Homework on Object-Oriented Programming

Let n be a positive integer. Define a function $th(n)$ as the sum of the digits of n when expressed as a decimal numeral. For instance, $th(348) = 3+4+8 = 15$, $th(15) = 1+5 = 6$, etc. Define

$$sh(n) =_{def} \begin{cases} n, \text{ if } n < 10. \\ sh(th(n)), \text{ otherwise} \end{cases}$$

For instance $sh(348) = sh(15) = 6$.

Define a positive integer n as a *sohot number* if it satisfies the following three conditions:

- $n \geq 10$.
- $sh(n) = 2, 3, 5$, or 7.
- For every $k < n$ such that k is a sohot number, k cannot divide n.

For instance, 10 is not sohot. 11, 12, 14, 16, 20, 21, 23, 25, 29, etc. are sohot.

In this homework, you are asked to write a C++/Java program to compute all the sohot numbers less than 200. You need to follow an object-oriented paradigm. You may modify the program *Eratosthenes's Sieve* for this homework.

Here is the source code of *Eratosthenes's Sieve*.

```
(1) #include <stdio.h>
(2) class Item {
(3) public:
(4)        Item *source;
(5)              Item(Item *src) {source = src; }
(6) virtual int out()            {return 0}
(7) };
(8) class Counter:public Item {
```

```
 (9)     int value;
(10) public:
(11)     int out()                  { return value++;}
(12)         Counter(int v):Item(0) {value = v;}
(13) };
(14) class Sieve:public Item {
(15) public:
(16)     int out();
(17)         Sieve(Item *src):Item(src) { }
(18) };
(19) class Filter:public Item {
(20)     int factor;
(21) public:
(22)     int out();
(23)         Filter(Item *src, int f):Item(src) {factor = f;}
(24) };
(25) main() {
(26)       Counter c(2);
(27)       Sieve   s(&c);
(28)       int     next;
(29)       do {
(30)          next = s.out();
(31)          printf(%d ,next);
(32)       } while (next < 61);
(33)       printf(\n);
(34) }
(35) int Sieve::out() {
(36)     int n  = source->out();
(37)     source = new Filter(source, n);
(38)     return n;
```

```
(39) };
(40) int Filter::out() {
(41)     while(1) {
(42)         int n = source->out();
(43)         if (n \% factor) return n;
(44)     };
(45) };
```