Full Name: _____

Email Address (UW Net ID): _____ @uw.edu

Section: _____

CSE 160 Winter 2025 - Midterm Exam

Instructions:

- You have **the entire class period** to complete this exam.
- The exam is **closed book**, including no calculators, computers, phones, watches or other electronics.
- You are allowed a single sheet of notes for yourself.
- We also provide a syntax reference sheet.
- Turn in ***all sheets*** of this exam, together and in the same order when you are finished.
- When time has been called, you must put down your pencil and stop writing.
    - **Points will be deducted if you are still writing after time has been called.**
- You may only use parts and features of Python that have been covered in class up to this point.
- You may ask questions by raising your hand, and a TA will come over to you.

Good luck!

Question	**Topic**	**Points**
Question 1	Expressions	10
Question 2	File I/O	10
Question 3	Nested Lists	8
Question 4	Dictionaries	8
Question 5	Functions	14

1 (10 pts). Given the table below, fill in the correct values and type for the matching expression. In other words, what will be outputted if this code is run in the Python interpreter. If there is an error, write "Error" in the value column. (You may leave the type column blank, and you do *not* have to explain the error.)

Expression	**Value**	**Type**
`len([1, 2, 3]) * 5 // 3`		
`"10" + "2" * 3`		
`5 % 2 >= 5 / 2`		
`list = [1, 2, 3, 4, 5]` `list[2:]`		
`d = {1: "A", 2: "B", 3: "C"}` `d["C"] * 2`		

2 (10 pts). Consider you are given a file named `transcript.txt`: The contents of the file are shown below:

`transcript.txt`

```
CSE,160,A
MATH,124,B
CSE,163,A+
STAT,110,A
STAT,111,C
```

You are also given the following function `get_grades(filepath, dept)` that takes in the name of a file and a department code (e.g. “CSE”, “STAT”, etc.)

```
def get_grades(filepath, dept):
    myfile = open(filepath, "r")
    list = []
    for line in myfile:
        tokens = line.strip().split(",")
        if(tokens[0] == dept):
            list.append(tokens[len(tokens) - 1])
    return list
```

For each of the function calls below, what will the function return? If you think there will be an error, write “Error”.

```
get_grades("transcript.txt")
```
_____

```
get_grades("transcript.txt", "CSE")
```
_____

```
get_grades("transcript.txt", "STAT")
```
_____

```
get_grades("transcript.txt", "Math")
```
_____

```
get_grades("transcript.txt", "CHEM")
```
_____

3 (8 pts). For this problem, consider the nested list below:

```
numbers = [ [1, 2, 3, 4] , [5, 6, 7, 8],
            [9, 10, 11, 12], [13, 14, 15, 16] ]
```

For this problem, write code that creates two lists. One of these lists should contain all the even elements found in the nested list. The second list should contain all of the odd elements found in the nested list.

You do not need to return or print the resulting lists, just create them.

```
# Write your code in this box, you can assume the nested list has
already been declared
```

4 (8 pts). Consider the following dictionary:

```
employees = {

"A1": {"name": "John", "department": "HR", "salary": 50000},
"A2": {"name": "Emily", "department": "Finance", "salary": 60000},
"A3": {"name": "David", "department": "IT", "salary": 70000},
"A4": {"name": "Jose", "department": "Marketing", "salary": 55000},
"A5": {"name": "Michael", "department": "Sales", "salary": 65000},

}
```

What will each expression evaluate to? Write "Error" if the statement causes an error.

Code	**Value**
`employees["A5"]["salary"]`	
`len(employees[0])`	
`employees["A2"]["department"]`	
`employees["A5"][1]`	

5 (14 pts). For this problem, you will be given a list of strings and asked to print out the consonant:vowel ratio for each of them using two functions: `consonant_vowel_count`, and `calculate_ratio`

A (6 pts). Implement a function called `calculate_ratio` which takes in a dictionary containing the numbers of consonants and vowels and returns a float which represents the number of consonants divided by the number of vowels. Assume that there is at least one vowel in the input dictionary, and the only keys are `"consonant"` and `"vowel"`.

Ex. `calculate_ratio({"consonant": 5, "vowel": 1})` will return `5.0`

```
def caclulate_ratio(counts_dictionary):
     # Write your code below
```

B (8 pts). For this part of the problem, assume that `consonant_vowel_count` is already implemented for you. A doc-string and example usage is provided below:

```
def consonant_vowel_count(word):
     """Given a one-word string, returns a dictionary mapping the
     number of vowels and consonants in the word"""
```

Ex. `consonant_vowel_count("example")` will return `{"consonant": 4, "vowel": 3}`

Using calls to `consonant_vowel_count` and `calculate_ratio`, write code that when given `word_list` will print out the consonant:vowel ratio of each word on separate lines. A for loop must be used for full credit.

Should print out:

```
1.3333333333333333
1.0
2.5
1.0
0.375
```

```
word_list = ["welcome", "to", "intro", "data", "programming]
# Write your code below
```

Extra Credit (1 point): Did you fill out a mid quarter feedback form? You will get this extra credit point if you do!