

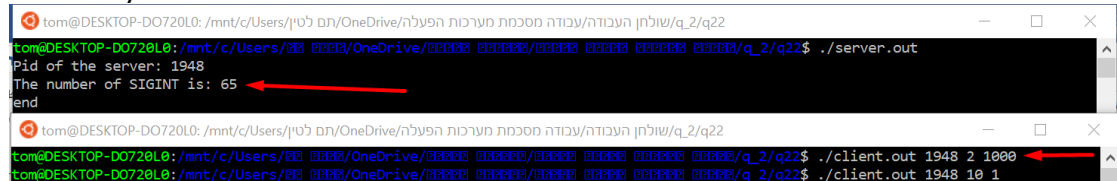
# Final work in operating systems:

## Second question:

### Section 2:

1. Proof that not all signals were received:

Women notice that in the picture you see that I sent 1000 signals and out of them only 65 were received:



```
tom@DESKTOP-DO720L0: /mnt/c/Users/ליטן/OneDrive/הפעלה מערכות מסכמת מעבודה/עבודה/שולחן/q_2/q22
tom@DESKTOP-DO720L0: /mnt/c/Users/ליטן/OneDrive/הפעלה מערכות מסכמת מעבודה/עבודה/שולחן/q_2/q22$ ./server.out
Pid of the server: 1948
The number of SIGINT is: 65
end

tom@DESKTOP-DO720L0: /mnt/c/Users/ליטן/OneDrive/הפעלה מערכות מסכמת מעבודה/עבודה/שולחן/q_2/q22$ ./client.out 1948 2 1000
tom@DESKTOP-DO720L0: /mnt/c/Users/ליטן/OneDrive/הפעלה מערכות מסכמת מעבודה/עבודה/שולחן/q_2/q22$ ./client.out 1948 10 1
```

2. There are two types of signals: standard signals and real-time signals. Each signal has a current disposition, which determines how the process behaves when it is delivered the signal.

There is couple difference between standard signals and real-time signals:

- Multiple instances of real-time signals can be queued. By contrast, if multiple instances of a standard signal are delivered while that signal is currently blocked, then only one instance is queued.
- Order of delivery of real-time signals is guaranteed to be the same as the sending order. In contrast to the standard signal that If multiple standard signals are pending for a process, the order in which the signals are delivered is unspecified.
- If the signal is sent using sigqueue, an accompanying value(either an integer or a pointer) can be sent with the signal. If the receiving process establishes a handler for this signal using the SA\_SIGINFO flag to sigaction, then it can obtain this data via the si\_value field of the siginfo\_t structure passed as the second argument to the handler. Furthermore, the si\_pid and si\_uid fields of this structure can be used to obtain the PID and real user ID of the process sending the signal. This option does not exist in standard signals.

**The differences between the two types of signals are the advantages of real-time signals and the disadvantages of standard signals**

### **Disadvantages of real-time signals:**

- Real-Time signals difficulties in use for application writers.
- The default action for an unhandled real-time signal is to terminate the receiving process.
- Unlike standard signals, real-time signals have no predefined meanings: the entire set of real-time signals can be used for application-defined purposes.
- The number of Real-Time signals sent to a process can grow infinitely. A single socket can send multiple I/O events, resulting in multiple signals in the queue. When the Real-Time signal **queue overflows**, the application must perform complicated steps to recover. The application receives a SIGIO signal when the queue overflows. To resolve the signal overflow, the application must disable Real-Time

signals on all sockets, and remove all signals from the signal queue. Then, to retrieve the lost socket events, the application must poll all the sockets. As expected, handling signal queue overflow increases the complexity of the application, and degrades the performance under high load. Queue overflow has been a major obstacle in using Real-Time signals. A solution to avoid signal queue overflow is to allow only one event per socket file descriptor in the signal queue. If multiple events occur on the same socket, the event flags for the `signinfo_t` data will be combined (using bitwise OR). If the signal queue size is greater than the maximum number of open file descriptors, the queue will never overflow.

#### **Advantages of standard signals:**

- Easier to use than real-time signals
- If both standard and real-time signals are pending for a process, POSIX leaves it unspecified which is delivered first. Linux, like many other implementations, gives priority to standard signals in this case.
- standard signals have predefined meanings, like: SIGKILL, SIGQUIT etc.

#### **Sources for the question:**

<https://man7.org/linux/man-pages/man7/signal.7.html>

<https://pdfs.semanticscholar.org/69ae/2bcd1714a558465cb0264e2796937f362e98.pdf>