COMPUTING ID ________________________ **PRINT NAME LEGIBLY:** ____

Read this entire page. You are responsible knowing what it says.

Honor

- By submitting solutions for this test, you are agreeing that
    - You neither given nor received help directly or indirectly to or from anyone else.
    - You did not directly or indirectly use materials from non-allowed sources.

Important

- You must use our files when coding.
- Do not print anything other than what is requested. Do not print labelling information for your output.
- Do not access past code or algorithms (yours, ours, or anyone else's).
- Do not access class notes, epistles, examples, artifacts, solutions on the web, or your own past assignments.
- Class personnel cannot help you debug your answers.
- Comment out or delete all debugging `print()` statements before submitting.
- Whether code is testable is important.
- The only device you may access during the exam is your laptop. The only open windows allowed are PyCharm and a browser with tabs linked from the class website.
- During the test you can access the course module descriptions and the course Python information sheet.
- You are responsible for submitting for your work, so check before leaving the test. Late submissions will not be graded, so do not submit once the test is over.
- Code should follow class programming practices, e.g., whitespace, identifier naming, etc.
- Because the solutions are all short, commenting is not necessary.
- You might add comments if you were unable to complete a problem and want to explain what you were attempting to do.

Programming

1. Implement program *im_honorable.py*. The program takes no input and *prints* either the lower-case word *yes* or *no*. The program does not print anything else. The program is to print *yes* if you were completely honorable when taking the test; and *no*, otherwise. The expected output is.

```
yes
```

COMPUTING ID ______________ **PRINT NAME LEGIBLY:** ______________________

2. Implement program *maximus.py*. The program has a single prompt requesting the user to supply two integers. The program prints the maximum supplied value.

   The program does not label its output. Here are three possible program runs.

```
Enter two numbers: 7 111
111
```

```
Enter two numbers: -3 -111
-3
```

```
Enter two numbers: 4 4
4
```

3. Implement program *ez.py*. The program has a single prompt requesting the user to supply text. The program produces two lines of output:

   - The first line is the number of occurrences of lower-case string `"ez"` in the text.
   - The second line is the average number of occurrences of `"ez"` per word in the text.

   The program does not label its output. Here are three possible program runs.

```
Enter text: eezy peezy
2
1.0
```

```
Enter text: eye lash tweezer set
1
0.25
```

```
Enter text: EZ come EZ go
0
0.0
```

4. Implement program *sneeze.py*. The program has a single prompt requesting the user to supply a number of sneezes. The program computes the number of days it would take an average person to sneeze that many times.

   Fun fact for your program to use: according to the experts that count, the average person sneezes three times a day. In printing the number of days, the program rounds the answer to at most two decimal places.

   The program does not label its output. Here are three possible program runs.

```
Number of sneezes: 6
2.0
```

```
Number of sneezes: 5
1.67
```

```
Number of sneezes: 4
1.33
```

5. Implement program *casing.py*. The program has two *separate* prompts requesting the user to supply text. The program produces three lines of output:

   - The first line is the first input in all lower case.
   - The second line is the second input in all upper case.
   - The third line is the total number of words in the input.

   The program does not label its output. Here are three possible program runs.

```
Enter text: Way to Go
Enter text: Which way did they go?
way to go
WHICH WAY DID THEY GO?
8
```

```
Enter text: Eye lash tweezers SCARE me a lot
Enter text: As a kid I thought several was a synonym for seven
eye lash tweezers scare me a lot
AS A KID I THOUGHT SEVERAL WAS A SYNONYM FOR SEVEN
18
```

```
Enter text: Red night at night sailor’s delight
Enter text: Why isn't a lot contracted to a lot?
red night at night sailor’s delight
WHY ISN'T A LOT CONTRACTED TO A LOT?
14
```

6. Implement program *triplets.py*. The program has a single prompt requesting the user to supply an integer number *n*. The program computes and prints the below sum

   $$0^3 + 1^3 + 2^3 + 3^3 + \ldots + n^3$$

   The program does not label its output. Here are three possible programs runs.

```
Enter n: 2
9
```

```
Enter n: 4
100
```

```
Enter n: 10
3025
```

7. Implement program *initialing.py*. The program has a single prompt requesting the user to supply text. The program produces two lines of output:

   - The first line is the list of supplied words.
   - The second line is a string composed of the first letter of each of the words in turn.

   The program does not label its output. Here are three possible program runs.

```
Enter text: So how are you
['So', 'how', 'are', 'you']
Shay
```

```
Enter text: Golf Hotel India Juliett
['Golf', 'Hotel', 'India', 'Juliett']
GHIJ
```

```
Enter text:
[]
```

8. Implement program *simpler_than_you_think.py*. The program has a single prompt requesting the user to supply integers.

   The program outputs the number of negative numbers entered (for our purposes zero is not negative). The program does not label its output.

   Observation: negative inputs have a `'-'`.

   Here are three possible program runs.

```
Enter integers: -7 0 -30 19
2
```

```
Enter integers: 12 -3 -4 4 -1 8 7 0
3
```

```
Enter integers: 21 191
0
```