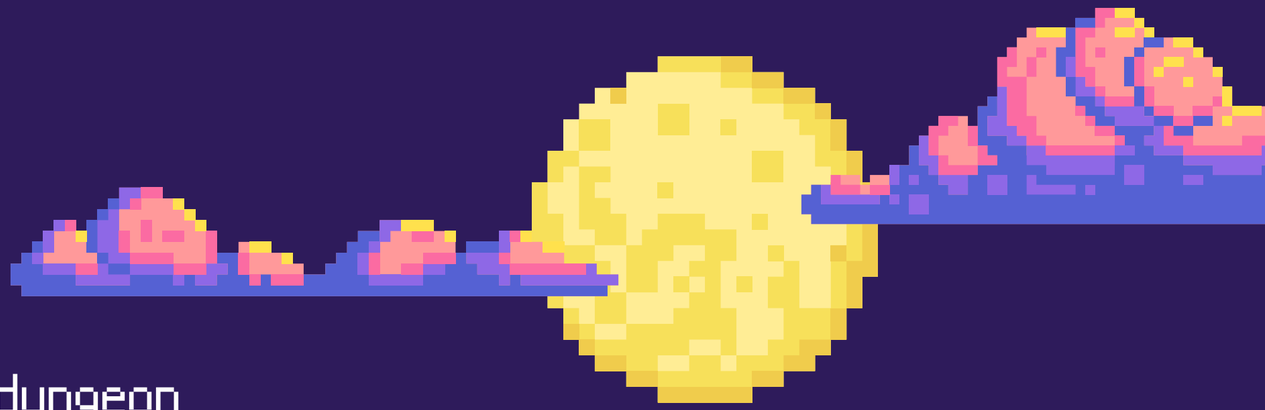
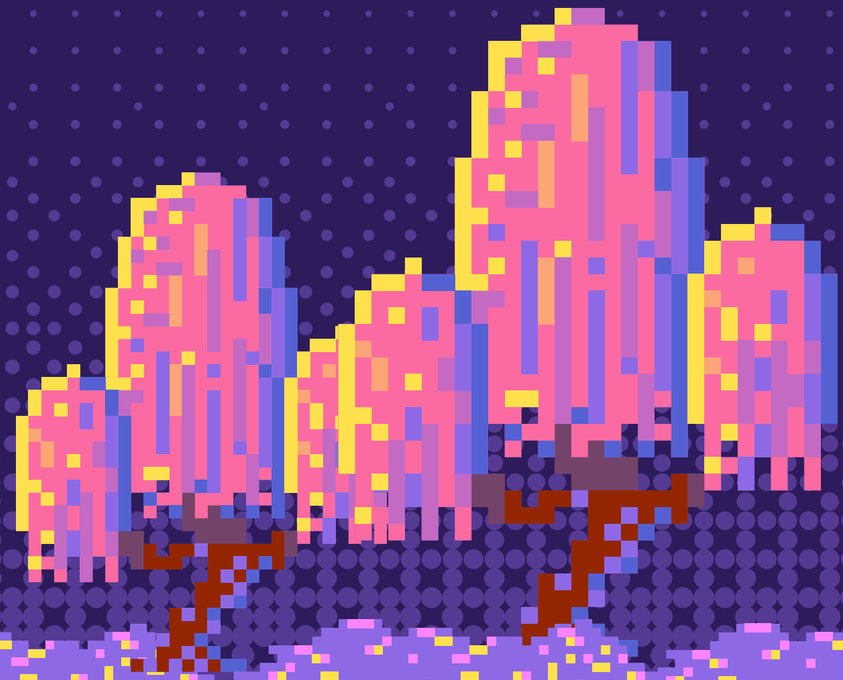

DUNGEON
SIMULATOR
LORD JOHN BENEDICT G. PINPIN

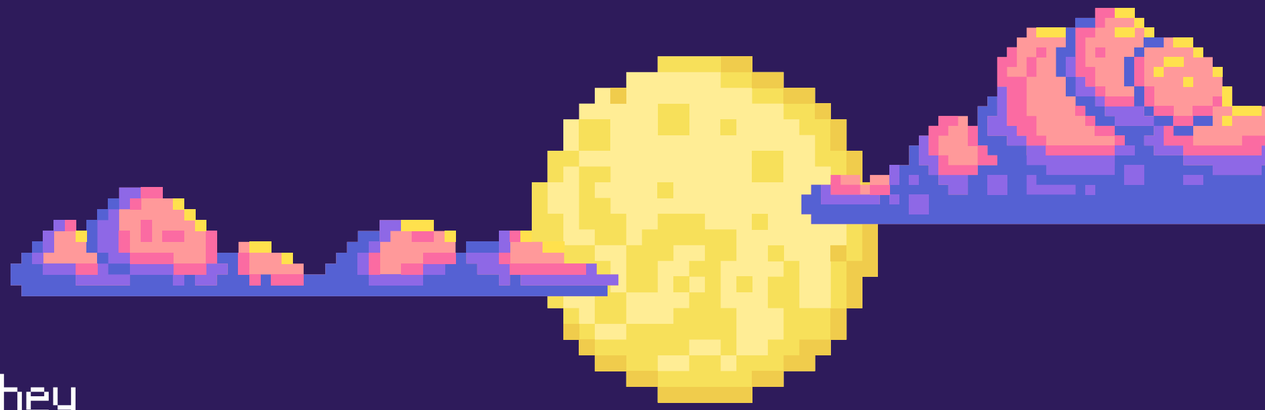
DEADLOCK EXPLANATION



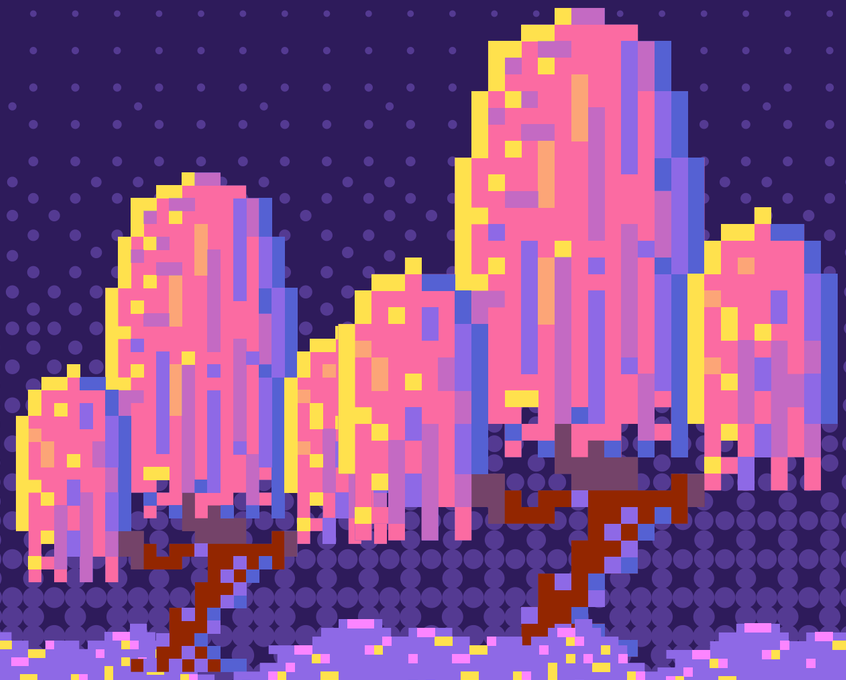
- The instanceSlots semaphore controls access to dungeon instances. If a thread acquires the semaphore but crashes or gets stuck before releasing it, the system could reach deadlock.
- If synchronized methods are used incorrectly, one thread might hold a lock while waiting for another, which is also holding a lock, leading to a circular wait condition.
- Example:
  - Thread 1 locks acquireInstance() while waiting to update servedParties.
  - Thread 2 locks servedParties while waiting to acquire an instance slot.
  - Both threads are stuck, causing a deadlock.
  -



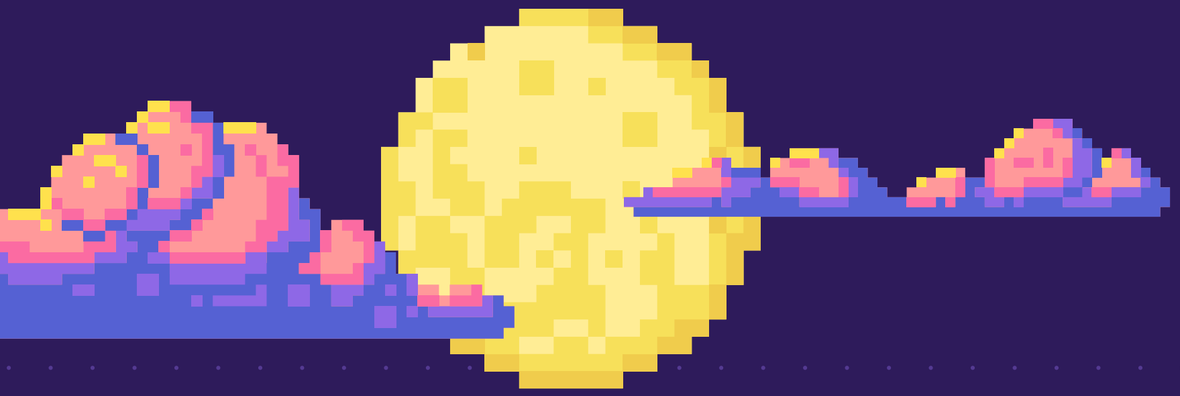
STARVATION EXPLANATION



- If there is a continuous influx of new parties and they always get instance slots first, older parties might wait indefinitely.
- If Thread.sleep() is used without proper thread prioritization, some parties could take significantly longer, delaying others.
- Example:
  - A thread keeps acquiring instances quickly and frequently, preventing other threads from entering the queue.
  - Players who joined earlier might not get a chance if newly created parties continuously take precedence.
  -



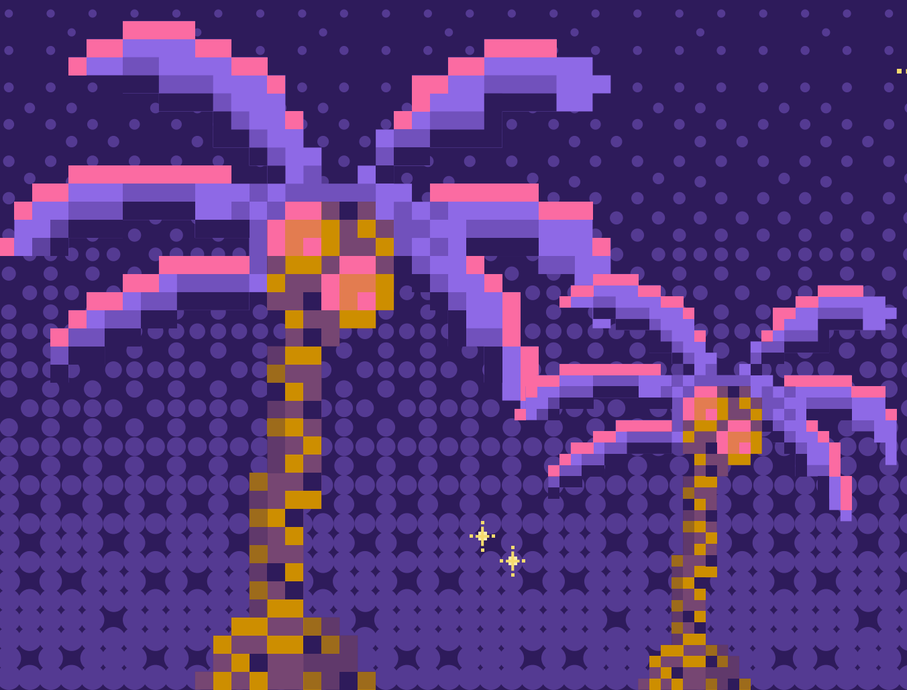
SYNCHRONIZATION MECHANISMS

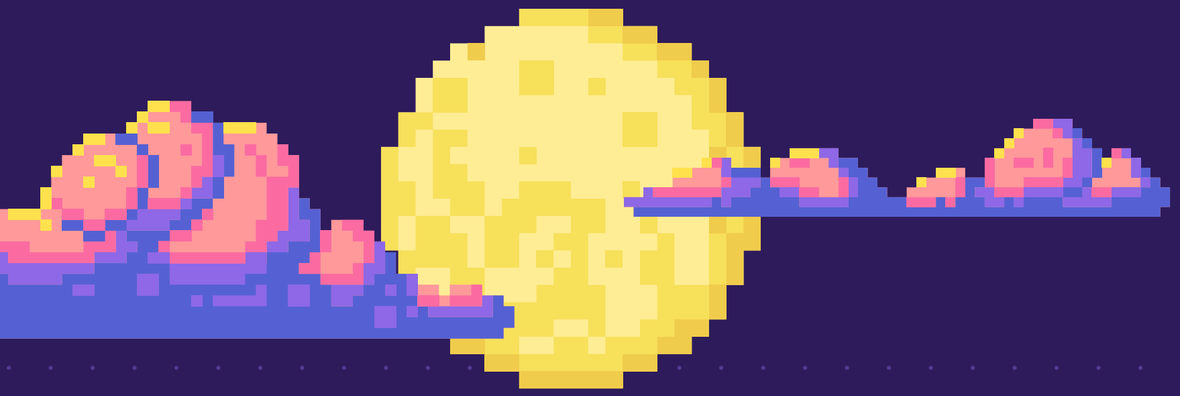


01

Semaphores (instanceSlots):

- Limits the number of concurrent dungeon instances.
- Prevents too many threads from executing at once.
- Ensures fair access by requiring each thread to wait for an available slot.





SYNCHRONIZATION MECHANISMS

01

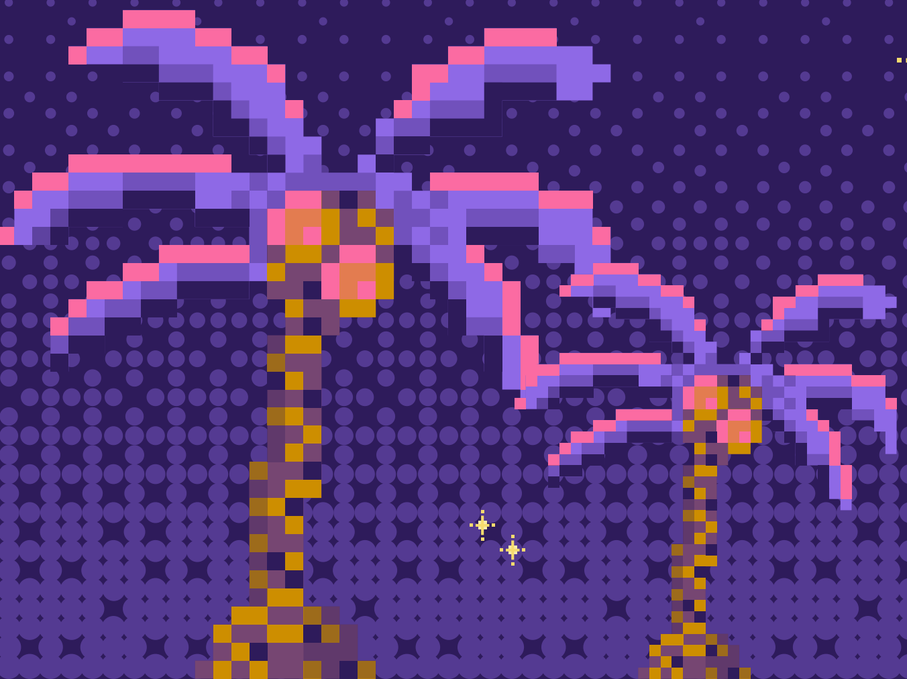
Semaphores (instanceSlots):

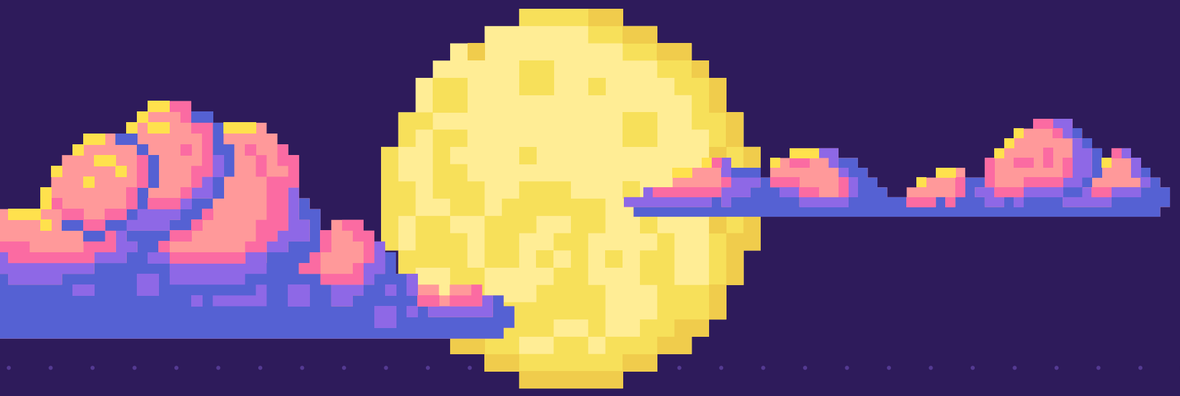
- Limits the number of concurrent dungeon instances/threads.
- Ensures fair access by requiring each thread to wait for an available slot.

02

Synchronized Methods and Blocks:

- synchronized (this): Ensures mutual exclusion when updating shared resources like servedParties and instanceStatus.
- Protects acquireInstance() and printStatus() to avoid race conditions.
- Prevents multiple threads from modifying the instance array simultaneously.





SYNCHRONIZATION MECHANISMS

01

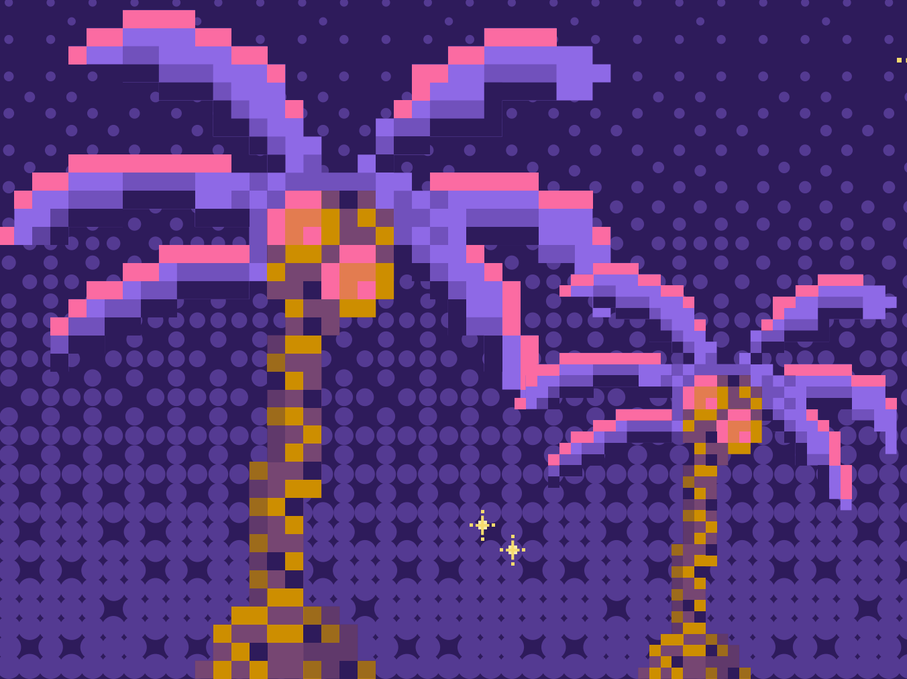
Semaphores (instanceSlots):

- Limits the number of concurrent dungeon instances/threads.
- Ensures fair access by requiring each thread to wait for an available slot.

02

Synchronized Methods and Blocks:

- synchronized (this): Ensures mutual exclusion when updating shared resources like servedParties and instanceStatus.
- Protects acquireInstance() and printStatus() to avoid race conditions.
- Prevents multiple threads from modifying the instance array simultaneously.




THANK YOU