded and stored in a text file for easy processing.

The Simulation Study

For our simulation study, we used 1,000 computer generated sets of responses. There were 10,000 replications of the entire process. We used the values from the software sections as our combinations; easy or difficult for R, Fortran, MPI, and OpenACC. We randomly generated exam scores for each response, based on our 60 point exam. For the initial scores, we utilized a standard normal distribution. We also wanted to consider learning recall over time. We simulated two sets of time lags, using exponential distributions with different rates. The first had a rate of 2, while the second had a rate of 5. The study was run on R and took 5.3 minutes.

Discussion of the Study

We grouped our results by letter grade. We let 90 percent and above represent an A, 80 through 89 percent as a B, 70 through 79 percent as a C, 60 through 69 percent as a D, and anything less as an F. Hence, we had three groups per grade, normal, exponential with the rate of 2, and exponential with the rate of 5.

We used Fisher's test to compare the results by grade. We obtained the p values for each test. We calculated the mean p value for each grade, along with the standard error. We found that there is a significant difference in recall for A grades. There are no significant results for the remaining grades. The standard errors for each level are quite stable (see Table 1). Not surprisingly, the recall impacts the highest grade most strongly. This supports the discussion of the impact of a time lag on FOK (Koriat, 1994). Using the different distributions also impacts the ease of access. Presumably, as the distributions tail off, the higher grades will fall accordingly.

Table 1

Simulation P Values by Grade

	A	B	C	D	F
Mean	0.058	0.393	0.500	0.501	0.500
SE	0.002	0.003	0.003	0.003	0.003

Conclusion

We would like to teach a class using these methods. This is an interesting thought experiment, but having students test out the methods, particularly with some recall, peer interaction, and instructor re-evaluation would be most gratifying. If that is not the case, more simulation studies which take student-to-student interaction into account, would be enlightening.

References

Back, J. (2018). Mindset and the power of "yet". Retrieved from https://www.gettingsmart.com/2018/02/mindset-and-the-power-of-yet (accessed 13 January 2019)

Bardowell, N. (2018). Five examples of smart goals for college students. Retrieved from https://itsallyouboo.com/examples-of-smart-goals-for-college-students (accessed 13 January 2019)

Chandrasek, S., & Jukeland, G. (Eds.). (2018). *OpenACC for programmers: Concepts and strategies.* Boston, MA, USA: Addison-Wesley.

Dweck, C. (2010). Mindset: The new philosophy of success. Retrieved from http://mindsetonline.com/whatisit/about/index.html (accessed 13 January 2019)

Gross-Loh, C. (2016). How praise became a consolation prize. Retrieved from https://www.theatlantic.com/education/archive/2016/12/how-praise-became-a-consolation-prize/510845/ (accessed 13 January 2019)

Hodgess, E., & Mhoon, K. (2018). Advances in ordinary kriging using the stampede and bridges supercomputers. *Journal of Geological Research and Engineering, 6,* 14-18.

Koriat, A. (1994). Memory's knowledge of its own knowledge: The accessibility account of the feeling of knowing. In J. Metcalfe and A. Shimamura (Eds.), *Metacognition: Knowing about knowing* (Chap. 6, pp. 115-135). Cambridge, MA, USA: MIT Press.

Larson, W. (2009). Feedback loops in software development. Retrieved from https://lethain.com/feedback-loops-in-software-development (accessed 13 January 2019)

Lovett, M. (2018). Exam wrappers. Retrieved from https://www.cmu.edu/teaching/designteach/ teach/examwrappers/ (accessed 13 January 2019)

Markus, A. (2012). *Modern Fortran in practice*. New York, NY, USA: Cambridge University Press.

Mazur, E. (1997). *Peer instruction: A user's manual*. Upper Saddle River, NJ, USA: Prentice Hall.

Miller, G., Galanter, E., & Pribham, H. (1960). *Plans and the structure of behavior*. New York, NY, USA: Holt.

Pacheco, P. (1997). *Parallel programming with MPI*. San Francisco, CA, USA: Morgan Kaufman.

Proust, J. (2013). *The philosophy of metacognition: Mental agency and self awareness*. New York, NY, USA: Oxford University Press.

Proust, J. (2014). Précis of the philosophy of metacognition. *Philosophy and Phenomenological Research, 89*(3), 703-709.

R Core Team. (2018). R: A language and environment for statistical computing. R foundation for statistical computing, Vienna, Austria. Retrieved from https://www.R-project. org/

Appendix

Exam 1

(10 points) 1. Write an R function for the Fibonacci series.

(5 points) 2. Write the pseudocode for a "Hello World" program with MPI.

(5 points) 3. Which of the following constants has the value 5000.0?

a. 0.5*e*-4 b. 5.0*e*-4 c. 5.0*e*4 d. 5.0*e*-3 e. 5.0*e*3

(5 points) 4. Write a logical assignment statement to carry out the following operation (not an if statement). Assign a value of true to an even number if M is an even number, otherwise assign a value of false. Hint: use the Mod function.

You may use either R or Fortran.

(5 points) 5. Which group of statements uses OpenACC to perform the following operation correctly: find $e^{\sin(a(i))}$ for $i = 1, n$.

a.
```
do i=1,n
b(i)=exp(sin(a(i)))
enddo
```

b.
```
b=exp(sin(a))
```

c.
```
call MPI_COMM_RANK(MPI_COMM_WORLD,myid,ierr)
 do i=myid+1,n,nprocs
 b(i)=exp(sin(a(i)))
 enddo
```

d.
```
!$acc parallelloop
do i=1,n
b(i)=exp(sin(a(i)))
enddo
!$acc end parallel loop
```

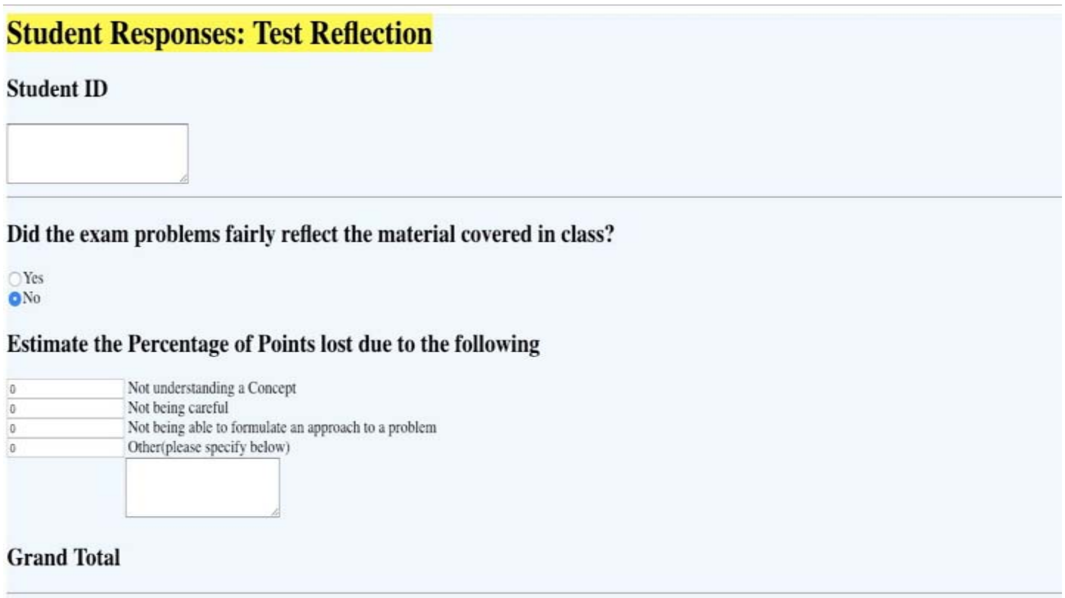
(5 points) 6. Which of the following statement corrects brings a Fortran program "test.o" into R?

a. library(test.o) b. dyn.load("test.o")

c. load("test.o") d. read.table("test.o")

(5 points) 7. What is the difference between an interpreted language and a compiled language?

(20 points) 8. This is complicated, so read the directions thoroughly. Write an R function which calls a Fortran program. That program uses OpenACC to sum the numbers from 1 to 10,000. Return the total to R.



Student Responses: Test Reflection

Student ID

Did the exam problems fairly reflect the material covered in class?

Yes
No

Estimate the Percentage of Points lost due to the following

0 Not understanding a Concept
0 Not being careful
0 Not being able to formulate an approach to a problem
0 Other(please specify below)

Grand Total

(10 points) 1. Write an R function for the Fibonacci series.

(5 points) 2. Write the pseudocode for a "Hello World" program with MPI.

(5 points) 3. Which of the following constants has the value 5000.0?

a. 0.5*e*-4 b. 5.0*e*-4 c. 5.0*e*4 d. 5.0*e*-3 e. 5.0*e*3

(5 points) 4. Write a logical assignment statement to carry out the following operation (not an if statement). Assign a value of true to an even number if M is an even number, otherwise assign a value of false. Hint: use the Mod function.

You may use either R or Fortran.

(5 points) 5. Which group of statements uses OpenACC to perform the following operation correctly: find $e^{\sin(a(i))}$ for $i = 1, n$.

e. do i=1,n
b(i)=exp(sin(a(i))) enddo

f. b=exp(sin(a))

g. call MPI_COMM_RANK(MPI_COMM_WORLD,myid,ierr)
do i=myid+1,n,nprocs
b(i)=exp(sin(a(i))) enddo

h. !$acc parallelloop
do i=1,n
b(i)=exp(sin(a(i))) enddo
!$acc end parallel loop

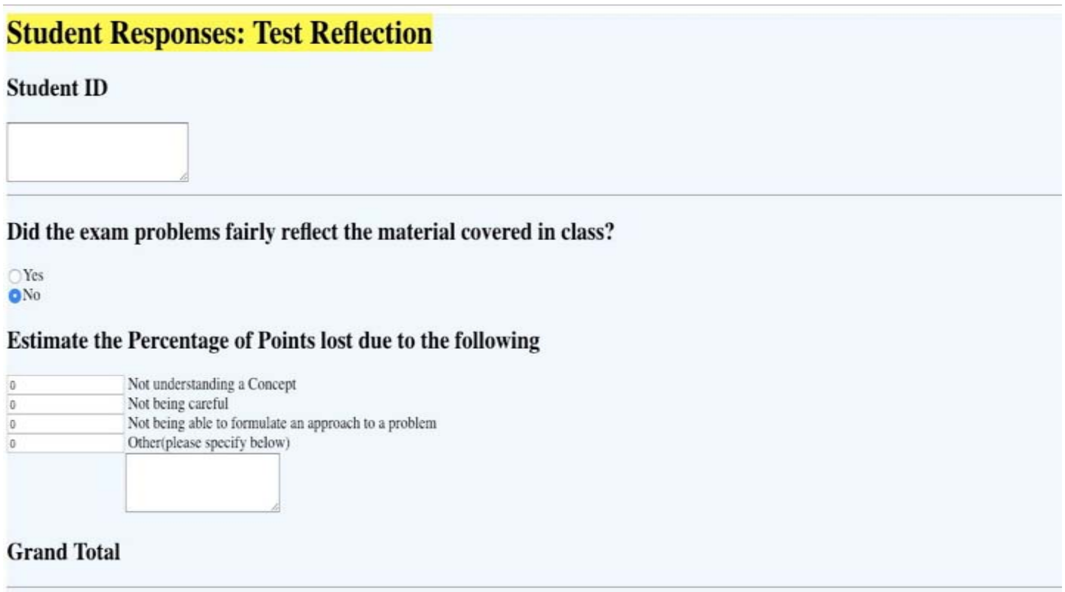
(5 points) 6. Which of the following statement corrects brings a Fortran program "test.o" into R?

a. library(test.o) b. dyn.load("test.o")

c. load("test.o") d. read.table("test.o")

(5 points) 7. What is the difference between an interpreted language and a compiled language?

(20 points) 8. This is complicated, so read the directions thoroughly. Write an R function which calls a Fortran program. That program uses OpenACC to sum the numbers from 1 to 10,000. Return the total to R.



Student Responses: Test Reflection

Student ID

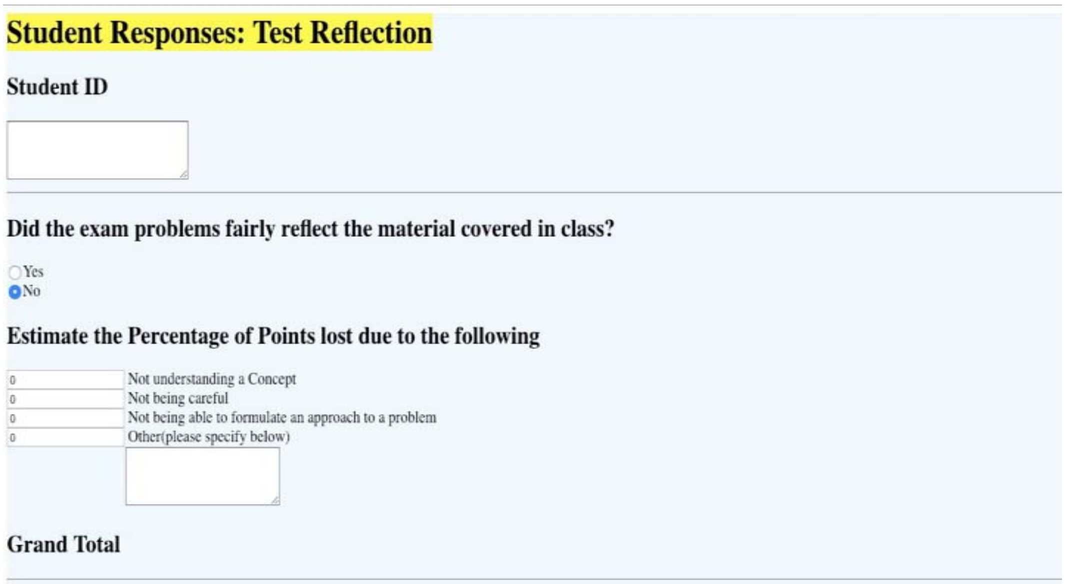
Did the exam problems fairly reflect the material covered in class?

Yes
No

Estimate the Percentage of Points lost due to the following

0	Not understanding a Concept
0	Not being careful
0	Not being able to formulate an approach to a problem
0	Other(please specify below)

Grand Total



Student Responses: Test Reflection

Student ID

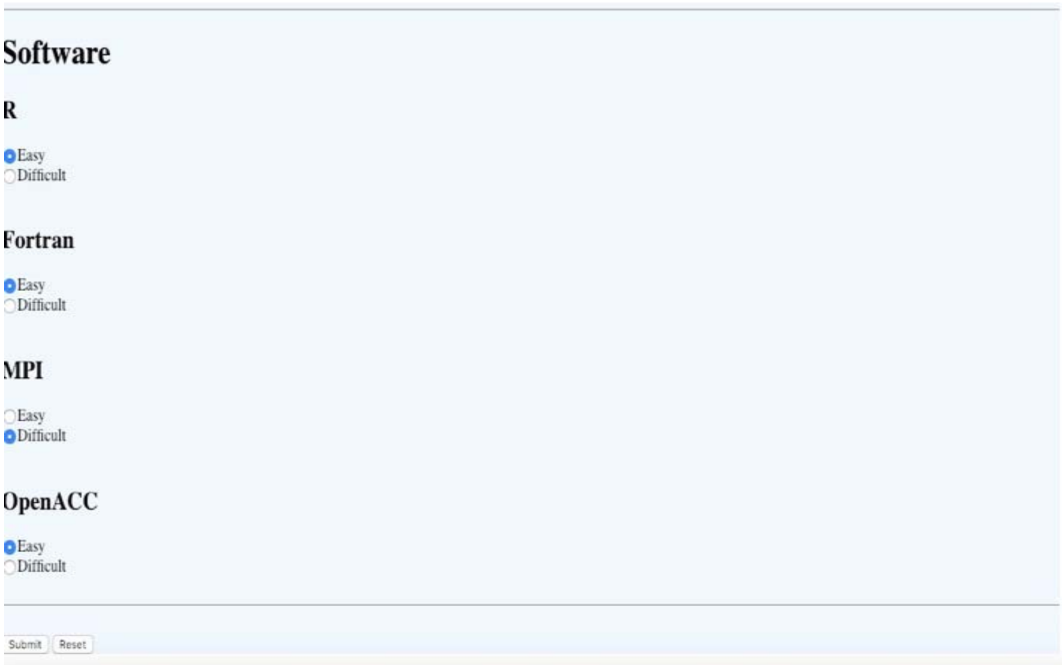
Did the exam problems fairly reflect the material covered in class?

Yes
No

Estimate the Percentage of Points lost due to the following

0	Not understanding a Concept
0	Not being careful
0	Not being able to formulate an approach to a problem
0	Other(please specify below)

Grand Total


Software
R
Easy
Difficult
Fortran
Easy
Difficult
MPI
Easy
Difficult
OpenACC
Easy
Difficult
Submit
Reset