CSE 8A/6R Fall 2025 Homework 1

Due: Thursday, October 9, 2025 before 11:59pm

Full Name: _____ PID: _____

Here's a recap about the general logic behind the homework component of CSE8A/6R:
The homework assignments will test your knowledge and understanding of programming across the ***5 core competencies***:
1. Code Tracing, 2. Code Comprehension, 3. Code Writing, 4. Debugging, and 5. Testing

Homework assignments will be released on Fridays and will be due the following Thursdays. Assignments will be distributed as worksheets and submitted via **Gradescope**. While you must submit your own work, collaboration and discussion are highly encouraged and recommended!

A significant portion of homework will be **handwritten**, either on paper or a digital tablet, and **manually graded** by staff. This format is intentional:

- Some tasks require diagrams or graphs that are difficult to type.
- Writing (including writing small blocks of code) encourages deeper and better engagement with the material.
- Homework assignments will focus on the **process** of how you got to your answer (i.e., showing your work or explaining your reasoning) rather than exclusively on the correctness of your answer. Sharing your reasoning process helps demonstrate understanding beyond the final answer.
- The format mirrors the handwritten exams, preparing you for similar constraints under timed conditions.

Homework 1 Topics Covered:
1. Variables, 2. Data types, 3. Arithmetic operations, 4. Functions

Homework 1 Submission Requirements:
*All questions up to question 6 must be **handwritten** on paper or tablets. You are NOT allowed to type any of your answers for this homework.* Your solutions should be put in the boxes provided, and the pages should be scanned/uploaded to Gradescope. We cannot grade solutions outside the boxes or on a separate sheet of paper.

If you need help printing Homeworks, please contact the instructor or TAs during our office hours and we will be happy to help print a copy of the homework for you! :)

Office hours schedule: https://cse8a.site/calendar

1. [Code Tracing] What is the output (i.e., what will be printed) when the following code is executed?

```
num_tas = 0
num_tutors = 0
num_instructors = 0
total_staff = num_tas + num_tutors + num_instructors
num_tas = 5
num_tutors = 21
num_instructors = 1
print("Total staff:", total_staff)
```

A) Show your work (we have done a similar problem in Lecture 1 Part 1G):

B) Output:

2. [Code Tracing] The following code snippet on the left produces the output on the right. Answer the questions that follow.

```
print("line A")
def rect_area(length, width):
    print("line B")
    area = length * width
    print("line C")
    return area
    print("line D")

print("line E")
area_1 = rect_area(5, 10)
print("line F")
print("area_1:", area_1)
print("line G")
```

Output:

```
line A
line E
line B
line C
line F
area_1: 50
line G
```

A) Why is line B executed after line E?

B) Why is line D not printed?

3. [Code Comprehension] Read the following code and answer the questions that follow.

```
def calculate_total_cost(item_price, quantity, tax_rate):
    subtotal = item_price * quantity
    total_cost = subtotal + (subtotal * tax_rate)
    return total_cost

price_per_item = 20
number_of_items = 3
tax = 0.07
final_cost = calculate_total_cost(price_per_item, number_of_items, tax)
print("Final cost:", final_cost)
```

A) In the above code, how many parameter(s) does the function `calculate_total_cost()` have, and what is/are the name(s) of the parameter(s)?

Number of parameter(s):

Name(s) of the parameter(s):

B) How many variables are defined outside the function `calculate_total_cost()`, and what are the name(s) and data types(s) of those variable(s)?

Number of variables defined *outside* the function `calculate_total_cost()`:

Name(s) and data type(s) of those variable(s):

C) What are the *arguments* that are passed to the function `calculate_total_cost()`?

4. [Code Comprehension] In *Explain in plain English* questions, you should not explain the code line-by-line. Instead, you should focus on the high-level functionality of the code, imagining yourself explaining the code to someone with no background in programming.

For example, **consider the following function:**

```
def greet(name):
    return "Hello, " + name + "!"
```

Explanation: This function takes a person's name and creates a greeting message using that name.

Now it's your turn! Explain in plain English the purpose of the following function.

In the question below, we intentionally use an ambiguous function name (i.e., mystery) and generic variable names (i.e., a, b) to encourage you to understand the underlying logic and purpose of the code itself, rather than relying on descriptive names to find what it is doing.

Hint: this function should reflect the mathematical formula for calculating a property of a geometric shape.

```
def mystery(a, b):
    result = 3.14 * a * a * b
    return result
```

A) Explanation:

5. [Code Writing, Testing] Unit conversion is a concept you learned in early math classes. Your goal for this question is to write a function that does some kind of unit conversion (e.g., km to miles, but you should pick something else). [we will do a similar problem in Lecture 3 on 10/7]:

A) What unit conversion do you want to implement?

I want to convert _____ to _____.

The mathematical formula for this conversion is:

List two examples of this conversion below using the formula:

B) Identify the properties of your function.

Name of the function:

Parameter(s):

Data type of return value:

C) Time to actually *define* your function! You can write the function in VS Code and test it following part D instructions to ensure it works as expected. Once you have your final code, please hand-write your code below.

D) Now let's test the results by *calling* your function and *printing* the returned values, using the two examples from part A. Again, you may call your function and test your output in your IDE (VS Code) before writing down your answer.

For each test, you must provide three things: the Test code, Expected Output, and Actual Output. Your tests should follow this specific format:
1. **The Test:** This should be the exact line of Python code you would use to run your function and see the result. You must use the print() function to display the output.
2. **The Expected Output:** This should be the exact value you expect the function to return when you run your test code. You should calculate this value on your own without using your function.
3. **The Actual Output:** This is the actual value that your function returns when you run your test.

Let's imagine you wrote a function called **km_to_miles()** that converts kilometers to miles. Here is a reasonable formatting of the tests. The output results might vary depending on your own choice of how to manage decimal places.

Test #1: `print ( km_to_miles ( 10))`

Expected output: 6.214

Actual output: 6.214

Below, write two tests for your function using the examples you came up with in Part A.

Test #1:

Expected output:

Actual output:

Test #2:

Expected output:

Actual output:

6. [Debugging] Consider the code on the left, along with the expected and actual outputs on the right.

```
def multiply(a, b): # line 1
    result = a * b # line 2
    print(result) # line 3

answer = multiply(3, 4)
print("The answer is:", answer)
```

Expected output:

```
The answer is: 12
```

Actual output:

```
12
The answer is: None
```

A) Why is the actual output different from the expected output? Why are we seeing `None` printed as the `answer`?

B) How could you change *one line of code* within the function such that the actual output matches the expected output?

Line to change:

Change code to:

Submission

On Gradescope, you should submit your **handwritten** solution for Questions 1-6 under **Homework 1** as a **pdf document**.

Instructions to submission (also under Canvas - Homework 1):
1) Using the gradescope website on your computer:
https://guides.gradescope.com/hc/en-us/articles/21864315441677-Submitting-a-PDF-for-an-assignmentLinks.

2) Using the gradescope mobile app:
https://guides.gradescope.com/hc/en-us/articles/22016028459789-Using-the-Gradescope-Mobile-App-for-StudentsLinks.

Within each assignment, individual problems will be graded on a **binary** scale:

- **Pass:** The process (e.g., work shown, explanation given, process journal) is described satisfactorily and the end product (e.g., final answer, program written) is correct.
- **Not yet:** Either the process is not shown or incomplete and/or the end product is incorrect.

Point breakdown for Homework 1:

- **Exemplary (3 homework points):** All problems receive a "Pass."
- **Satisfactory (2 homework points):** At least 4 out of 6 questions receive a "Pass."
- **Needs Improvement (1 homework point):** At least 1 out of 6 questions receive a "Pass."
- **Incomplete (0 homework points):** No problems receive a "Pass," or the assignment was not submitted, or the assignment in ungradable (e.g., illegible handwriting), or if the submission does not follow the instructions (e.g., homework is typed instead of handwritten)

Your work will be marked **Ungradable** if your submission cannot be assessed because of issues that prevent interpretation of the work. This may include, but is not limited to, **illegible handwriting**, a corrupted or unreadable file, missing pages or sections, or work that is completely off-topic or irrelevant to the assignment.