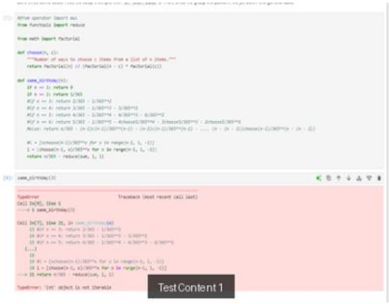
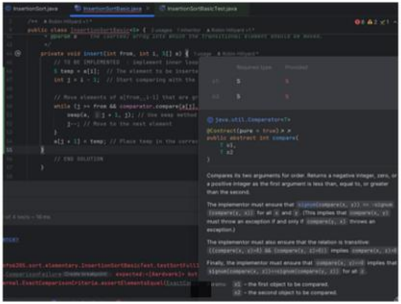
We can stretch any budget!
Ignore the ceiling, its fine
Why won't this machine work?
It just made things worse.
We can stretch any budget!
Ignore the ceiling, its fine
Why won't this machine work?
We can stretch any budget!
Ignore the ceiling, its fine
Why won't this machine work?
It just made things worse.
We can stretch any budget!
Ignore the ceiling, its fine
Why won't this machine work?
We can stretch any budget!
Ignore the ceiling, its fine
Why won't this machine work?
We can stretch any budget!
Ignore the ceiling, its fine
Why won't this machine work?



```
from functools import reduce

from math import factorial
```

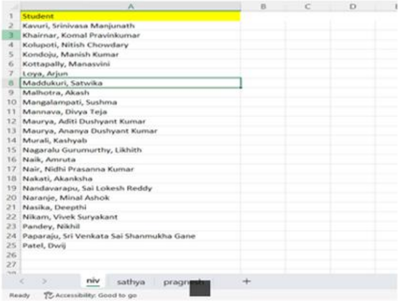
```
TypeError: 'int' object is not iterable
```



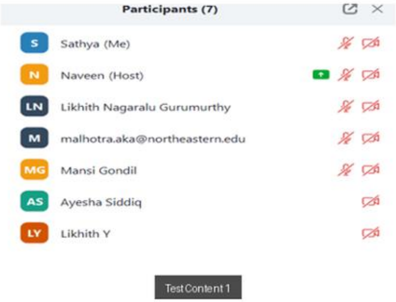
```
// END SOLUTION
```

Compares its two arguments for order. Returns a negative integer, zero, or a positive integer as the first argument is less than, equal to, or greater than the second.

x1 - the first object to be compared.
x2 - the second object to be compared.

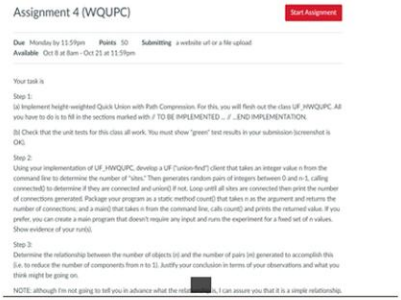


	A
1	Student
2	Kavuri, Srinivasa Manjunath
3	Khairnar, Komal Pravinkumar
4	Kolupoti, Nitish Chowdary
5	Kondoju, Manish Kumar
6	Kottapally, Manasvini
7	Loya, Arjun
8	Maddukuri, Satwika
9	Malhotra, Akash
10	Mangalampati, Sushma
11	Mannava, Divya Teja
12	Maurya, Aditi Dushyant Kumar
13	Maurya, Ananya Dushyant Kumar
14	Murali, Kashyab
15	Nagaralu Gurumurthy, Likhith
16	Naik, Amruta
17	Nair, Nidhi Prasanna Kumar
18	Nakati, Akanksha
19	Nandavarapu, Sai Lokesh Reddy
20	Naranje, Minal Ashok
21	Nasika, Deepthi
22	Nikam, Vivek Suryakant
23	Pandey, Nikhil
24	Paparaju, Sri Venkata Sai Shanmukha Gane
25	Patel, Dwij
26	
27	



Participants (7)

S Sathya (Me)
N Naveen (Host)
LN Likhith Nagaralu Gurumurthy
M malhotra.aka@northeastern.edu
MG Mansi Gondil
AS Ayesha Siddiq
LY Likhith Y



Assignment 4 (WQUPC)

Due Monday by 11:59pm **Points** 50 **Submitting** a website url or a file upload
Available Oct 8 at 8am - Oct 21 at 11:59pm

Your task is

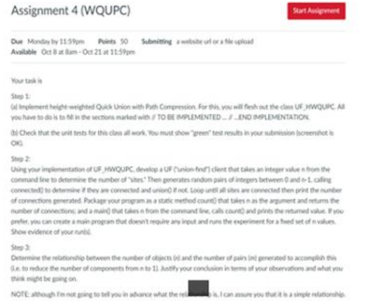
Step 1:
(a) Implement height-weighted Quick Union with Path Compression. For this, you will flesh out the class UF_HWQUPC. All you have to do is to fill in the sections marked with // TO BE IMPLEMENTED ... // ...END IMPLEMENTATION.

(b) Check that the unit tests for this class all work. You must show "green" test results in your submission (screenshot is OK).

Step 2:
Using your implementation of UF_HWQUPC, develop a UF ("union-find") client that takes an integer value n from the command line to determine the number of "sites." Then generates random pairs of integers between 0 and n-1, calling connected() to determine if they are connected and union() if not. Loop until all sites are connected then print the number of connections generated. Package your program as a static method count() that takes n as the argument and returns the number of connections; and a main() that takes n from the command line, calls count() and prints the returned value. If you prefer, you can create a main program that doesn't require any input and runs the experiment for a fixed set of n values. Show evidence of your run(s).

Step 3:
Determine the relationship between the number of objects (n) and the number of pairs (m) generated to accomplish this (i.e. to reduce the number of components from n to 1). Justify your conclusion in terms of your observations and what you think might be going on.

NOTE: although I'm not going to tell you in advance what the rela[redacted]is, I can assure you that it is a simple relationship.



Assignment 4 (WQUPC)

Start Assignment

Due Monday by 11:59pm **Points** 50 **Submitting** a website url or a file upload
Available Oct 8 at 8am - Oct 21 at 11:59pm

Your task is

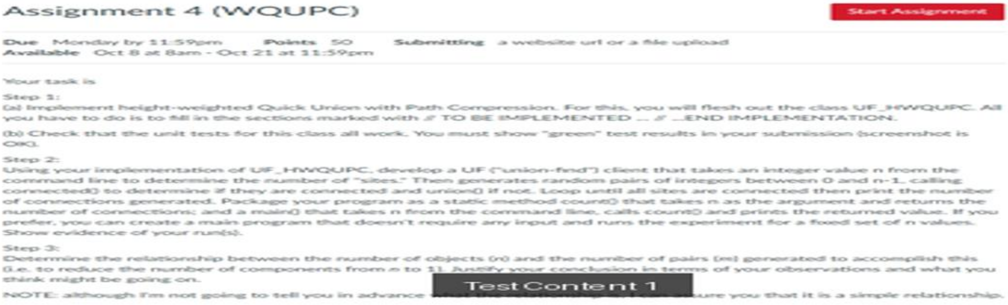
Step 1:
(a) Implement height-weighted Quick Union with Path Compression. For this, you will flesh out the class UF_HWQUPC. All you have to do is to fill in the sections marked with // TO BE IMPLEMENTED ... // ...END IMPLEMENTATION.

(b) Check that the unit tests for this class all work. You must show "green" test results in your submission (screenshot is OK).

Step 2:
Using your implementation of UF_HWQUPC, develop a UF ("union-find") client that takes an integer value n from the command line to determine the number of "sites." Then generates random pairs of integers between 0 and n-1, calling connected() to determine if they are connected and union() if not. Loop until all sites are connected then print the number of connections generated. Package your program as a static method count() that takes n as the argument and returns the number of connections; and a main() that takes n from the command line, calls count() and prints the returned value. If you prefer, you can create a main program that doesn't require any input and runs the experiment for a fixed set of n values. Show evidence of your run(s).

Step 3:
Determine the relationship between the number of objects (n) and the number of pairs (m) generated to accomplish this (i.e. to reduce the number of components from n to 1). Justify your conclusion in terms of your observations and what you think might be going on.

NOTE: although I'm not going to tell you in advance what the rel[redacted]is, I can assure you that it is a simple relationship.



Assignment 4 (WQUPC)

Start Assignment

Due Monday by 11:59pm **Points** 50 **Submitting** a website url or a file upload
Available Oct 8 at 8am - Oct 21 at 11:59pm

Your task is

Step 1:
(a) Implement height-weighted Quick Union with Path Compression. For this, you will flesh out the class UF_HWQUPC. All you have to do is to fill in the sections marked with // TO BE IMPLEMENTED ... // ...END IMPLEMENTATION.

(b) Check that the unit tests for this class all work. You must show "green" test results in your submission (screenshot is OK).

Step 2:
Using your implementation of UF_HWQUPC, develop a UF ("union-find") client that takes an integer value n from the command line to determine the number of "sites." Then generates random pairs of integers between 0 and n-1, calling connected() to determine if they are connected and union() if not. Loop until all sites are connected then print the number of connections generated. Package your program as a static method count() that takes n as the argument and returns the number of connections; and a main() that takes n from the command line, calls count() and prints the returned value. If you prefer, you can create a main program that doesn't require any input and runs the experiment for a fixed set of n values. Show evidence of your run(s).

Step 3:
Determine the relationship between the number of objects (n) and the number of pairs (m) generated to accomplish this (i.e. to reduce the number of components from n to 1). Justify your conclusion in terms of your observations and what you think might be going on.

NOTE: although I'm not going to tell you in advance

sure you that it is a simple relationship.

Melissa staring at engagement numbers.

Janine sipping coffee with Melissa.

Melissa nervously sipping oversized mug.

Janine sketching 'Mood Index' chart.

Melissa staring at engagement numbers.

Janine sipping coffee with Melissa.

Melissa nervously sipping oversized mug.

Melissa staring at engagement numbers.

Janine sipping coffee with Melissa.

Melissa nervously sipping oversized mug.

Janine sketching 'Mood Index' chart.

Melissa staring at engagement numbers.

Janine sipping coffee with Melissa.

Melissa nervously sipping oversized mug.

Melissa staring at engagement numbers.

Janine sipping coffee with Melissa.

Melissa nervously sipping oversized mug.

Melissa staring at engagement numbers.

Janine sipping coffee with Melissa.

Melissa nervously sipping oversized mug.

Janine holding up data charts.

Melissa unloading her supplies quietly.

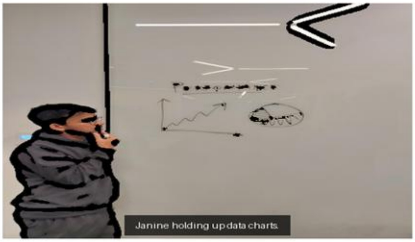
Ava shocked by enrollment numbers.

Janine holding up data charts.

Melissa unloading her supplies quietly.

Ava shocked by enrollment numbers.

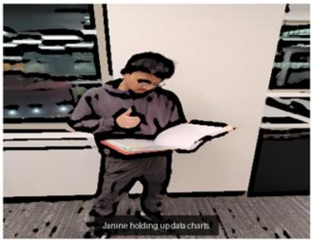
Gregory nervously cutting pizza slices.



Janine holding up data charts.

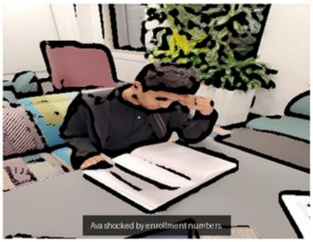
Melissa unloading her supplies quietly.

Ava shocked by enrollment numbers.



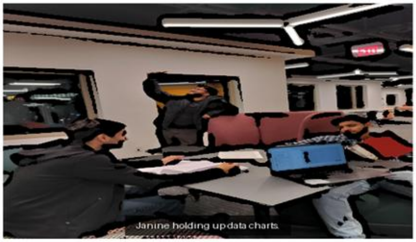
Janine holding up data charts.

Melissa unloading her supplies quietly.



Ava shocked by enrollment numbers.

Gregory nervously cutting pizza slices.



Janine holding up data charts.

Melissa unloading her supplies quietly.

Ava shocked by enrollment numbers.

Janine holding up data charts.

Melissa unloading her supplies quietly.

Ava shocked by enrollment numbers.