LATE

minutes remaining

Hide Time

Manual Save

Navigation

- Question 1
- Question 2
- Question 3
- Question 4
- Question 5
- Question 6
- Question 7
- Extra Credit
- Pledge & Submit

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

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

A. — Creational Design Pattern	B. — Delta Debugging	C. — Elicitation	D. — Fault Localization
E. — Informal Goal	F. — Maintainability	G. — Medical Imaging	H. — Priority
I. — Productivity	J. — Profiling	K. — Program Synthesis	L. — Quantum Computing
M. — Requirements	N. — Risk	O. — Stakeholder	P. — Static Analysis
Q. — Structural Design Pattern	R. — Validation	S. — Watchpoint	T. — Weak Conflict

Q1.1: You are called in by metamorphic rock shop company AllSlate to create a website for them. They tell you the things that the website needs: a feature to search through their rock catalog, a feature to add rocks to a user's cart, and the ability to purchase the rocks in their cart securely.

Q1.2: This includes both the odds of an event happening and also the consequences of that event happening.

Q1.3: Internally, developers find a defect related to the security of customer records that could have significant consequences if exploited. However, CreditCo management is worried about market reactions and decides not to publicly pursue creating and deploying a patch in the short term.

Q1.4: Miyamoto uses a series of interviews, walkthroughs and checklists to ensure that the requirements are complete and consistent.

Q1.5: Maya is facing a looming deadline and is tasked with identifying methods that are likely to be slow. Rather than running the program and recording execution times, she writes a script to count the number of loops in each method's source code and uses that to estimate how long it will take that method to run.

Q1.6: Decades ago, software engineering work was mistakenly believed to be partitionable. In practice, adding more people to a late project tends to make it later, informing our ability to plan with respect to *this concept*.

Q1.7: ToSoftware is working on a difficult game. Developers are instructed to be sure that "no one can complete the game in under 5 hours" but also that "speedrunners can complete the game in times that can make high-revenue YouTube videos".

Q1.8: Gabriel is writing a program that has access to video files and he wants to use an analysis library that operates on sets of images. He writes a wrapper extracts the frames from the videos as individual images and invokes the library on those individual images.

Q1.9: Yang is responsible for maintaining a circularly-linked list. Unfortunately, on some long runs the outgoing pointer from the first element is replaced with an incorrect value. Yang runs the program and arranges for execution to pause when that pointer value is changed.

Q1.10: Golda is writing software for a microphone company. The company's market research suggests that customers want the audio to "not sound too breathy". This information, without elaboration, is provided to Golda to help guide software development.

Q1.11: In an approach conceptually similar to machine learning, Pedro carries out a task manually and then uses a formal grammar to automatically create code that does that same task.

Q1.12: Li Bai structures his class so that the normal constructor cannot be called and another method must be called instead. This allows him to hide type information.

Q1.13: The University of Michigan is considering revamping Wolverine Access. Lawyers, representatives from human resources, professors, and students are all gathered to provide input.

Q1.14: Wei's manager mistakenly believes that this technology will definitely allow the hardest problems in computing to be solved in polynomial time.

LATE

minutes remaining

Hide Time

Manual Save

Navigation

- Question 1
- Question 2
- Question 3
- Question 4
- Question 5
- Question 6
- Question 7
- Extra Credit
- Pledge & Submit

Question 2. Delta Debugging (20 points)

We want to design a new fault localization algorithm, Tetra Debugging, which functions very similar to delta debugging but hopefully converges on an answer more quickly. Rather than splitting the test set in half, Tetra Debugging partitions the test set into fourths and tests if each subset is interesting. A brief implementation of parts of the `tetra` algorithm is provided below.

```
1  def split(lst):
2      pivot = math.ceil(len(lst) / 4)
3      return lst[:pivot], lst[pivot:pivot*2], lst[pivot*2:pivot*3], lst[pivot*3:]
4
5  # def minimize_lt_four(lst):
6      # Returns a list of the minimal interesting subset of `lst`
7      # Assumes that lst is interesting and len(lst) < 4
8      # Makes exactly one call to Interesting() for each element of `lst`
9      # The code is not shown, but you can assume it works correctly
10
11 # def union(*lsts):
12     # Returns the union of the lists passed in as arguments
13     # The code is not shown, but you can assume it works correctly
14
15 def tetra(P, C):
16     if len(C) < 4:
17         return minimize_lt_four(C)
18     p1, p2, p3, p4 = split(C)
19     if (interesting(p1)): return tetra(P, p1)
20     if (interesting(p2)): return tetra(P, p2)
21     if (interesting(p3)): return tetra(P, p3)
22     if (interesting(p4)): return tetra(P, p4)
23
24     result = union(
25       tetra(union(P, p2, p3, p4), p1),
26       tetra(union(P, p1, p3, p4), p2),
27       tetra(union(P, p1, p2, p4), p3),
28       tetra(union(P, p1, p2, p3), p4))
29
30     return result
31
```

(a) (4 points) Pick the code snippet that, when put in place of lines 24-28, makes `tetra` yield a correct minimal subset. If there are multiple correct answers, pick any one of them. (Assume there are no syntax errors or similar concerns: this is a question about algorithm logic, not Python details.)

○
```
result = union(
    tetra(union(P, p1, p2, p3, p4), p1),
    tetra(union(P, p1, p2, p3, p4), p2),
    tetra(union(P, p1, p2, p3, p4), p3),
    tetra(union(P, p1, p2, p3, p4), p4)
)

```

○
```
result = union(
    tetra(union(P, p3, p4), union(p1, p2)),
    tetra(union(P, p1, p2), union(p3, p4))
)

```

LATE

minutes remaining

Hide Time

Manual Save

Navigation

- Question 1
- Question 2
- Question 3
- Question 4
- Question 5
- Question 6
- Question 7
- Extra Credit
- Pledge & Submit


```
result = union(
    tetra(union(P, p2, p4), p1),
    tetra(union(P, p2, p4), p3),
    tetra(union(P, p1, p3), p2),
    tetra(union(P, p1, p3), p4)
)
```

```
result = union(
    tetra(P, p1),
    tetra(P, p2),
    tetra(P, p3),
    tetra(P, p4)
)
```

```
result = union(
    tetra(union(P, p3, p4), p1),
    tetra(union(P, p3, p4), p2),
    tetra(union(P, p1, p2), p3),
    tetra(union(P, p1, p2), p4)
)
```

(b) (2 points each, 6 points) Consider an `Interesting` function that returns true if a list contains the elements `2` and `7`. When **Tetra Debugging** is run on the following inputs, how many probes to `Interesting` are made to obtain the minimal interesting subset? (When answering, assume Tetra works according to its specification, regardless of your answer above.) If the program does not terminate, returns a subset that is not interesting, or returns a subset that is not minimal, answer "INVALID".

(i) (2 points) `P = [], C = [2, 0, 4, 6, 3, 5, 1, 7]`

Your answer here.

(ii) (2 points) `P = [], C = [1, 6, 3, 7, 2, 4, 5, 0]`

Your answer here.

(iii) (2 points) `P = [], C = [2, 7, 6, 1, 5, 4, 0, 3]`

Your answer here.

(c) (2 points each, 4 points) Assuming constant time per `Interesting()` check:

(ci) (2 points) What is the worst-case Big-Oh asymptotic time complexity of Tetra with respect to the size **n** of the initial input changeset if a single change induces the failure? (In other words: using Big-Oh notation, how many calls does Tetra make to `Interesting()` if there is one single-element list that is interesting?)

Your answer here.

(cii) (2 points) What is the worst-case Big-Oh asymptotic time complexity of Tetra with respect to the size **n** of the initial input changeset?

LATE

minutes remaining

Hide Time

Manual Save

Navigation

- Question 1
- Question 2
- Question 3
- Question 4
- Question 5
- Question 6
- Question 7
- Extra Credit
- Pledge & Submit

Your answer here.

(d) (1 point each, 6 points) Consider the following situations. For each situation, indicate whether **Delta Debugging** will obtain a minimal interesting subset and whether **Tetra Debugging** will obtain a minimal interesting subset.

Situation #1: The day before project 5 was due, your teammate committed 8 changes to your project repository, collectively causing your code to fail all of your test cases. That night, they pushed 8 more changes, one of which fixed the bug and made the code pass all of the test cases. From the full set of 16 changes your partner committed, you want to find the original change that caused the test cases to fail.

(i) (1 points) **True / False**: Delta debugging can obtain a minimal interesting subset for situation #1.

- ◯ True
- ◯ False

(ii) (1 points) **True / False**: Tetra debugging can obtain a minimal interesting subset for situation #1.

- ◯ True
- ◯ False

Situation #2: We have an HTML tag `<div class="widget" role="navigatino" aria-lable="Outline" tabindex="0" style="padding-right: 16px;">` that fails to parse with our HTML parser. We want to find which tag attribute may be causing the tag to fail to parse.

(iii) (1 points) **True / False**: Delta debugging can obtain a minimal interesting subset for situation #2.

- ◯ True
- ◯ False

(iv) (1 points) **True / False**: Tetra debugging can obtain a minimal interesting subset for situation #2.

- ◯ True
- ◯ False

Situation #3: We are putting together a software development team, and among our whole team, we need at least one developer who knows each of the languages Python, C++, and C#. We have the following list of candidates: { Aidan: [Python, C#], Cameron: [Python], Conner: [C#], Daniel: [C++], Holly: [C++, C#], Jason: [Python, C++, C#]} and want to find the smallest team we can assemble that together knows all of the languages.

(v) (1 points) **True / False**: Delta debugging can obtain a minimal interesting subset for situation #3.

- ◯ True
- ◯ False

(vi) (1 points) **True / False**: Tetra debugging can obtain a minimal interesting subset for situation #3.

- ◯ True
- ◯ False

Question 3. Short Answer (4 points each, 20 points)

(a) (4 points) Lizzy is making preparations to open her new hot air balloon resale shop on State Street called *Maize and Blue'ns*. Her store will sell locally to Michigan but also online to Ohio. The local and online sales have different taxes and policies. In 3 sentences or fewer, describe which design pattern you would use and why, and one risk associated with that design pattern.

Your answer here.

(b) (4 points) Cassie is adding a new feature to decrease wait times at her carnival attraction Euphoric Carousel. Describe four steps or activities she might follow for effective requirement elicitation. Use 4 sentences or fewer.

Your answer here.

(c) (4 points) Describe 2 considerations Ray Buse mentioned during his lecture regarding how Google automates the testing of phone applications. For each consideration, describe the problem and then describe the impact or solution. Use a total of 4 sentences or fewer.

Your answer here.

(d) (4 points) List two similarities between Program Synthesis and Pair Programming. Then list two differences. Strong answers should include a focus on the inputs and outcomes of the two tools — on when they can be used and what benefits they provide. For example, do not simply say that one involves a second human and the other does not. Instead, compare and contrast them in terms of their potential use in a software development process. Use at most 4 sentences.

Your answer here.

(e) (4 points) Describe, in your own words, a project that might require multiple languages. Then, provide one advantage and one disadvantage of multi-language projects. Use at most 4 sentences.

Your answer here.

Question 4. Fault Localization (10 points)

You are hired to write a program that generates random questions for a midterm exam. For each input (the name of a user taking the exam) the program should generate slightly different exam text as its output. You write a long Python program to do this, but unfortunately the output of your program is sometimes incorrect. In other words, for some inputs your program produces the correct output but for some other inputs your program produces incorrect output.

```
1   def quality_question():
2     random.seed(int(username))
3     username = argv[2]
...
17    prompts = [alternate_1, alternate_2, alternate_3]
...
37    # Randomly sample two prompts
38    a_prompt, b_prompt = random.sample(prompts, 2)
...
89    c_prompt = generate_prompt()
90    # If the randomly generated prompt is already chosen, regenerate it
91    if c_prompt in [a_prompt, b_prompt]:
92      c_prompt = generate_prompt()
...
150 def name_scrambler():
...
```

You decide to use fault localization to pin down where the issues in your code are.

(a) (4 points) You consider two automated approaches to fault localization: an approach in which Delta Debugging is used to find a minimal set of suspicious lines, and an approach in which Tarantula is used to rank suspicious lines. (The details of these two potential approaches are intentionally not provided. You must think of possibilities based on course concepts.) Which fault localization method would work better here: Delta Debugging or Tarantula? Why? (Use at most four sentences.)

Your answer here.

LATE

minutes remaining

Hide Time

Manual Save

Navigation

- Question 1
- Question 2
- Question 3
- Question 4
- Question 5
- Question 6
- Question 7
- Extra Credit
- Pledge & Submit

LATE

minutes remaining

Hide Time

Manual Save

Navigation

- Question 1
- Question 2
- Question 3
- Question 4
- Question 5
- Question 6
- Question 7
- Extra Credit
- Pledge & Submit

(b) (4 points) Consider the following table of program runs. Each row corresponds to one test case execution. In other words, each row reports one run of your program in which it takes a username as input and produces output that is either correct or incorrect. Each row also includes the lines visited while your program executes on that input.

Test Input	Passed?	Lines visited
bakalm	True	[1, 2, 3, 17, 37, 38, 89, 90, 91, 150, 151]
weimerw	True	[1, 2, 3, 17, 37, 38, 89, 90, 91, 92, 150, 151]
hstauff	False	[1, 2, 3, 17, 37, 38, 89, 90, 91, 92, 150]
chein	False	[1, 17, 37, 38, 89, 90, 91, 92, 150]
haasea	True	[1, 2, 3, 17, 37, 38, 89, 90, 91, 150]

Compute the **Tarantula** suspiciousness score for each line in the program and provide the suspiciousness scores for the top 3 most suspicious lines of code. Express the final answer as a list of tuples of line numbers (ints) and scores (floats, 3 significant figures), sorted by score descending and then by line number ascending; if line 2 has a suspiciousness of 0.667, line 1 has a suspiciousness of 0.5, and line 5 has a suspiciousness of 0.5, your answer should be `[(2, 0.667), (1, 0.500), (5, 0.500)]`.

Your answer here.

(c) (2 points) Which line from your list do you think is causing the problem and why? (If you think your list does not contain a relevant line, indicate that instead and explain why.)

Your answer here.

Question 5. Profiling (7 points)

Consider the function call profile below. The `main` function is run with an *event-based profiler* and the following hierarchical profile is generated:

```
1 * main()
2     2 * can()
3         2 * head()
4         3 * kay()
5         4 * ever()
6     1 * direction()
7         5 * head()
8         5 * kay()
9         5 * ever()
10
```

In addition to the call graph above, the following non-cumulative, per-one-single-call *self time* information is also generated:

```
main() - 100ms
can() - 250ms
direction() - 720ms
head() - 50ms
kay() - 100ms
ever() - 40ms

```

Note, the call graph specifies how many times a function is called in its individual context. For example, each time can is called, head will run multiple times.

(a) (5 points) In this problem, the self-time of a function is the time the function's stack frame spends on the top of the stack

for one single call to that function. (Equivalently, the self-time of a function is the time taken by one single run of that function *not* including the time taken by any children or parent functions.)

`main` is run with a statistical profiler. Assuming that the self-times remain constant across runs, write the names of each function in descending order of probability that a random probe of the profiler would interrupt the program in that function. If there is a tie, list the function with the alphabetically-earlier name first (e.g., so if you believe "ant" and "bat" are equally likely to be running when the program is sampled, list "ant" before "bat"). Your answer must be a Python-formatted list. For example, if you believe that the order should be (most probable) A, B, C, D (least probable), you should answer: `["A", "B", "C", "D"]`. Your list should contain 6 elements.

Your answer here.

(b) (2 points) Support or refute the claim that a call-graph execution profiler like `gprof` would produce actionable insights (i.e., useful information) for a project that includes both Java and C code. Use two pieces of evidence to support or refute the claim. Use at most 4 sentences.

Your answer here.

Question 6. Interviews (14 points)

As an interviewer, you give the following technical challenge to a potential candidate: "Write `isPalindrome()`, a function that returns `true` if parameter `x` is a palindrome integer. Note that an integer, like 12321, is a palindrome if it reads the same forwards and backwards."

The candidate's implementation of `isPalindrome()` is below along with two questions the candidate asked you:

```
// Q: Can integer x be in range [0-9]? A: Yes.
// Q: Should I account for integer overflow? A: Yes.

bool isPalindrome(int x) {
    // a single digit is a palindrome
    if (x < 10) {
        return true;
    }

  // if x's last digit is 0, then its first digit must be 0 in order
  // to be a palindrome. In this case, only 0 can be a palindrome.
  if (x % 10 == 0 && x != 0) {
    return false;
  }

  // revert the last half of x for comparison against first half
  int revertedNumber = 0;
  while(x > revertedNumber) {
    revertedNumber = revertedNumber * 10 + x % 10;
    x /= 10;
  }

  return x == revertedNumber || x == revertedNumber / 10;
}

```

(a) (2 points) Identify two test cases where the provided `isPalindrome()` would return `false`.

Your answer here.

(b) (4 points) Identify four things that the candidate did well. (In other words, identify four properties that a company might desire in a software engineer that could potentially be shown by a candidate taking the interview and that were shown by this particular candidate.) Use at most 4 sentences.

LATE

minutes remaining

Hide Time

Manual Save

Navigation

- Question 1
- Question 2
- Question 3
- Question 4
- Question 5
- Question 6
- Question 7
- Extra Credit
- Pledge & Submit

LATE

minutes remaining

Hide Time

Manual Save

Navigation

- Question 1
- Question 2
- Question 3
- Question 4
- Question 5
- Question 6
- Question 7
- Extra Credit
- Pledge & Submit

Your answer here.

(c) (4 points) Support or refute the claim that the candidate's implementation of `isPalindrome()` is functionally correct. Use at most 4 sentences.

Your answer here.

(d) (2 points) Describe a hypothetical defect in `isPalindrome()` that Automated Program Repair would likely be able to fix and explain why. Then describe a hypothetical defect in `isPalindrome()` that APR would be unlikely to fix and explain why. Use at most 4 sentences.

Your answer here.

(e) (2 points) Support or refute the claim that programmer productivity is essentially fixed (i.e., individual performance differences are predominantly due to inborn talents rather than learned skills). Reference at least two results from scientific literature (e.g., from Computer Science or Psychology, etc.) as evidence as you support or refute the claim. Use at most 4 sentences.

Your answer here.

Question 7. Requirements Elicitation (15 points)

(a) (6 points) Suppose that the university is in the beginning stages of developing a streaming service, specifically for students to watch University of Michigan related videos (sports, news segments, etc.) called BlueTV. BlueTV should have a feature in which it recommends new relevant or related videos to watchers. Name 3 *quality requirements* that the team might have for the video recommender aspect of BlueTV, noting if each is verifiable or informal. Include at least one verifiable requirement and one informal requirement in your answer. Use 3-6 sentences if possible.

Your answer here.

(b) (6 points) Identify 2 different stakeholders for BlueTV, and describe two different *conflicts* that might arise between the two identified stakeholders during the requirements elicitation process. For each conflict, include a formal keyword to describe the conflict if possible (e.g., what type or name of conflict is it?), and then describe the *strength* of the conflict (if possible). Use at most 6 sentences.

Your answer here.

(c) (3 points) Choose one of the conflicts you identified previously. What would be the best way to resolve this conflict? Use at most 3 sentences.

Your answer here.

LATE

minutes remaining

Hide Time

Manual Save

Navigation

- Question 1
- Question 2
- Question 3
- Question 4
- Question 5
- Question 6
- Question 7
- Extra Credit
- Pledge & Submit

Question 8. Extra Credit (1 point each)

(*Feedback*) What was your favorite topic or activity during the course?

What is one thing you like about this class?

(*Feedback*) What do you think we should do more of next semester (or what is the thing you would most recommend that we change for future semesters)?

What is one thing you dislike about this class?

(*Guest Lecture*) List one thing you learned from guest speaker Peter K. Shultz of Microsoft or otherwise convince us that you paid careful attention during that lecture. Your answer must be distinct from any references to the Shultz lecture you may have made earlier in the exam.

Shultz guest lecture.

(*Optional Reading 1*) Identify a single optional reading that was assigned after Exam 1. Write two sentences about it that convince us you read it critically. (The most common student mistakes for these questions in Exam 1 were choosing a required reading instead of an optional reading or failing to "identify" or name the reading selected.)

Optional Reading 1

(*Optional Reading / Piazza 2*) Identify a different single optional reading that was assigned after Exam 1 or a *"long instructor post"* that was posted on Piazza after Exam 1. Write two sentences about it that convince us you read it critically.

Optional Reading 2

(*Guest Lecture*) List one thing you learned from guest speaker Dr. Ray Buse of Google that was not listed on an introductory summary slide or otherwise convince us that you paid careful attention during that lecture. Your answer must be distinct from any references to the Buse lecture you may have made earlier in the exam.

Buse guest lecture

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.*

Note that your submission will be marked as late. You can still submit, and we will retain all submissions you make, but unless you have a documented extenuating circumstance, we will not consider this submission.

Submit My Exam

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