```
void complex_fnt( std::string& s)
{
    std::stringstream buff;
    std::ofstream f("./test.txt");
    //write to file string passed + thread id
    if(f.is_open())
    {
        buff << s << std::this_thread::get_id() << std::endl;
        f << buff.str();
        f.close();
    }
}
```

```
std::string s = "Hello world ";
//complex object passed by reference
std::thread t(complex_fnt, std::ref(s));

//detach the thread... if the program exits before the threads
completes its job, no job is done
t.detach();
```

join vs detach

```
void complex_fnt( std::string& s)
{
   std::stringstream buff;
   std::ofstream f("./test.txt");
   //write to file string passed + thread id
   if(f.is_open())
   {
      buff << s << std::this_thread::get_id() << std::endl;
      f << buff.str();
      f.close();
    }
}


  std::string s = "Hello world ";
  //complex object passed by reference
  std::thread t(complex_fnt, std::ref(s));

  //detach the thread... if the program exits before the threads
  completes its job, no job is done
  t.detach();
```

join vs detach

```
nik@Nicolas-MacBook-Air:~/GitHub/cpp_sandbox/multithreading/thread_spawn$ ./a.out  2
-- Detach a complex task --
nik@Nicolas-MacBook-Air:~/GitHub/cpp_sandbox/multithreading/thread_spawn$ ls
a.out*          async_check.cpp  fancy_object.h   scoped_thread.h  thread1.cpp
thread2.cpp     thread3.cpp      thread4.cpp
nik@Nicolas-MacBook-Air:~/GitHub/cpp_sandbox/multithreading/thread_spawn$
```