

Operating Systems

Course Code: 71203002004

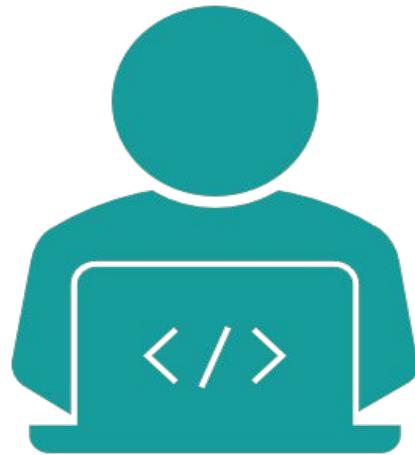
by -
Asst. Prof. Minal Rajwar



Learning Objectives

Define	Define a simple batch system
Explain	Explain how it evolved historically
Describe	Describe system architecture with diagrams
Understand	Understand job processing with the Resident Monitor
Evaluate	Evaluate its pros and cons

Introduction to Batch Systems



□ Serial Processing

- With the earliest computers, from the late 1940s to the mid-1950s, the programmer interacted directly with the computer hardware; there was no OS.
- These computers were run from a console consisting of display lights, toggle switches, some form of input device, and a printer. Programs in machine code were loaded via the input device (e.g., a card reader). If an error halted the program, the error condition was indicated by the lights.
- If the program proceeded to a normal completion, the output appeared on the printer. This mode of operation could be termed serial processing, reflecting the fact that users have access to the computer in series.

THE EVOLUTION OF OPERATING SYSTEMS

□ Simple Batch Systems

- Early computers were very expensive, and therefore it was important to maximize processor utilization. To improve utilization, the concept of a batch operating system was developed.
- It appears that the first batch operating system was developed in the mid-1950s by General Motors for use on an IBM 701.
- The concept was subsequently refined and implemented on the IBM 704 by a number of IBM customers. By the early 1960s, a number of vendors had developed batch operating systems for their computer systems. IBSYS, the IBM operating system for the 7090/7094 computers, is particularly notable because of its widespread influence on other systems.

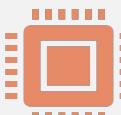
Need for Batch Systems



Without automation,
operators had to manually
load programs & data.



This led to **wasted CPU
time** while jobs were
loaded



Batch processing
increased efficiency
through automatic job
sequencing.

Basic Working of a Batch System

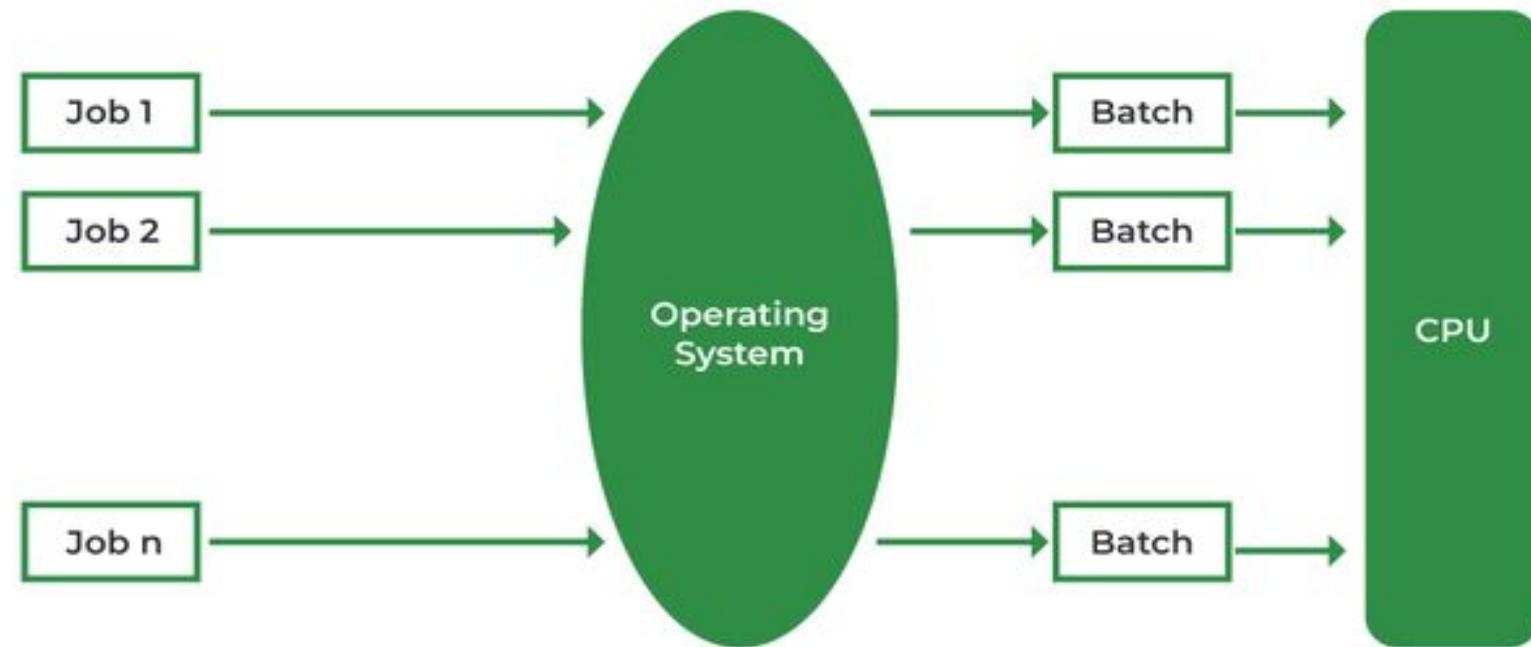
Users submit jobs to an **operator** who collects them.

Jobs are grouped into a batch and loaded together.

A program called the **Resident Monitor** manages execution.

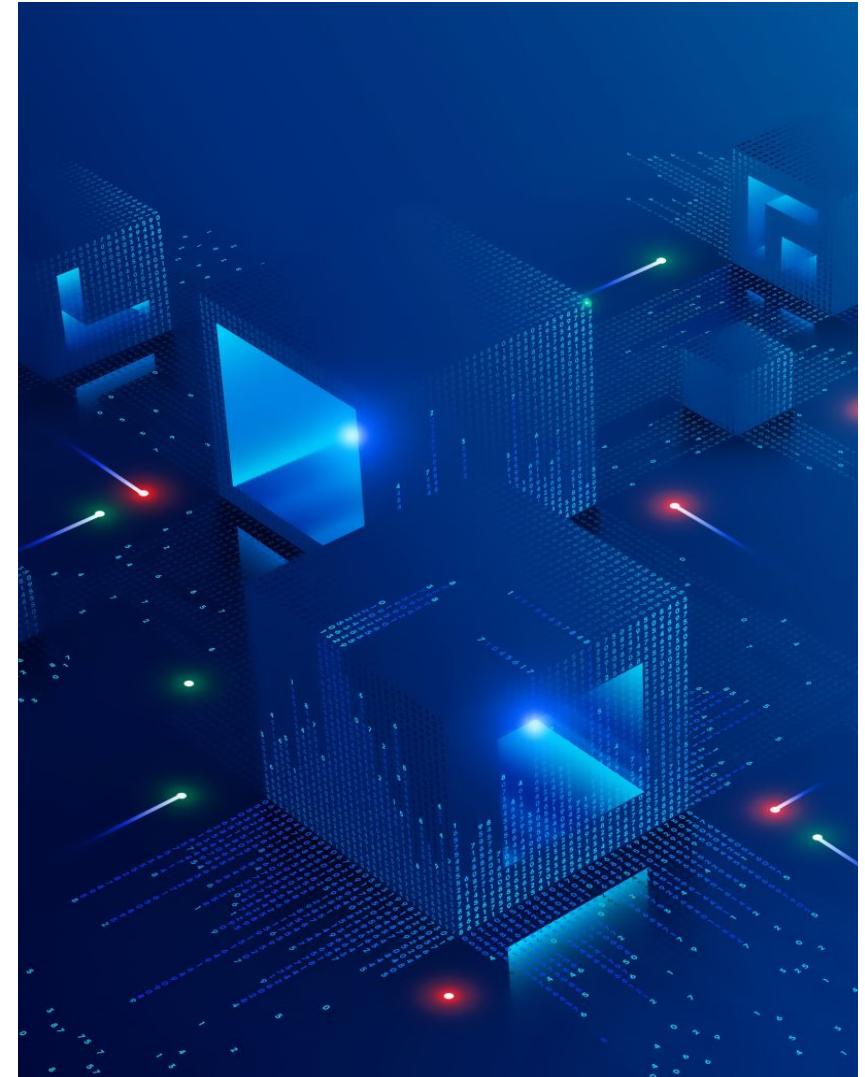
Users receive output later; **no interaction** happens during processing.

Simple Batch System Architecture



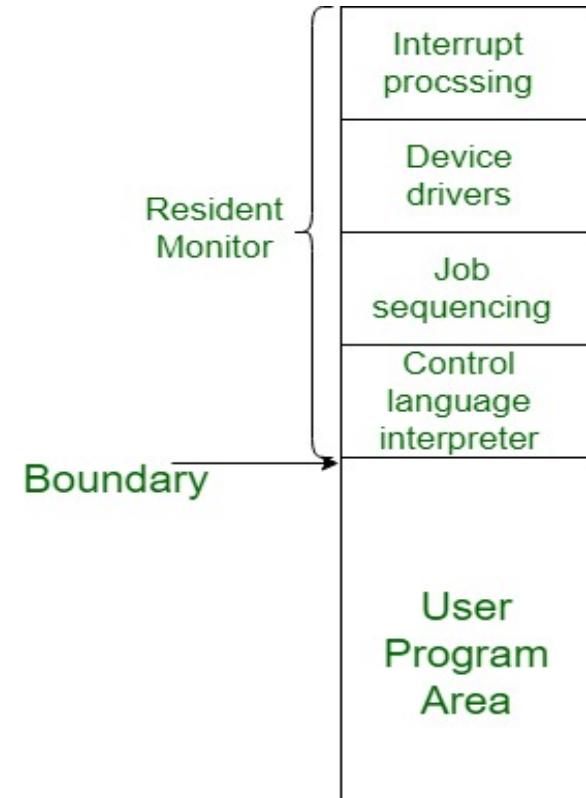
Simple Batch System Architecture

- Jobs enter via input device (card reader, disk, etc.)
- Stored in a **job queue** (on secondary storage)
- Monitor loads and executes jobs one by one
- Output sent to printer or output disk
- This architecture increases CPU utilization and minimizes idle time

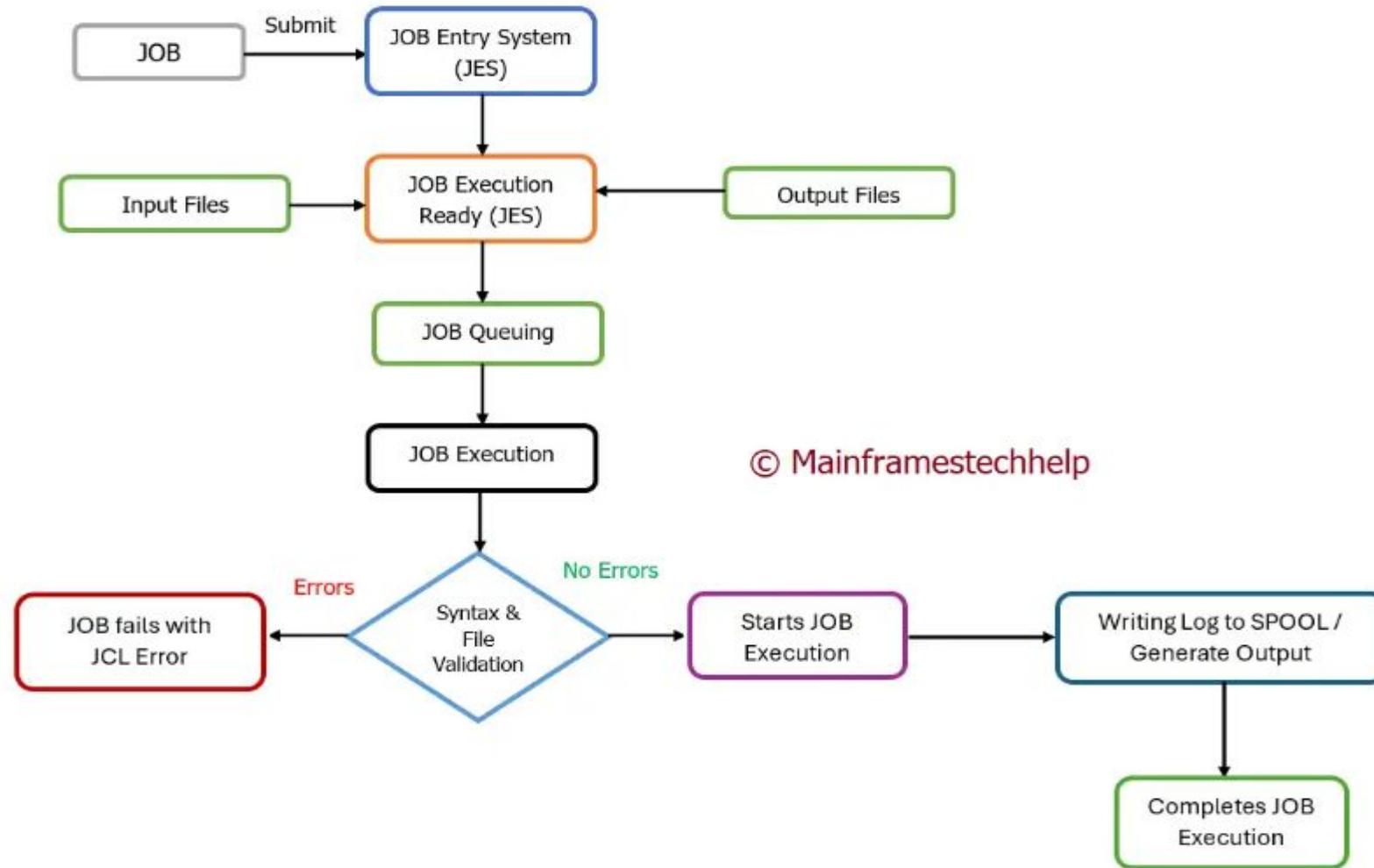


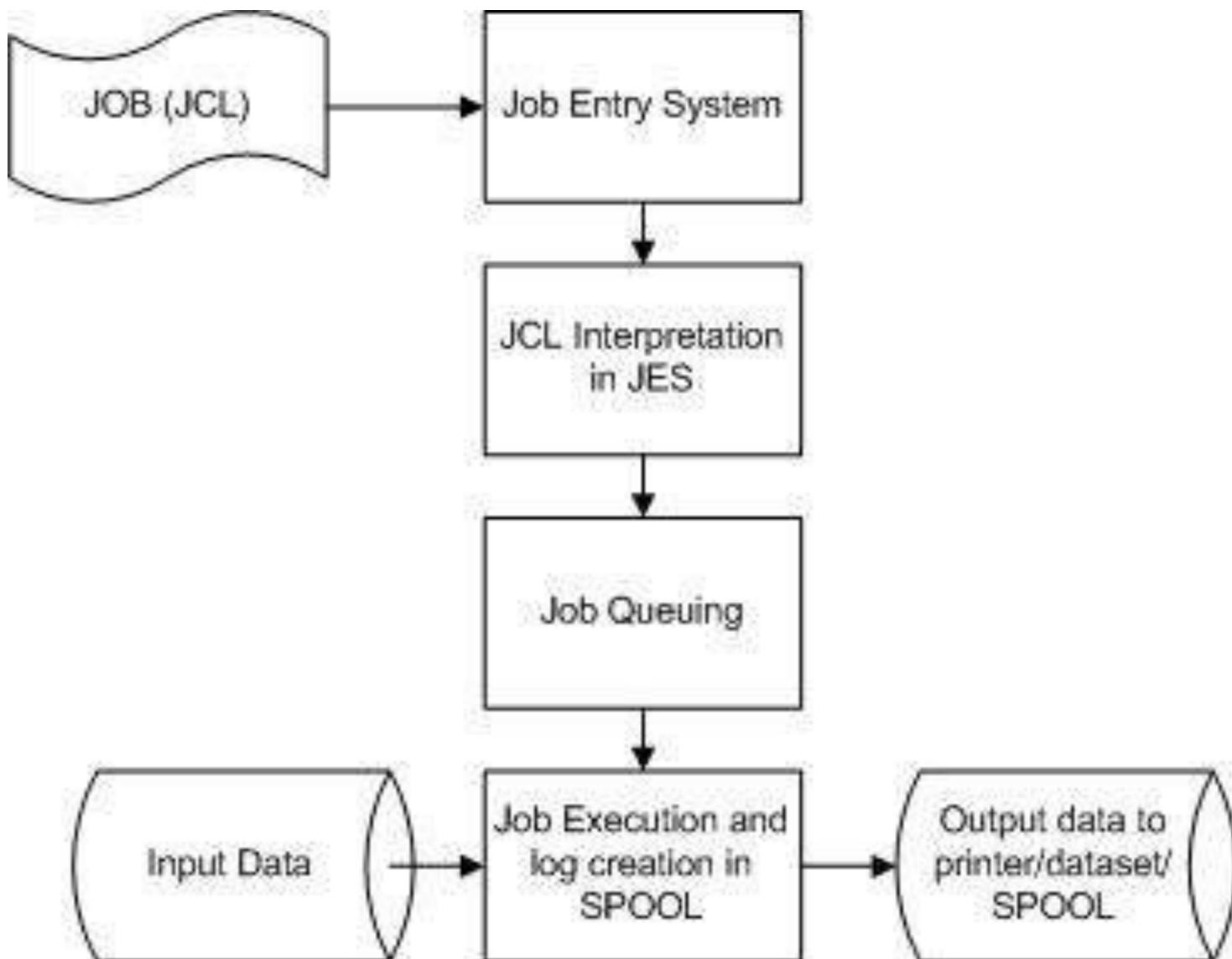
Resident Monitor

- **Resident Monitor** remains permanently in low memory.
- Loads user programs from disk into memory
- Once a job finishes, it returns control to the monitor
- Monitor then loads and starts the next job

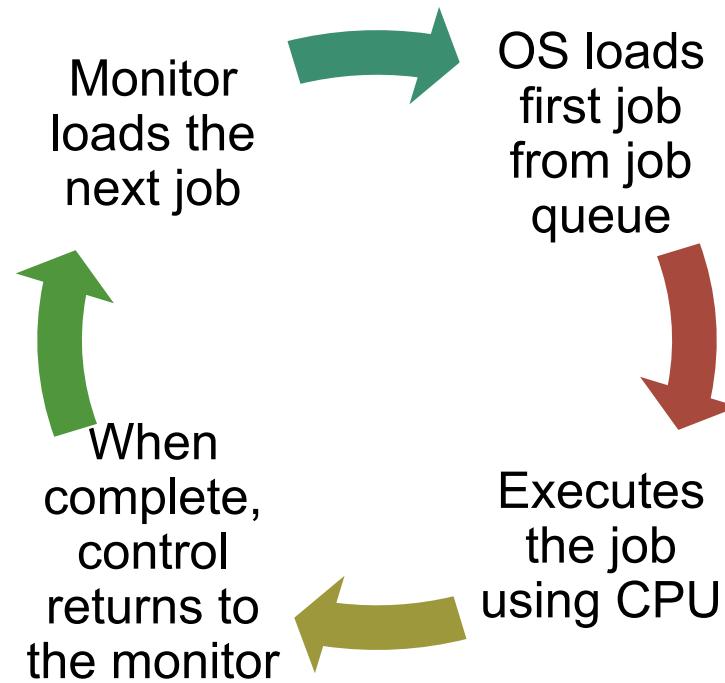


Job Execution Process





Job Execution Process



Features of Simple Batch Systems

Sequential execution of jobs

Uses Job Control Language (JCL)
for job instructions

No interaction during job execution

Fixed memory partitioning

**Monitors handle job loading and
unloading**

Advantages of Batch Systems



Reduces CPU idle time



Automates job sequencing

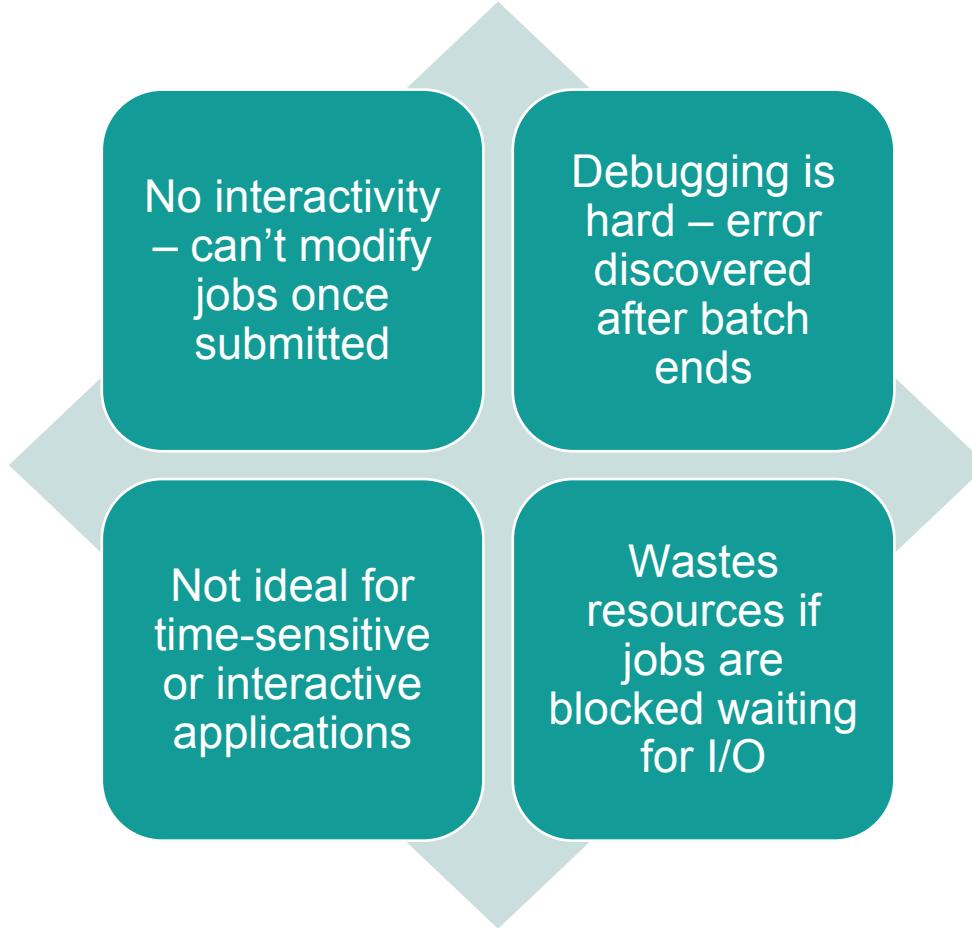


Suitable for repetitive tasks (e.g., billing, payroll)



Efficient for large-scale data processing

Limitations of Batch Systems



No interactivity – can't modify jobs once submitted

Debugging is hard – error discovered after batch ends

Not ideal for time-sensitive or interactive applications

Wastes resources if jobs are blocked waiting for I/O

Q&A / Interactive Quiz

What is the role of the resident monitor?

Why can't we use batch systems for real-time applications?

What hardware device was used for input in early batch systems?

List one advantage and one limitation of batch systems.