

Exercises week 7 - Multi-threading I

Klaas Isaac Bijlsma
s2394480

David Vroom
s2309939

January 17, 2018

Exercise 49

Learn to apply basic multi-threading

We used the following code.

main.ih

```
1 | #include <string>
2 | #include <thread>
3 | #include <chrono>
4 | #include <vector>
5 | #include <algorithm>
6 | #include <iostream>
7 | #include <iomanip>
8 |
9 | using namespace std;
10 | using namespace chrono;
11 |
12 | void waiting(bool &ready);
```

waiting.cc

```
1 | #include "main.ih"
2 |
3 | void waiting(bool &ready)
4 | {
```

```

5   while (!ready)
6   {
7       cerr << '.';
8       this_thread::sleep_for(seconds(1));
9   }
10  cerr << '\n';
11 }

```

main.cc

```

1  #include "main.ih"
2
3  int main(int argc, char **argv)
4  {
5      size_t nPrimes = stoull(argv[1]);
6      bool ready = false;
7
8      thread wait(waiting, ref(ready));
9      auto startTime = system_clock::to_time_t(system_clock::now());
10
11     vector<size_t> vec{2};
12     size_t next = 3;
13
14     while (vec.size() != nPrimes)
15     {
16         auto iter =
17             find_if(vec.begin(), vec.end(),
18                 [=](size_t prime)
19                 {
20                     return next % prime == 0;
21                 }
22             ); // Eratosthenes sieve
23         if (iter == vec.end())
24             vec.push_back(next); // next is prime number
25         ++next;
26     }
27
28     auto endTime = system_clock::to_time_t(system_clock::now());
29     ready = true; // Notify waiting thread that computation finished
30     wait.join();

```

```

31
32     for (size_t elem: vec)
33         cout << elem << ' ';
34     cout << '\n';
35
36     cout << put_time(localtime(&startTime), "Starting time: %c") << '\n'
37         << put_time(localtime(&endTime), "Ending time: %c") << '\n'
38         << "Computation of " << nPrimes << " primes took "
39         << endTime - startTime << " seconds\n";
40 }

```

Exercise 50

Learn to perform time conversions

We used the following code.

main.cc

```
1 #include <iostream>
2 #include <chrono>
3
4 using namespace std;
5 using namespace chrono;
6
7 int main()
8 {
9     cout << "Hours: ";
10    int nHours;
11    cin >> nHours;
12
13    cout << "is equal to "
14          << minutes(hours(nHours)).count()
15          << " minutes\n";
16
17    cout << "Seconds: ";
18    int nSec;
19    cin >> nSec;
20
21    cout << "is equal to "
22          << seconds(nSec).count() / seconds(minutes(1)).count()
23          << " minutes\n";
24 }
```

Exercise 51

Learn to use the chrono/clock facilities

We used the following code.

main.cc

```
1 #include <iostream>
2 #include <chrono>
3 #include <iomanip>
4 #include <string.h>
5
6 using namespace std;
7 using namespace chrono;
8
9 int main(int argc, char **argv)
10 {
11     // get the current time
12     time_point<system_clock> timePoint{system_clock::now()};
13
14     // convert it to a std::time_t:
15     time_t time = system_clock::to_time_t(timePoint);
16
17     // display the time:
18     cout << left << setw(14) << "Current time:"
19          << put_time(localtime(&time), "%c") << '\n';
20
21     // display the gmtime
22     cout << left << setw(14) << "Gmtime:"
23          << put_time(gmtime(&time), "%c") << '\n';
24
25     string arg = argv[1];
26     char suffix = arg.back();
27     int count = stoi(arg);
28
29     // add or subtract specified time to now
30     if (suffix == 's')
31         timePoint += seconds(count);
32     else if (suffix == 'm')
33         timePoint += minutes(count);
```

```

34 |     else if (suffix == 'h')
35 |         timePoint += hours(count);
36 |
37 |         // convert it to a std::time_t:
38 |         time_t newTime = system_clock::to_time_t(timePoint);
39 |
40 |         // display the time:
41 |         cout << left << setw(14) << "New time:"
42 |             << put_time(localtime(&newTime), "%c") << '\n';
43 |     }

```

Exercise 52

Learn to define a thread with objects that aren't functors

We used the following code.

handler/handler.ih

```
1 #include "handler.h"
2 #include <iostream>
3
4 using namespace std;
```

handler/handler.h

```
1 #ifndef INCLUDED_HANDLER_H
2 #define INCLUDED_HANDLER_H
3
4 #include <ostream>
5 #include <string>
6
7 class Handler
8 {
9     public:
10         void shift(std::ostream &out, std::string const &text);
11 };
12
13 #endif
```

handler/shift.cc

```
1 #include "handler.ih"
2
3 void Handler::shift(ostream &out, string const &text)
4 {
5     string str(text);
6     out << str << '\n';
7
8     for (size_t idx = 1; idx != str.size(); ++idx)
```

```

9      {
10         char first = str[0];
11         str.erase(0,1);
12         str.push_back(first);
13         out << str << '\n';
14     }
15 }

```

main.cc

```

1 #include <iostream>
2 #include <fstream>
3 #include "handler/handler.h"
4
5 using namespace std;
6
7 int main()
8 {
9     Handler object;
10    string txt = "mouse";
11    ofstream out("output.txt");
12    object.shift(out, txt);
13 }

```


Exercise 53

Learn to design a simple producer/consumer program