Code in C++

...\Computational Physics\Sieve of Eratosthenes\Main.cpp 1

```cpp
#include <memory>   //for smart pointers
#include <iostream> //for input-output


uint64_t SieveOfEratosthenes(size_t n) {    //Definition of a function that returns nth prime
    size_t upperbound = n * n;  //Upper bound for nth prime, search until this number
    auto tags = std::make_unique<bool[]>(upperbound);   //unique pointer to array of booleans(indicating prime/composite)

    size_t count = 0;   //a counter to keep track of the index of prime numbers

    for (size_t i = 2; i * i < upperbound; i++) {//count from 2 to square root of upper bound
        if (!tags[i]) { //if it is not already listed as composite,
            for (size_t multiplier = i; i * multiplier < upperbound; multiplier++) { //Loop over the multipliers
                tags[i * multiplier] = true;    //tag the multiples as composite
            }
        }
    }

    for (size_t i = 2; i < upperbound; i++) //count and return the n-th prime number
    {
        if (!tags[i]) {
            count++;
            if (count == n)
                return i;
        }
    }
}

int main() {
    size_t n = 100;
    for (size_t i = 5; i <= n; i++) { //print from 5th to 100th prime numbers
        std::cout << i << "-th prime is:" << SieveOfEratosthenes(i) << "\n";
    }

    return 0;
}
```

Code in C

...omputational Physics\Sieve of Eratosthenes - C\Main.c 1

```c
#include <stdio.h>
#include <stdbool.h>


size_t SieveOfEratosthenes(size_t n) {  //Definition of a function that returns nth prime
    size_t upperbound = 100 * 100;  //Upper bound for nth prime, search until this number
    bool tags[100 * 100];   //no dynamic memory allocation like in c++, have to hardcode it or use a buffer system
    memset(tags, false, sizeof(tags));  //set all as tentative primes

    size_t count = 0;   //a counter to keep track of the index of prime numbers

    for (size_t i = 2; i * i < upperbound; i++) {//count from 2 to square root of upper bound
        if (!tags[i]) { //if it is not already listed as composite,
            for (size_t multiplier = i; i * multiplier < upperbound; multiplier++) { //Loop over the multipliers
                tags[i * multiplier] = true;    //tag the multiples as composite
            }
        }
    }

    for (size_t i = 2; i < upperbound; i++) //count and return the n-th prime number
    {
        if (!tags[i]) {
            count++;
            if (count == n)
                return i;
        }
    }
}

int main() {
    size_t n = 100;
    for (size_t i = 5; i <= n; i++) { //print from 5th to 100th prime numbers
        printf("%i-th prime is: %i\n", i, SieveOfEratosthenes(i));
    }

    return 0;
}
```

Microsoft Visual Studio Debug Console

```
5-th prime is: 11
6-th prime is: 13
7-th prime is: 17
8-th prime is: 19
9-th prime is: 23
10-th prime is: 29
11-th prime is: 31
12-th prime is: 37
13-th prime is: 41
14-th prime is: 43
15-th prime is: 47
16-th prime is: 53
17-th prime is: 59
18-th prime is: 61
19-th prime is: 67
20-th prime is: 71
21-th prime is: 73
22-th prime is: 79
23-th prime is: 83
24-th prime is: 89
25-th prime is: 97
26-th prime is: 101
27-th prime is: 103
28-th prime is: 107
29-th prime is: 109
30-th prime is: 113
31-th prime is: 127
32-th prime is: 131
33-th prime is: 137
34-th prime is: 139
35-th prime is: 149
36-th prime is: 151
37-th prime is: 157
38-th prime is: 163
39-th prime is: 167
40-th prime is: 173
41-th prime is: 179
42-th prime is: 181
43-th prime is: 191
44-th prime is: 193
45-th prime is: 197
46-th prime is: 199
47-th prime is: 211
48-th prime is: 223
49-th prime is: 227
50-th prime is: 229
51-th prime is: 233
52-th prime is: 239
53-th prime is: 241
54-th prime is: 251
```

Microsoft Visual Studio Debug Console

```
55-th prime is: 257
56-th prime is: 263
57-th prime is: 269
58-th prime is: 271
59-th prime is: 277
60-th prime is: 281
61-th prime is: 283
62-th prime is: 293
63-th prime is: 307
64-th prime is: 311
65-th prime is: 313
66-th prime is: 317
67-th prime is: 331
68-th prime is: 337
69-th prime is: 347
70-th prime is: 349
71-th prime is: 353
72-th prime is: 359
73-th prime is: 367
74-th prime is: 373
75-th prime is: 379
76-th prime is: 383
77-th prime is: 389
78-th prime is: 397
79-th prime is: 401
80-th prime is: 409
81-th prime is: 419
82-th prime is: 421
83-th prime is: 431
84-th prime is: 433
85-th prime is: 439
86-th prime is: 443
87-th prime is: 449
88-th prime is: 457
89-th prime is: 461
90-th prime is: 463
91-th prime is: 467
92-th prime is: 479
93-th prime is: 487
94-th prime is: 491
95-th prime is: 499
96-th prime is: 503
97-th prime is: 509
98-th prime is: 521
99-th prime is: 523
100-th prime is: 541

C:\Users\rajat\Documents\Visual Studio 2022\projects\Computational Physics\x64\Debug\Sieve of Eratosthenes - C.exe (process 1396) exited with code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops.
Press any key to close this window . . .
```