Name: ______________________________

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

Section: ______________________________

CSE 160 Spring 2023 - Midterm Exam

Instructions:

- You have **40 minutes** 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.
- All questions assume Python version 3.7, as we have been using all quarter.
- You may ask questions by raising your hand, and a TA will come over to you.

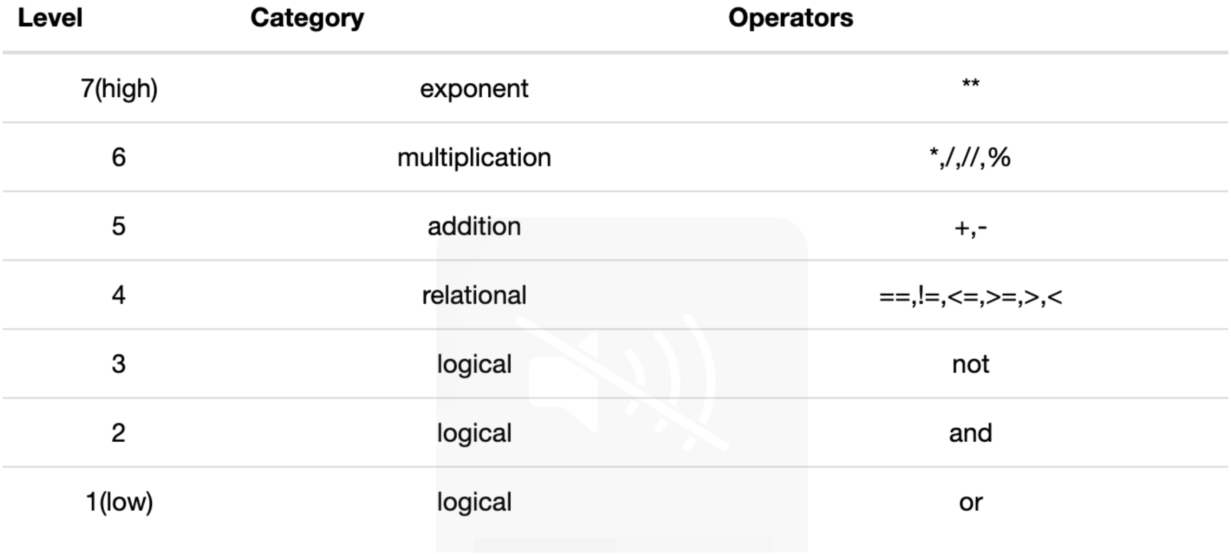
Good luck!

Question	**Points**
Question 1	
Question 2	
Question 3	
Question 4	
Question 5	
TOTAL	

1. **[6 pts] For each of the below expressions, write what the expression evaluates to and the type of that value.**

Expression	**Evaluation**	**Type**
11 / 2	5.5	float
2 + 11	13	int
11 % 2 == 0 or not False	True	bool
not(len(‘hello’) < 5 and 10.5 > 2)	True	bool
‘hello’[1] + str(3 * 7.0)	“e21.0”	str

a = 10
b = ‘hello’
{"a": 1, 1: "a", "b": b}[‘b’][2] + f[1]
	“la”	str



Level	**Category**	**Operators**
7(high)	exponent	**
6	multiplication	*,/,//,%
5	addition	+,-
4	relational	==,!=,<=,>=,>,<
3	logical	not
2	logical	and
1(low)	logical	or

2. Write a function `tricky_function(arr)` that modifies a given `list` in the following way: at index `i`, if the value at index `i` is divisible by 3, then the function should multiply the value at index `i` by its index; otherwise leave the value as it is.

   Examples:

   ```
   tricky_function ([0, 1, 2, 3, 4, 5, 6, 7, 8]) = [0, 1, 2, 9, 4, 5, 36, 7, 8]
   tricky_function ([2, 1, 21, 7, 4, 5, 4, 7, 18]) = [2, 1, 21, 21, 4, 5, 24, 7, 18]
   tricky_function ([2, 3, 2]) = [2, 3, 2]
   tricky_function ([1, 1, 1, 1]) = [1, 1, 1, 1]
   ```

   Possible/Example Answer(s):

   ```
   def tricky_function(numbers: list[int]) -> numbers: list[int]
        for i in range(len(numbers)):
             if i % 3 == 0:
                  numbers[i] = numbers[i] * i
        return numbers
   ```

3. Write a function `join(list1, list2)` that, given two lists of characters `list1` and `list2`, returns the number of pairs in `list1` and `list2` whose values are equal.

   Examples:

   ```
   join ([a,b,c,e], [a,a,e,b,c,c,d]) = 6
   *[[a,a], [a,a], [b,b], [c,c], [c,c,], [e,e,]]

   join ([a,a], [a,a]) = 4
   *[[a,a], [a,a], [a,a], [a,a]]

   join ([a,b,c], [a,b,c]) = 3
   *[[a,a], [a,a], [b,b], [c,c]]

   join ([x,y,z], [a,a,e,b,c,c,d]) = 0
   *[]
   ```

   Possible/Example Answer(s):

   ```
   def join(list1: list[char], list2: list[char]) -> numbers: int
         count = 0
         for i in range(len(list1)):
               for j in range(len(list2)):
                     if list1[i] == list2[i]:
                           count += 1
         return count
   ```

4. After running this snippet of code, what will this function print?

```
def foo(n1, n2, n3, b, s, d):
    if n1 % 2 == 0 or not b:
        return "A"
    elif str(n2 * n3) == 21.0:
        return "B"
    if d["b"][2] + s[1] == la":
        return "C"
    if d["a"] == n1 - n2:
        return "D"
    else:
        return "E"

a = 11;
b = "hello";
c = False;
d = 10.5;
e = 2;
f = { "a": 1, 1: "a", "b": b }

result1 = foo(a, d, e, c, b, f)
result2 = foo(17, 21, 1, true, "hallo", {"a": 1, 1: "a", "b":
"hello"})
result3 = foo(0, 3, 7.0, false, "hallo",{"a": -3, 1: "a", "b":
"hallo"})

print(result1)
print(result2)
print(result3)
```

Example Answer(s):

```
AE
CE
ACD
```

5. Write a function `max_consecutive_sum` that, given a list of integers, returns the absolute value of the smallest sum between two consecutive numbers in the list. Return -1 if the length of the list is less than or equal to 1.

   Examples:

   ```
   max_consecutive_sum ([36, 45, 7]) = 81
   max_consecutive_sum ([3]) = -1
   max_consecutive_sum ([2, 3, 2]) = 5
   max_consecutive_sum ([1, 1, 1, 1]) = 2
   ```

   Possible/Example Answer(s):

   ```
   def max_consecutive_sum(numbers: list[int]) -> int:
         max = 0
         for i in range(0, len(numbers) - 1):
               sum = numbers[i] + numbers[i + 1]
               if sum > max:
                     max = sum
         return max
   ```