Dr. B. Güsmann VGU

Bachelor of Computer Science.
Introduction to Programming with C: Exercises

Exercise Sheet 14

---

1. a) Run E.9-4 from the lecture.

   b) Set Sleep(10000L) in main() in comment brackets and enter immediately a key when asked by the program. Set also getchar() in main in comment brackets. What happens?

   c) Set Sleep(10L) in printslowly() to Sleep(100L). What happens.
   Set Sleep(10L) in printslowly() to Sleep(1L). What happens.
   Set Sleep(10L) in printslowly() in comment brackets. What happens.

2. a) Extend E.9-4 from the lecture to E.9-5 using a mutex object.

   b) Use the mutex object only for thread 2() and thread3(), n o t for thread1(). What happens?

   c) Set the time-out interval in WaitForSingleObject() to 1L in all 3 threads. What happens?

3. In a scientific paper (see (6) in References) I found the following function of number theory:

$$f(x) = if\ odd(x)\ then\ \frac{3*x+1}{2}\ else\ \frac{1}{2}x$$

   An iteration is the repeated application of the function, starting with a positive int value, that is: x_0 = start value, $x_{i+1} = f(x_i)$. It is supposed, that this iteration is finite and ends with 1. You shall test it.

   Define an *enumeration type boolean* with values *FALSE* and *TRUE*. Define a *function boolean odd(unsigned int);* , which returns *TRUE*, if the actual parameter is odd, else returns *FALSE*.

   Now define *f(x)* as defined above. *f(x)* uses *odd(x)* .

   Finally program an iteration in main() which stops at the value 1 as a return value from *f(x)* or after at most 50 iterations. Print all values of the iteration.

   Start the ietration several times with the values

   3, 5, 7, 9, 11, 13, 15

   Do all iterations end with 1? If you compare your intermediate results with the results in the scientific paper, you will realize that there are printing errors in the paper.