You don't have a submission that is not late.

minutes remaining

Hide Time

Manual Save

Navigation

- Question 1
- Question 2
- Question 3
- Question 4
- Question 5
- Extra Credit
- Pledge & Submit

Question 1. Word Bank Matching (1 point each, 14 points)

For each statement below, input the letter of the term that is *best* described. Note that you can click each word (cell) to mark it off. Each word is used at most once.

A. — A/B Testing	B. — Agile development	C. — Alpha Testing	D. — Beta Testing
E. — Competent Programmer Hypothesis	F. — Constructive Cost Model	G. — Dynamic Analysis	H. — Formal Code Inspection
I. — Fuzz Testing	J. — Milestone	K. — Mocking	L. — Oracle
M. — Pair Programming	N. — Pass Around Code Review	O. — Priority	P. — Race Condition
Q. — Regression testing	R. — Risk	S. — Severity	T. — Spiral Development
U. — Streetlight Effect	V. — Triage	W. — Uncertainty	X. — Unit Testing
Y. — Waterfall Model			

Q1.1:

B

Miscord recently started working on a new project. Ben is in constant contact with the stakeholders, modifying the project as their needs change.

Q1.2:

X

EECSon Mobile has a policy that for every created function, there must be corresponding inputs, outputs and oracles to assess that function.

Q1.3:

P

Livia decides, on a whim, to watch the latest Parvel movie, Avengers: End of Germs. Upon arriving at the theater, they discover that someone else is already sitting in their assigned seat. It turns out both people bought tickets online at the last minute and were assigned the same seat.

Q1.4:

K

Aang is developing a social media app and designing the user interface. To verify that the interface displays information properly, they are simulating the backend data using a static JSON file.

Q1.5:

C

Veecsa is preparing to launch their latest application, Adobe Shotofop. Prior to making it available to the general public, they plan to have a group within the company evaluate it to uncover potential issues.

Q1.6:

R

Omkar has recently been brainstorming things that could go wrong in an effort to manage *this*.

Q1.7:

S

Boogle is finding maintenance to be a very expensive part of the development process. This is due to Boogle not accurately identifying the impact each bug has on the business.

Q1.8:

L

Antony is developing a program to determine whether a number is prime. They double-check corner cases, such as the inputs 0, 1 and 2, to be sure of what the answers should be.

Q1.9:

T

PiedPiper is developing a new software product and plans to repeat this process until completion: create a prototype, gather user feedback, mitigate risk, and make an improved prototype.

Q1.10:

G

FloorMart is a travel agency that arranges global tours. They are encountering problems where customers can book tickets for tours that are already sold out because multiple bookings are processed simultaneously. Harry proposes using *this* technique to identify the root cause of this issue automatically.

Q1.11:

D

The highly-anticipated game, AldenRing, is about to be launched. Six months before its release, Devforce distributes a pre-release version to a select group of users.

Q1.12:

N

Mesla Inc. recently found that developers were making commits directly to the main branch without input from others. To address this, they introduced a policy requiring at least one other developer familiar with the codebase to review and approve any changes before they are committed.

Q1.13:

H

MunchyRoll recently hired new developers who have expressed concerns about critical code being difficult to read and filled with bugs. To address this, Gwen, a senior developer, decides to conduct a series of structured meetings with the team to review and resolve the issues in that code.

Q1.14:

J

Yushu is working on a new game for GogoWeimer. GogoWeimer sets a goal to complete the physics module within the next two months.

Question 2. Code Coverage (25 points)

You are given the following code. (You can scroll down to see the all the code) In this question, we consider the entire program when calculating coverage.

minutes remaining

Hide Time

Manual Save

Navigation

- Question 1
- Question 2
- Question 3
- Question 4
- Question 5
- Extra Credit
- Pledge & Submit

```
void coverage(bool a, bool b){
  if (a || b) {
    std::cout << "1";
  }
  if (!b || !b || !b || !b) {
    std::cout << "2";
  }
}

```

(a) (4 points)

What is the minimum number of test cases required for 100% statement coverage? Enter a whole number.

Your answer here.

ANSWER: 1

(b) (4 points)

What is the minimum number of test cases required for 100% branch coverage? Enter a whole number.

Your answer here.

ANSWER: 2

(c) (4 points)

What is the minimum number of test cases required for 100% path coverage? Enter a whole number.

Your answer here.

ANSWER: 3 (There are only 3 **feasible** paths, so covering 3 paths is 100%)

Now consider the `silly_goose` function. Answer the following questions.

```
void silly_goose(bool a, bool b, bool c){
    if ((a || !b) || (c && !a)) {
        STMT_1;
    }
    if ((c && !b) || (a || b)) {
         STMT_2;
    }
    if ((!a && !b) && c){
        STMT_3;
    }
}

```

minutes remaining

Hide Time

Manual Save

Navigation

- Question 1
- Question 2
- Question 3
- Question 4
- Question 5
- Extra Credit
- Pledge & Submit

(d) (4 points)

How many of the STMT statements does the test case `(true, true, true)` cover? Write your answer as a whole number. (We are asking for the statement coverage without the denominator.)

Your answer here.

ANSWER: 2

(e) (4 points)

How many of the branch directions does the suite `[(false, false, true), (false, true, false), (false, false, false)]` cover? Write your answer as a whole number. (We are asking for the branch coverage without the denominator.)

Your answer here.

ANSWER: 6

(f) (5 points)

Would EECS481 Homework 1b have been easier or harder if students had only been asked to provide 1% *(acylic) path* coverage for full points (instead of the listed coverage requirements)? Justify your answer in 5 sentences or less. Your answer should reference details about the program in question.

Your answer here.

ANSWER:
+1 Concludes that 1% path coverage would be harder than original requirements

+2 Mentions the fact that the number of paths grows exponentially/quickly as the complexity of the program (e.g., number of conditional statements like number of ifs) increases.

+1 Mentions or describes HW1b (libpng) and mentions number of conditionals/lines in the libpng program

+1 Mentions that path coverage is harder than branch coverage and/or statement coverage

Sample Answer:

It would almost certainly be harder.

In class we discussed how the number of acyclic paths through a method with N conditional statements grows with 2^N. Each input can only visit one path, so obtaining coverage of 2^N paths requires 2^N inputs. By contrast, branch coverage for N conditional statements can, in the best case, be covered with two inputs: one that always takes true paths and one that always takes false paths.

The question asks specifically about libpng from HW1b. In that program, the png_read_info() method alone (one single 170-line method in the 95,000 line program) has 30+ conditional statements. Obtaining 1% path coverage of such a method could require 10,737,418 test inputs (1% of 2^30) — much, much more than the 50 that suffice for 36% branch coverage. Even if the math approximation is off "slightly" (say, by a factor of 100,000), path coverage is still harder.

Question 3. Short Answer (24 points)

minutes remaining

Hide Time

Manual Save

Navigation

- Question 1
- Question 2
- Question 3
- Question 4
- Question 5
- Extra Credit
- Pledge & Submit

This question (a-b) concerns bug triage. Consider the bug report problems from the reading *"What Makes a Good Bug Report?"*. For each bug below, list an **uncommon** problem (as reported by that reading) that could cause an organization to mistakenly assign the bug report the wrong priority, then give a potential example of such a scenario using a real, specific tool or assignment from EECS 481 as a basis.
For each bug, please limit your answer to at most 4 sentences for that bug.

(a) (3 points)

A bug that causes dropdown menu items to be displayed out of order.

Your answer here.

ANSWER:

+1 Valid uncommon problem from reading that relates to this specific bug

+0.5 If the problem is common (eg. incomplete information, steps to reproduce)

+1 Valid explanation of why it could cause organization to mistakenly assign this specific bug wrong priority

+1 Good example from EECS 481 that relates to the given bug

Sample Answer: You were given the wrong expected behavior. Some user of the eecs481 website reports that the current homework spec isn't in the right place in the dropdown menu and they struggled to find it at first. In reality, they expected the hw4 spec to show up first since it is the current homework, but the 481 website lists homeworks by chronological order. This could cause this "bug" to be given priority that it shouldn't be since it isn't really a bug.

(b) (3 points)

A bug that causes some images to swap locations.

Your answer here.

ANSWER:

+1 Valid uncommon problem from reading that relates to this specific bug

+0.5 If the problem is common (eg. incomplete information, steps to reproduce)

+1 Valid explanation of why it could cause organization to mistakenly assign this specific bug wrong priority

+1 Good example from EECS 481 that relates to the given bug

Sample Answer: You were given the wrong version number. The bug reporter may be using an earlier version, but report that they are using the latest version, causing the organization to mistakenly assign too much priority to the bug. The hw0 spec on last semester's EECS 481 website could be putting some images in the wrong place, and this could mistakenly be reported as this semester's website.

Each of the following questions (c)-(e) gives a pair of concepts. It can be a pair of techniques, a pair of tools, or a pair of processes, etc. Consider the reading *"Analyze This! 145 Questions for Data Scientists in Software Engineering"*. For each pair, choose a quoted respondent question from Section 4.1 of that reading, copy the exact quote into the answer space, and then argue that one element of the pair would be better at answering that quoted question than the other.

For example, given the pair **Code Inspection vs. Measurement**, you might copy the quote "How long does it take to gain a payoff from moving to new software-writing tools?" and then argue that Measurement is more relevant to answering that question.

For each question, after copying the quote, please use at most four sentences. Use each quote at most once.

minutes remaining

Hide Time

Manual Save

Navigation

- Question 1
- Question 2
- Question 3
- Question 4
- Question 5
- Extra Credit
- Pledge & Submit

(c) (3 points)

A/B Testing vs. Risk

Your answer here.

ANSWER:

Quote

+1 Student correctly provided quote from the reading that is relevant to A/B Testing and Risk

+0.5 quote provided but is irrelevant to A/B Testing and Risk

+0 No quote or quote provided was not in the reading

Argument quality

+1 Student clearly explains why either A/B Testing or Risk is better suited, providing logical and well-supported reasoning

+0.5 Argument is valid but underdeveloped or lacks clarity (e.g. sufficient support for claims are not provided, student does not clearly identify which option would be better)

+0 No argument provided or argument is invalid

Conciseness

+1 Student stayed within 4 sentences

+0 Student went 3+ sentences over the limit

(d) (3 points)

Regression Testing vs. Integration Testing

Your answer here.

ANSWER:

Quote

+1 Student correctly provided quote from the reading that is relevant to regression/integration testing

+0.5 quote provided but is irrelevant to regression/integration testing

+0 No quote or quote provided was not in the reading

Argument quality

+1 Student clearly explains why either Regression Testing or Integration Testing is better suited, providing logical and well-supported reasoning

+0.5 Argument is valid but underdeveloped or lacks clarity (e.g. sufficient support for claims are not provided, student does not clearly identify which option would be better)

+0 No argument provided or argument is invalid

Conciseness

+1 Student stayed within 4 sentences

+0 Student went 3+ sentences over the limit

minutes remaining

Hide Time

Manual Save

(e) (3 points)

Fuzz Testing vs. the Eraser Dynamic Analysis

Your answer here.

ANSWER:

Quote

+1 Student correctly provided quote from the reading that is relevant to Fuzz Testing and the Eraser Dynamic Analysis

+0.5 quote provided but is irrelevant to Fuzz Testing and the Eraser Dynamic Analysis

+0 No quote or quote provided was not in the reading

Argument quality

+1 Student clearly explains why either Fuzz Testing or the Eraser Dynamic Analysis is better suited, providing logical and well-supported reasoning

+0.5 Argument is valid but underdeveloped or lacks clarity (e.g. sufficient support for claims are not provided, student does not clearly identify which option would be better)

+0 No argument provided or argument is invalid

Conciseness

+1 Student stayed within 4 sentences

+0 Student went 3+ sentences over the limit

(f) (6 points)

Suppose you are asked the following question during a technical interview (the particular programming language isn't relevant for this problem): **"Write a method that takes a singly-linked list as input and reverses that list."** Write 2 questions you can ask of the interviewer before you start implementing the solution. For each, use a direct quote from the reading *"The Google Technical Interview"* to support your answer. Copy the exact quote into the answer space as part of your answer. (Do not use a question that literally occurs in the reading, such as "How big could the input be?". Instead, create a new question and support it with a general claim from the reading.)

Your answer here.

ANSWER:

For each question: +1 pt Good question that doesn't occur in the reading. Questions that just repeated something from the reading without being more specific didn't receive this point (eg. "Can I have an example?")

+1 Valid quote from reading

+1 Supports question with the quote. Most question/quote pairs require at least a brief explanation on how the quote supports the question.

Sample partial answer (full answer needs two questions):

Should the list be reversed in place or should we return a new list?

"First, make sure you understand the problem properly. It helps to clarify assumptions before diving in too deeply, and if you are confused about the question, do ask for examples, or for the question to be reworded. The interviewer will often not offer information until you ask for it." This question helps clarify some assumptions about the result and output which could change the solution, and it is necessary to ask this question because often the interviewer won't always give much information about how the output should be.

minutes remaining

Hide Time

Manual Save

Navigation

You are an engineering manager who must decide whether or not to employ pair programming for a series of tasks (questions (g)-(k)).

Use the same sort of mathematical reasoning generally described in slides 18 and 19 of the lecture. For each task, write the pair programming cost divided by the solo programming cost (e.g., if the pair cost is 91 and the solo cost is 100, write 0.91). Use two figures after the decimal point (e.g., 1.23). The interpretation of the various features (e.g., "fewer total lines" or "fewer total defects") is as covered in class.

(g) (1 points)

Task 1: 50,000 LOC program, Coding at 25 LOC/hour, Defect rate of 10 defects / KLOC, Defect fix time of 20 hours / defect, Pair programming results in 15% fewer total defects, Pair programming results in 15% fewer total lines of code.

Your answer here.

ANSWER:
Individual: 2000 + 500*20 = 12000

Pair: 42500/25 + 425 * 20 = 10200

0.85

See slide 19 of the pair programming lecture for pair calculation reasoning.

Answers that mistakenly treated pair programming as 2x or 15% slower receive half-credit

(h) (1 points)

Task 2: 50,000 LOC program, Coding at 25 LOC/hour, Defect rate of 10 defects / KLOC, Defect fix time of 20 hours / defect, Pair programming results in 20% fewer total defects, Pair programming results in 10% fewer total lines of code.

Your answer here.

ANSWER:
Individual: 2000 + 500*20 = 12000

Pair: 45000/25 + 400 * 20 = 9800

0.82

See slide 19 of the pair programming lecture for pair calculation reasoning.

Answers that mistakenly treated pair programming as 2x or 15% slower receive half-credit

(i) (1 points)

Task 3: 75,000 LOC program, Coding at 50 LOC/hour, Defect rate of 10 defects / KLOC, Defect fix time of 5 hours / defect, Pair programming results in 10% fewer total defects, Pair programming results in 12% fewer total lines of code.

minutes remaining

Hide Time

Manual Save

Navigation

- Question 1
- Question 2
- Question 3
- Question 4
- Question 5
- Extra Credit
- Pledge & Submit

Your answer here.

ANSWER:
Individual: 1500 + 750*5 = 5250

Pair: 66000/50 + 675 * 5 = 4695

0.89

See slide 19 of the pair programming lecture for pair calculation reasoning.

Answers that mistakenly treated pair programming as 2x or 15% slower receive half-credit

Question 4. Mutation Testing (20 points)

Consider the following code.

```
 1 int func2(int n) {
 2   if (n <= 1) {
 3     return n;
 4   }
 5   int a = 1;
 6   int b = 4;
 7   for (int i = 1; i < n; ++i)
 8   {
 9     int c = b − (2 ∗ a);
10     a = b;
11     b = c;
12   }
13   return b;
```

(a-1) (3 points) Your mutants can swap between the operators <, == and <= (but no other operators). Your mutants can swap between the numbers 1, 0 and -1 (but no other numbers). If you are only allowed to mutate the base (line 2), how many mutants are killed by the test case with input n = 3?

Your answer here.

ANSWER: 0

(a-2) (3 points) Your mutants can swap between the operators <, == and <= (but no other operators). Your mutants cannot change anything else. If you are only allowed to mutate the loop guard (line 7), how many mutants are killed by the test case with input n = 2?

Your answer here.

ANSWER: 2

minutes remaining

Hide Time

Manual Save

Navigation

- Question 1
- Question 2
- Question 3
- Question 4
- Question 5
- Extra Credit
- Pledge & Submit

(b) (4 points)

Suppose that a group of researchers find very convincing evidence in favor of the competent programmer hypothesis but also find very convincing evidence against the coupling effect hypothesis. What would this mean for mutation testing? For credit, include a direct quote from *An Analysis and Survey of the Development of Mutation Testing* (part of the HW3 spec) to support your argument. (After copying the quote, use at most four sentences.)

Your answer here.

ANSWER:
+1 includes a quote that is relevant to their response from An analysis and survey of the development of mutation testing

+1 Defines correctly what the competent programmer hypothesis (CPH) is and what the coupling effect hypothesis (CEH) is

The CPH says that when programmers do make mistakes they are small and limited to a small section (typically one line) of code.

The CEH says that complex/additional problems emerge from multiple small mistakes (like the ones made by programmers according to the CPH) combining together

+1 Describes how the CPH and CEH relate to mutation testing

Mutation testing simulates small mistakes confined to one line. The CPH supports that the mistakes introduced by mutation testing are similar to the ones made by real developers.

Traditional mutation testing uses the CEH to justify using only first order mutants. Since the CEH says that complex problems are made up of small mistakes, complex problems can be detected simply by looking for small mistakes (i.e., first order mutants)

+1 Concludes that mutation testing cannot rely solely on first order mutants. Higher order mutants or other strategies must be used.

Since some issues are not just a combination of small problems (first order mutants), detecting only small problems (first order mutants) will not detect all issues.

Alternatively: +1 if the student claims that the CEH is unnecessary if the CPH is true. Since the CPH claims that competent programmers only make small mistakes, issues that are more complicated than a small mistake must be extremely rare / non-existent. Therefore, mutation testing can continue to use only first order mutants.

(c-1) (10 points)

Consider the following Python function `distracted_goldstine(n)`:

```python
def distracted_goldstine(n):
    if n < 0:
        return False
    low, high = 0, n
    while low <= high:
        mid = (low + high) // 2
        mid_squared = mid * mid

        if mid_squared == n:
            return True
        elif mid_squared < n:
            low = mid + 1
        else:
```

We have exactly three test cases: -1, 1 and 16.

Your task is to create two different first-order mutants of the `distracted_goldstine(n)` function by modifying **only the branch conditions** (i.e., the expressions in `if`, `elif`, and `while` statements). Create your two mutants by editing the two copies of the code below. You can edit only the lines with the branch conditions; do not change other lines.

Each of your mutants should fail a different and non-zero number of these test cases. For example, your first mutant might fail one test, while your second mutant fails two tests.

Each mutant should have exactly one modification from the original code (i.e., each of your mutants should be a first-order mutant). For example, you may change comparison operators (e.g., `==` to `!=`, `<` to `<=`), logical operators, or negate conditions. Do not add or remove entire statements; focus only on modifying the conditions.

Below are two code boxes where you will create your mutants. Click on the lines with branch conditions (they are editable) to modify them. Remember to modify only the conditions in the `if`, `elif`, and `while` statements. You should not change other lines.

minutes remaining

Hide Time

Manual Save

Navigation

- Question 1
- Question 2
- Question 3
- Question 4
- Question 5
- Extra Credit
- Pledge & Submit

Mutant 1:

```
def distracted_goldstine(n):
    if n < 0:
        return False
    low, high = 0, n
    while low <= high:
        mid = (low + high) // 2
        mid_squared = mid * mid

        if mid_squared == n:
            return True
        elif mid_squared < n:
            low = mid + 1
        else:
```

Mutant 2:

```
def distracted_goldstine(n):
    if n < 0:
        return False
    low, high = 0, n
    while low <= high:
        mid = (low + high) // 2
        mid_squared = mid * mid

        if mid_squared == n:
            return True
        elif mid_squared < n:
            low = mid + 1
        else:
```

- +1 for each mutant that is killed by 0 test cases
- +3 for each mutant that is killed by >0 test cases
- +1 if both mutants are killed by 0 test cases
- +1 if both mutants are killed by >0 test cases, but the kill number is the same
- +4 if both mutants are killed by >0 test cases and the kill number is different

Question 5. Dynamic Analysis (17 points)

You have developed a dynamic analysis tool, called Marbles, that aims to quickly identify potential memory leaks. A *memory leak* occurs when memory is allocated but never deallocated. In one run of a program, every allocation at address *x* must be paired with a deallocation at exactly address *x*. Failing to do so indicates a memory leak.

Marbles works by tracking and logging memory allocations and deallocations during program execution. Because dynamic analyses can be slow, you design Marbles so that it runs and analyzes multiple different threads of the same subject program code simultaneously. Each concurrent execution is assigned a unique thread number (e.g., Thread 1, Thread 2, etc.). Marbles collects a single unified log of information about each memory operation, potentially including the allocated memory address, deallocated memory address, size, and the thread number.

However, the Marbles instrumentation may be buggy, so the log may be missing events or details or may contain spurious events. Marbles reports memory leaks based on analyzing its log file, so this may result in false positives and/or false negatives.

In this context:
A *false positive* occurs when Marbles incorrectly reports the presence of a memory leak, but no leak is actually possible (on any

minutes remaining

Hide Time

Manual Save

execution).
A *false negative* occurs when Marbles fails to report a memory leak, but a leak is actually possible based on the source code (on some executions).

Consider the programs below. We have run Marbles on each of them. The log file for each analysis is also shown below. For each program, you are asked to determine if Marbles would report a memory leak based on the Marbles log file. In addition, you will also be asked to determine whether Marbles has incurred any false positives/negatives during the analysis process.

NOTE: You may need to scroll down on some of the code snippets to view the full program and/or log file.

```
void p1() {
    // Dynamically allocates two pointers (4 bytes each)
    int* ptr1 = new int;
    int* ptr2 = new int;
    *ptr1 = 42;
    *ptr2 = 84;
    delete ptr1;
}
```

```
Marbles log file:
Thread 1: Allocate 4 bytes at address 0x7000
Thread 2: Allocate 4 bytes at address 0x7100
Thread 2: Allocate 4 bytes at address 0x7200
```

(a-1) (1 points) **True / False**: Marbles would report a memory leak for program `p1()`, based on the log file.

○ True
○ False
ANSWER: True
The code does not deallocate memory and Marbles reports this correctly.

(a-2) (1 points) **True / False**: Marbles incurred a false positive or false negative for `p1`.

○ True
○ False
ANSWER: False
No false positives/negatives. Marbles worked correctly in this instance.

```
void p2() {
    // Dynamically allocates a new buffer (150 bytes)
    char* x = new char(150);
    if (foo) {
      char* y = &x[16];
    }
    delete[] x;
}
```

```
Marbles log file:
Thread 1: Allocate 150 bytes at address 0x8000
Thread 2: Allocate 150 bytes at address 0x8200
Thread 1: Deallocate bytes at address 0x8000
```

(b-1) (1 points) **True / False**: Marbles would report a memory leak for program `p2()`, based on the log file (the allocation at 0x8210 is not paired with a deallocation).

○ True
○ False
ANSWER: True
The Marbles log file mistakenly records the `y = &x[16]` as an allocation (of 16 bytes at 0x8210). However, there is no memory leak: that it not actually a new allocation.

minutes remaining

Hide Time

Manual Save

Navigation

(b-2) (1 points) **True / False**: Marbles incurred a false positive or false negative for p2, during its logging process.

○ True
○ False
ANSWER: True
False positive. The Marbles log file has a spurious entry, causing it to mistakenly report a memory leak.

```
void p3() {
    // Dynamically allocate arr1 (32 bytes)
    int* arr1 = new int[8];
    // Dynamically allocate arr2 (12 bytes)
    int* arr2 = new int[3];
    // Deallocate arr2
    delete[] arr2;
    // Deallocate arr1
    if (random(0,100) <= 50) {
      delete[] arr1;
    }
}
```

(c-1) (1 points) **True / False**: Marbles would report a memory leak for program `p3()`, based on the log file.

○ True
○ False
ANSWER: False
In this run, the Code correctly deallocates memory, and Marbles accurately logs these operations.

(c-2) (1 points) **True / False**: Marbles incurred a false positive or false negative for p3, during the logging process.

○ True
○ False
ANSWER: True
The code actually contains a potential memory leak, but Marbles does not report it. This is a false negative.

```
void p4(){
    // Dynamically allocates arr1 (40 bytes)
    int* arr1 = new int[10];
        // Dynamically allocates arr2 (24 bytes)
    int* arr2 = new int[6];
    // Deallocates arr2
    delete[] arr2;
    // Deallocate arr1
    delete[] arr1;
}
```

(d-1) (1 points) **True / False**: Marbles would report a memory leak for program `p4()`, based on the log file.

○ True
○ False
ANSWER: True
Although code does correctly deallocate all memory, the Marbles log does not contain the deallocation of arr2. Marbles thus

reports a memory leak.

(d-2) (1 points) **True / False**: Marbles incurred a false positive or false negative for p4, during the logging process.

○ True
○ False

ANSWER: True
False positive, Marbles did not work correctly in this instance.

(e) (3 points)

Compare and contrast two approaches for race condition detection: the dynamic analysis tool Eraser and static code inspection. Describe one situation in which the former would work well and the latter would not, then describe a situation in which the latter would work well and the former would not. Reference some of the human factors associated with code inspection from the lectures and reading. (Use at most six sentences.)

Your answer here.

ANSWER:
Answers may vary. Dynamic Analysis (e.g., Eraser using Lockset Algorithm): detects race conditions during program execution with high accuracy which results in performance overhead and limited path coverage. Static analysis provides broader code coverage without runtime impact but can often create false positives and lacks runtime context. The lecture slides discussed a number of human factors. For example, on the positive side for formal code inspection, some report that formal code inspection meetings uncover deeper, more important bugs. On the negative side, formal code inspection can typically only handle a small number of lines before attention wanes. Perhaps most critically, formal code inspection does not require any inputs to the program and does not require the code to be working correctly (e.g., to compile). By contrast, Eraser requires the program to compile and run, and it requires a good set of inputs, but after that it is automatic.

(f) (3 points)

You are developing a real-time trading system that must process market data and be able to execute trades with low latency. Describe how dynamic analysis can help ensure the performance and the reliability of your system. What is the relationship between dynamic analysis utility and test suite coverage? (Use at most 3 sentences.)

Your answer here.

ANSWER:
Answers may vary. Dynamic analysis can be useful for measuring runtime behavior, identifying performance bottlenecks, and detecting concurrency issues which may not be able to be detected by static analysis. For example, dynamic analyses can provide not only "execution time" profiling, but also measurements of particular events (e.g., how often a function is called or how often a resource is allocated). However, dynamic analyses are only as good as their inputs. For example, consider a "real-time trading system" program that uses bubblesort. If it is only tested on small examples (e.g., two or three trades that must be sorted before being processed), it may appear to meet real-time constraints. However, that same dynamic analysis applied to that program with much larger, more indicative inputs would show a much slower running time. A dynamic analysis cannot reveal information about lines of code it does not execute.

(g) (3 points)

In the lectures we discussed an implantable medical device and an associated bug. In that context, describe a situation in which a static analysis technique could be a better approach than a dynamic analysis technique to ensure security. (Use at most three sentences.)

Your answer here.

minutes remaining

Hide Time

Manual Save

Navigation

- Question 1
- Question 2
- Question 3
- Question 4
- Question 5
- Extra Credit
- Pledge & Submit

ANSWER:
Answers may vary. In class we discussed the Abboty Labs pacemaker, where a bug capable of delivering a shock to patients was only found after deployment. Static analysis can spot coding errors and design issues before deployment. In that particular instance, the instance was a security vulnerability accessible via networking. A static analysis might check that network inputs are correctly sanitized before being used. By contrast, a dynamic analysis technique might require running the pacemaker, at which point a failing test might do real damage. (A technique like mocking could mitigate this, but may be difficult to do correctly.) Note that this question just asks about static analysis in general: this could include something like code review or code inspection, not necessarily an automated static analysis.

Extra Credit

(1) What is your favorite part of the class so far? (1 point)

Your answer here.

(2) What is your least favorite part of the class so far? (1 point)

Your answer here.

(3) In the context of HW2, how would the SAGE tool from the *"Automated Whitebox Fuzz Testing"* paper compare to EvoSuite on jsoup? Demonstrate that you have read the paper critically and tie it in to your experiences with HW2 (i.e., what did EvoSuite do badly at for 481 specifically?), going beyond a Generative AI summary. (2 points)

Your answer here.

(4) If you read any *other* optional reading, identify it and demonstrate to us that you have read it critically, going beyond a Generative AI summary. (2 points)

Your answer here.

(5) Did you use ChatGPT or any Generative AI tool on this exam? Do such tools help with this sort of exam? Should we allow ChatGPT on Exam #2? (Remember, free ChatGPT is allowed, so you're not cheating. This is to help us improve the course, not to get you in trouble.) (2 points)

Your answer here.

Honor Pledge and Exam Submission

You must check the boxes below before you can submit your exam.

☐ I have neither given nor received unauthorized aid on this exam.

☐ *I am ready to submit my exam.*

Submit My Exam

Once you submit, you will be able to leave the page without issue. Please don't try to mash the button.

The exam is graded out of 100 points.

minutes remaining

Hide Time

Manual Save

Navigation