In this assignment, you have to write a class to implement an abstract data type called permutation. A permutation of n is a one-to-one and onto function p from {0,1,2,...,n-1} to itself, such that p(i) = p(j) if and only if i=j. The name of the class must be permutation, which will be used in the main program.

Implement the following operations on permutations. Note that you can assume that all operations would be used correctly, and do not need to do any error checking. Here p denotes the variable to which these methods are applied. The identity permutation is given by p(i) = i for 0 <= i < n.

1. Constructor: This should create a new object of type permutation. This should have two parameters, an integer n, and an integer array a of size n, where a[i] gives the value of the permutation at i. It can be assumed that 1 <= n <= 100, and the array is a permutation. The declaration for this would be

```
permutation(int n, int a[]);
```

Note that whenever you define your own constructor, you also need to define your own destructor, and copy constructor for the class. It would also help to define an assignment operator = to assign the value of one permutation to another. The copy constructor is used when a new variable of type permutation is created, while assignment is used when assigning a value to an already created variable. The declarations for these would be

```
~permutation();
permutation(permutation const &q);
permutation const operator=(permutation const &q);
```

2. int size() const;

returns the size n of p.

2.int* to_array() const;

Converts the permutation p to an integer array a of size n such that a[i] = p(i) and returns the array.

3. permutation const operator-() const;

returns the inverse of the permutation p. This will be used as -p.

4. permutation const operator*(permutation const &q) const;

This is the composition operator. If p, q are permutations of the same size, p*q will return the permutation r such that r(i) = p(q(i)) for 0 <= i < n.

5. permutation const square_root() const;

If the permutation p = f*f for some permutation f, then returns the permutation f, otherwise returns identity. Note that there may be many choices of f, in which case any one can be returned.

6. permutation const operator^(long long int i) const;

Given a long long integer i >= 0, this returns the permutation p^i defined by p^0 = identity and p^(i+1) = p*(p^i). Note that i may be a long integer so 0 <= i <= 10^18.

7. bool is_power(permutation const &q) const;

returns true if there is some integer i such that p = q^i and false otherwise.

8. int log(permutation const &q) const;

Find the smallest possible value of i, if it exists, such that p = q^i. This may be very large, so compute it modulo 10^9+7. Output 0 if there is no such value.

You should create a single header file containing the class definition and the methods used in the class. This should NOT contain any main function. If you use a main function for testing, make sure you delete it before submitting. You can include any other standard C++ library files if needed. Name the file RollNo_1.h and submit on moodle.