1. Polymorphism

Write three classes:

- A **Parent** class having a **filter()** method (string type returning null).
- Two classes **ChildOne** and **ChildTwo** that extend from the class **Parent**, both overriding the **filter()** method from the **Parent** class.

1. You are given two integers representing the range with **start** value and **end** value.
2. Declare two variables in the **Parent** class of type **int** and scope **public** named **startElement** and **endElement** which will represent the start and end element of the range of integers.
3. The **filter()** method in the **ChildOne** class should return a string consisting of all the **prime** numbers within the given range.
4. The **filter()** method in the **ChildTwo** class should return a string consisting of all the **happy** numbers within the given range.

Happy numbers are those numbers that return **1** when they are replaced by the sum of the square of the digits repeatedly.

Example:

$91 - 9^2 + 1^2$
$82 - 8^2 + 2^2$
$68 - 6^2 + 8^2$
$100 - 1^2 + 0^2 + 0^2$
1

The successive addition of squares of the digits of 91 yields 1. Therefore, 91 is a happy number.

Note: *A number is considered* ***unhappy*** *when repeatedly the sum of the square of the digits returns* ***4****.*

Input Specifications:
The first line is an integer representing the starting number in the range (inclusive).
The second line is an integer representing the ending number in the range (inclusive).

Output Specifications:
The first line should consist of a string with the prime numbers in the range (each number separated by a space).
The second line should consist of a string with the happy numbers in the range (each number separated by a space).

Sample Input:

1

150

Sample Output:

2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89 97 101 103 107 109 113 127 131 137 139 149

1 7 10 13 19 23 28 31 32 44 49 68 70 79 82 86 91 94 97 100 103 109 129 130 133 139