

1. **Test Content 1**  
 2. **Test Content 2**  
 3. **Test Content 3**

1. **Test Content 1**  
 2. **Test Content 2**  
 3. **Test Content 3**

1. **Test Content 1**  
 2. **Test Content 2**  
 3. **Test Content 3**

## 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

Test Content 1

## 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: To implement height-weighted Quick Union with Path Compression. For this, you will flesh out the class UF\_HWQUPC. All you have to do is fill in the sections marked with `// TO BE IMPLEMENTED`.
- Step 2: Check that the unit tests for this class all work. You must show "green" test results in your submission (screenshot is OK).
- Step 3: 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 "islands" (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 runs).
- Step 4: 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

Test Content 2

are you that it is a simple relationship.

## 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: To implement height-weighted Quick Union with Path Compression. For this, you will flesh out the class UF\_HWQUPC. All you have to do is fill in the sections marked with `// TO BE IMPLEMENTED`.
- Step 2: Check that the unit tests for this class all work. You must show "green" test results in your submission (screenshot is OK).
- Step 3: 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 "islands" (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 runs).
- Step 4: 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

Test Content 3

are you that it is a simple relationship.

## 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: To implement height-weighted Quick Union with Path Compression. For this, you will flesh out the class UF\_HWQUPC. All you have to do is fill in the sections marked with `// TO BE IMPLEMENTED`.
- Step 2: Check that the unit tests for this class all work. You must show "green" test results in your submission (screenshot is OK).
- Step 3: 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 "islands" (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 runs).
- Step 4: 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

Test Content 1

are you that it is a simple relationship.

	A	B	C	D
1	Student			
2	Karori, Srinivasa Manjunath			
3	Khairnar, Komal Praveenkumar			
4	Kirkpoti, Nitish Chowdhary			
5	Kondibole, Manish Kumar			
6	Kottapally, Manasvini			
7	Loye, Arjun			
8	Madhukrishna, Satwikha			
9	Malhotra, Akash			
10	Mangalamarti, Surinsha			
11	Maniavaia, Divya Teja			
12	Mauriya, Aditi Dushyant Kumar			
13	Mauriya, Ananya Dushyant Kumar			
14	Murali, Kishalya			
15	Nagaraju Gurumurthy, Likhith			
16	Nethi, Anuradha			
17	Nair, Nidhi Prasanna Kumar			
18	Nakati, Akanksha			
19	Nandamavara, Sai Lokesh Reddy			
20	Naranjan, Minat Ashok			
21	Nasika, Durgesh			
22	Nekam, Vivek Suryakant			
23	Pandey, Nikhil			
24	Papareddy, Sri Venkata Sai Shanmukha Gane			
25	Patel, Divi			
26				
27				

## Assignment 4 (WQUC)

Start Assignment

Due Monday by 11:59pm Points: 10 Submitting a website or a file upload Available On 8 Sat - On 8 Sat at 11:59pm

---

Your task is

- Step 1:  
Implement helper-weighted Quicksort with  $P$ -Bit Compression. For this, you will find out the class `WQUC`. All you have to do is to fix in the section marked with  $P$  BIT IMPLEMENTATION.
- Step 2:  
Check that the unit tests for this class are ok. You must show "green" test results in your submission screenshot is OK.
- Step 3:  
Using your implementation of `WQUC`, develop a `WQUC` function that takes an integer value  $n$  from the keyboard to determine the number of "wins". Then generate random pairs of integers between 1 and  $n$ , calling `WQUC` to determine if they are connected and output `WQUC`. Your code unit will also be connected flow print the number of connections generated. Package your program in a static method `countWins` in the class `WQUC` and return the number of connections, and a matrix that takes from the class `WQUC` and prints the returned value. From here, you can create a main program that doesn't require any input and run the experiment for a fixed set of values. (Show evidence of your work).
- Step 4:  
Determine the relationship between the number of objects in `WQUC` and the number of pairs in `WQUC` to accomplish this. Is 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 `WQUC` is, I can assure you that it is a simple relationship.

Due: Monday by 11:59pm      #Pals: 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 "sides." Then generates random pairs of integers between 0 and `n-1`, calling `union(i,j)` to determine if they are connected and `union(i,j)` if not. Loop until all sides 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 may write 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. Use to reduce the number of components from `m` 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, **Test Content 1** assure you that it is a simple relationship.