```cpp
// ComparePrimeSieves.cpp (c) Saturday September 14, 2024
// by James Pate Williams, Jr.
// Compares the Sieve of Atkin to the Sieve of Eratosthenes

#include "Sieves.h"
#include <chrono>
#include <iomanip>
#include <iostream>
using namespace std::chrono;
using namespace std;

long sieve[SIEVE_SIZE];

void CreateRuntimeTable()
{
    cout << "Runtimes in microseconds:" << endl;
    cout << setw(6) << "Limit";
    cout << setw(12) << "Atkin1";
    cout << setw(12) << "Erato1";
    cout << setw(12) << "Atkin2";
    cout << setw(12) << "Erato2";
    cout << endl;

    for (long limit = 100000; limit <= 250000; limit += 10000)
    {
        cout << setw(6) << limit;
        auto start1 = std::chrono::high_resolution_clock::now();
        vector<long> slowAtkinPrimes = Sieves::SlowGetAtkinPrimes(limit);
        auto stop1 = std::chrono::high_resolution_clock::now();
        auto duration1 = std::chrono::duration_cast<microseconds>
            (stop1 - start1);
        auto start2 = std::chrono::high_resolution_clock::now();
        vector<long> slowEratoPrimes = Sieves::SlowGetEratoPrimes(limit);
        auto stop2 = std::chrono::high_resolution_clock::now();
        auto duration2 = std::chrono::duration_cast<std::chrono::microseconds>
            (stop2 - start2);
        auto start3 = std::chrono::high_resolution_clock::now();
        vector<long> fastAtkinPrimes = Sieves::FastGetAtkinPrimes(
            limit, sieve);
        auto stop3 = std::chrono::high_resolution_clock::now();
        auto duration3 = std::chrono::duration_cast<std::chrono::microseconds>
            (stop3 - start3);
        auto start4 = std::chrono::high_resolution_clock::now();
        vector<long> fastEratoPrimes = Sieves::FastGetEratoPrimes(
            limit, sieve);
        auto stop4 = std::chrono::high_resolution_clock::now();
        auto duration4 = std::chrono::duration_cast<std::chrono::microseconds>
            (stop4 - start4);
        cout << setw(12) << duration1.count();
        cout << setw(12) << duration2.count();
        cout << setw(12) << duration3.count();
        cout << setw(12) << duration4.count();
```

```
53          cout << endl;
54      }
55  }
56
57  int main()
58  {
59      for (;;) {
60          long option = 0;
61          cout << "== Menu ==" << endl;
62          cout << "1 Test prime number generation" << endl;
63          cout << "2 Create runtimes table" << endl;
64          cout << "3 Exit" << endl;
65          cout << "Enter an option: ";
66          cin >> option;
67
68          if (option == 3)
69              break;
70
71          if (option == 1)
72          {
73              int limit = 0;
74              cout << "limit = ";
75              cin >> limit;
76              vector<long> slowAtkinPrimes = Sieves::SlowGetAtkinPrimes(limit);
77              vector<long> slowEratoPrimes = Sieves::SlowGetEratoPrimes(limit);
78              vector<long> fastAtkinPrimes = Sieves::FastGetAtkinPrimes(
79                  limit, sieve);
80              vector<long> fastEratoPrimes = Sieves::FastGetEratoPrimes(
81                  limit, sieve);
82              long sapn = slowAtkinPrimes.size();
83              long sepn = slowEratoPrimes.size();
84              long fapn = slowAtkinPrimes.size();
85              long fepn = slowEratoPrimes.size();
86              for (int i = 0; i < sapn; i++) {
87                  cout << setw(2) << (i + 1) << '\t';
88                  cout << fastAtkinPrimes[i] << '\t' << fastAtkinPrimes[i] << '\t';
89                  cout << fastEratoPrimes[i] << '\t' << fastEratoPrimes[i] << endl;
90              }
91          }
92          else if (option == 2)
93              CreateRuntimeTable();
94      }
95
96      return 0;
97  }
```