**CS604 GDB 2023**

Name: Abdul Rehman.

VU-ID: BC220424444.

**Solution:**

According to the scenario the answer is **One-to-One** model.

The **one-to-one** model provides true concurrency when mapping the user-level threads to kernel-level threads. One-to-one model can also be used to improve performance on multicore systems. In this model, each user-level thread is mapped to a separate kernel-level thread. This allows the kernel to schedule each thread independently, which can lead to true concurrency.

The choice of which multithreading model to use depends on the specific application. For applications that require true concurrency, the one-to-one model is the best choice. For applications that do not require true concurrency, the many-to-one or many-to-many models be a better choice to use.

**Models and Advantages/Disadvantages:**

|  |  |  |
| --- | --- | --- |
| **Model** | **Advantages** | **Disadvantages** |
| **One-to-one** | * Provides true concurrency. * Can be used to improve performance on multicore systems. | * Can be more complex to implement. * Can use more resources than other models. |
| **Many-to-one** | * Simple to implement. * Efficient use of resources. | * Does not provide true concurrency. |
| **Many-to-many** | * Can provide better concurrency than many-to-one model. * Can be used to improve performance on multicore systems. | * More complex to implement. * Can use more resources than other models. |

**Conclusion:**

So, according to the above discussion, the one-to-one model provides true concurrency when mapping user-level threads to kernel-level threads.