



Министерство науки и высшего образования Российской Федерации  
Федеральное государственное бюджетное образовательное  
учреждение высшего образования  
«Московский государственный технический университет имени  
Н.Э. Баумана  
(национальный исследовательский университет)»  
(МГТУ им. Н.Э. Баумана)

---

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

## Отчет по лабораторной работе №6 по курсу «Операционные системы»

Тема Задача «Читатели – писатели» под ОС Windows

Студент Кононенко С.С.

Группа ИУ7-53Б

Оценка (баллы) \_\_\_\_\_

Преподаватели Рязанова Н.Ю.

# Задача «Читатели – писатели»

Листинг 1 – Реализация задачи «читатели – писатели»

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <stdbool.h>
4 #include <windows.h>
5
6 #define READERS_COUNT 5
7 #define WRITERS_COUNT 3
8
9 #define READ_ITERS 7
10 #define WRITE_ITERS 8
11
12 #define READ_TIMEOUT 300
13 #define WRITE_TIMEOUT 300
14
15 #define DIFF 4000
16
17 HANDLE mutex;
18 HANDLE can_read;
19 HANDLE can_write;
20
21 LONG waiting_writers = 0;
22 LONG waiting_readers = 0;
23 LONG active_readers = 0;
24
25 bool active_writer = false;
26
27 int val = 0;
28
29 void start_read()
30 {
31     InterlockedIncrement(&waiting_readers);
32
33     if (active_writer || (WaitForSingleObject(can_write, 0) ==
34         WAIT_OBJECT_0 && waiting_writers))
35     {
36         WaitForSingleObject(can_read, INFINITE);
37     }
38     WaitForSingleObject(mutex, INFINITE);
39
40     InterlockedDecrement(&waiting_readers);
41     InterlockedIncrement(&active_readers);
```

```

41
42     SetEvent(can_read);
43     ReleaseMutex(mutex);
44 }
45
46 void stop_read()
47 {
48     InterlockedDecrement(&active_readers);
49
50     if (active_readers == 0)
51     {
52         ResetEvent(can_read);
53         SetEvent(can_write);
54     }
55 }
56
57 void start_write(void)
58 {
59     InterlockedIncrement(&waiting_writers);
60
61     if (active_writer || active_readers > 0)
62     {
63         WaitForSingleObject(can_write, INFINITE);
64     }
65
66     InterlockedDecrement(&waiting_writers);
67
68     active_writer = true;
69 }
70
71 void stop_write(void)
72 {
73     active_writer = false;
74
75     if (waiting_readers)
76     {
77         SetEvent(can_read);
78     }
79     else
80     {
81         SetEvent(can_write);
82     }
83 }
84
85 DWORD WINAPI rr_run(CONST LPVOID lpParams)
86 {
87     int r_id = (int)lpParams;
88     srand(time(NULL) + r_id);

```

```

89
90     int stime;
91
92     for (size_t i = 0; i < READ_ITERS; i++)
93     {
94         stime = READ_TIMEOUT + rand() % DIFF;
95         Sleep(stime);
96         start_read();
97         printf("?Reader_#%d_read:_%3d_//_Idle_time:_%dms\n", r_id, val,
98             stime);
99         stop_read();
100     }
101
102     return 0;
103 }
104
105 DWORD WINAPI wr_run(CONST LPVOID lpParams)
106 {
107     int w_id = (int)lpParams;
108     srand(time(NULL) + w_id + READERS_COUNT);
109
110     int stime;
111
112     for (size_t i = 0; i < WRITE_ITERS; ++i)
113     {
114         stime = WRITE_TIMEOUT + rand() % DIFF;
115         Sleep(stime);
116         start_write();
117         ++val;
118         printf("!Writer_#%d_wrote:_%3d_//_Idle_time:_%dms\n", w_id, val,
119             stime);
120         stop_write();
121     }
122
123     return 0;
124 }
125
126 int main()
127 {
128     setbuf(stdout, NULL);
129
130     HANDLE readers_threads[READERS_COUNT];
131     HANDLE writers_threads[WRITERS_COUNT];
132
133     if ((mutex = CreateMutex(NULL, FALSE, NULL)) == NULL)
134     {
135         perror("Failed_call_of_CreateMutex");
136
137         return -1;
138     }
139 }

```

```

135     }
136
137     can_read = CreateEvent(NULL, FALSE, FALSE, NULL);
138     can_write = CreateEvent(NULL, FALSE, FALSE, NULL);
139
140     if (can_read == NULL || can_write == NULL)
141     {
142         perror("Failed_call_of_CreateEvent");
143
144         return -1;
145     }
146
147     for (size_t i = 0; i < READERS_COUNT; ++i)
148     {
149         readers_threads[i] = CreateThread(NULL, 0, rr_run, (LPVOID)i, 0,
150             NULL);
151         if (readers_threads[i] == NULL)
152         {
153             perror("Failed_call_of_CreateThread");
154             return -1;
155         }
156     }
157
158     for (size_t i = 0; i < WRITERS_COUNT; ++i)
159     {
160         writers_threads[i] = CreateThread(NULL, 0, wr_run, (LPVOID)i, 0,
161             NULL);
162         if (writers_threads[i] == NULL)
163         {
164             perror("Failed_call_of_CreateThread");
165             return -1;
166         }
167     }
168
169     WaitForMultipleObjects(READERS_COUNT, readers_threads, TRUE, INFINITE)
170     ;
171     WaitForMultipleObjects(WRITERS_COUNT, writers_threads, TRUE, INFINITE)
172     ;
173
174     CloseHandle(mutex);
175     CloseHandle(can_read);
176     CloseHandle(can_write);
177
178     return 0;
179 }

```

```
$ ./a
?Reader #0 read: 0 // Idle time: 3702ms
?Reader #1 read: 0 // Idle time: 3705ms
?Reader #2 read: 0 // Idle time: 3709ms
?Reader #3 read: 0 // Idle time: 3712ms
?Reader #4 read: 0 // Idle time: 3715ms
!Writer #0 wrote: 1 // Idle time: 3718ms
!Writer #1 wrote: 2 // Idle time: 3722ms
!Writer #2 wrote: 3 // Idle time: 3725ms
?Reader #1 read: 3 // Idle time: 475ms
!Writer #1 wrote: 4 // Idle time: 682ms
?Reader #3 read: 4 // Idle time: 1204ms
?Reader #0 read: 4 // Idle time: 1727ms
!Writer #0 wrote: 5 // Idle time: 2701ms
?Reader #2 read: 5 // Idle time: 3224ms
!Writer #2 wrote: 6 // Idle time: 3430ms
!Writer #1 wrote: 7 // Idle time: 2967ms
?Reader #0 read: 7 // Idle time: 2086ms
!Writer #0 wrote: 8 // Idle time: 1103ms
?Reader #4 read: 8 // Idle time: 3953ms
?Reader #3 read: 8 // Idle time: 2910ms
?Reader #2 read: 8 // Idle time: 1046ms
?Reader #1 read: 8 // Idle time: 3950ms
!Writer #1 wrote: 9 // Idle time: 1304ms
!Writer #0 wrote: 10 // Idle time: 1241ms
!Writer #0 wrote: 11 // Idle time: 413ms
!Writer #0 wrote: 12 // Idle time: 314ms
?Reader #3 read: 12 // Idle time: 2650ms
?Reader #3 read: 12 // Idle time: 552ms
!Writer #0 wrote: 13 // Idle time: 1592ms
!Writer #2 wrote: 14 // Idle time: 4063ms
?Reader #2 read: 14 // Idle time: 3355ms
!Writer #1 wrote: 15 // Idle time: 2727ms
?Reader #1 read: 15 // Idle time: 3292ms
?Reader #3 read: 15 // Idle time: 461ms
?Reader #0 read: 15 // Idle time: 3996ms
?Reader #4 read: 15 // Idle time: 4007ms
!Writer #2 wrote: 16 // Idle time: 600ms
?Reader #1 read: 16 // Idle time: 690ms
!Writer #0 wrote: 17 // Idle time: 1484ms
?Reader #1 read: 17 // Idle time: 607ms
!Writer #2 wrote: 18 // Idle time: 1042ms
```

Рисунок 1 – Демонстрация работы программы