Problem 1. (*Name and Age*) Write a program called `name_age.py` that accepts *name* (str) and *age* (str) as command-line arguments, and writes "*name* is *age* years old." to standard output.

```
>_ ~/workspace/exercise1
$ python3 name_age.py Alice 19
Alice is 19 years old.
$ python3 name_age.py Bob 23
Bob is 23 years old.
```

Problem 2. (*Greet Three*) Write a program called `greet_three.py` that accepts *name*1 (str), *name*2 (str), and *name*3 as command-line arguments, and writes "Hi *name3*, *name2*, and *name1*." to standard output.

```
>_ ~/workspace/exercise1
$ python3 greet_three.py Alice Bob Carol
Hi Carol, Bob, and Alice.
$ python3 greet_three.py Dan Eve Fred
Hi Fred, Eve, and Dan.
```

Problem 3. (*Triangle Inequality*) Write a program called `triangle.py` that accepts x (int), y (int), and z (int) as command-line arguments, and writes to standard output `True` if each one of them is less than or equal to the sum of the other two, and `False` otherwise.

```
>_ ~/workspace/exercise1
$ python3 triangle.py 3 3 3
True
$ python3 triangle.py 2 4 7
False
```

Problem 4. (*Body Mass Index*) The body mass index (BMI) is the ratio of the weight w of a person (in kg) to the square of the height h (in m). Write a program called `bmi.py` that accepts w (float) and h (float) as command-line arguments, and writes the BMI to standard output.

```
>_ ~/workspace/exercise1
$ python3 bmi.py 75 1.83
22.395413419331717
$ python3 bmi.py 97 1.75
31.6734693877551
```

Problem 5. (*Random Integer*) Write a program called `random_int.py` that accepts a (int) and b (int) as command-line arguments, and writes to standard output a random integer between a (inclusive) and b (exclusive).

```
>_ ~/workspace/exercise1
$ python3 random_int.py 10 20
13
$ python3 random_int.py 10 20
10
```

Files to Submit

1. `name_age.py`
2. `greet_three.py`
3. `triangle.py`
4. `bmi.py`

5. `random_int.py`

Before you submit your files, make sure:

- You do not use concepts from sections beyond "Basic Data Types".
- Your programs meet the style requirements by running the following command in the terminal.

```
>_ ~/workspace/exercise1
$ pycodestyle <program>
```

- Your code is adequately commented, follows good programming principles, and meets any specific requirements such as corner cases and running times.