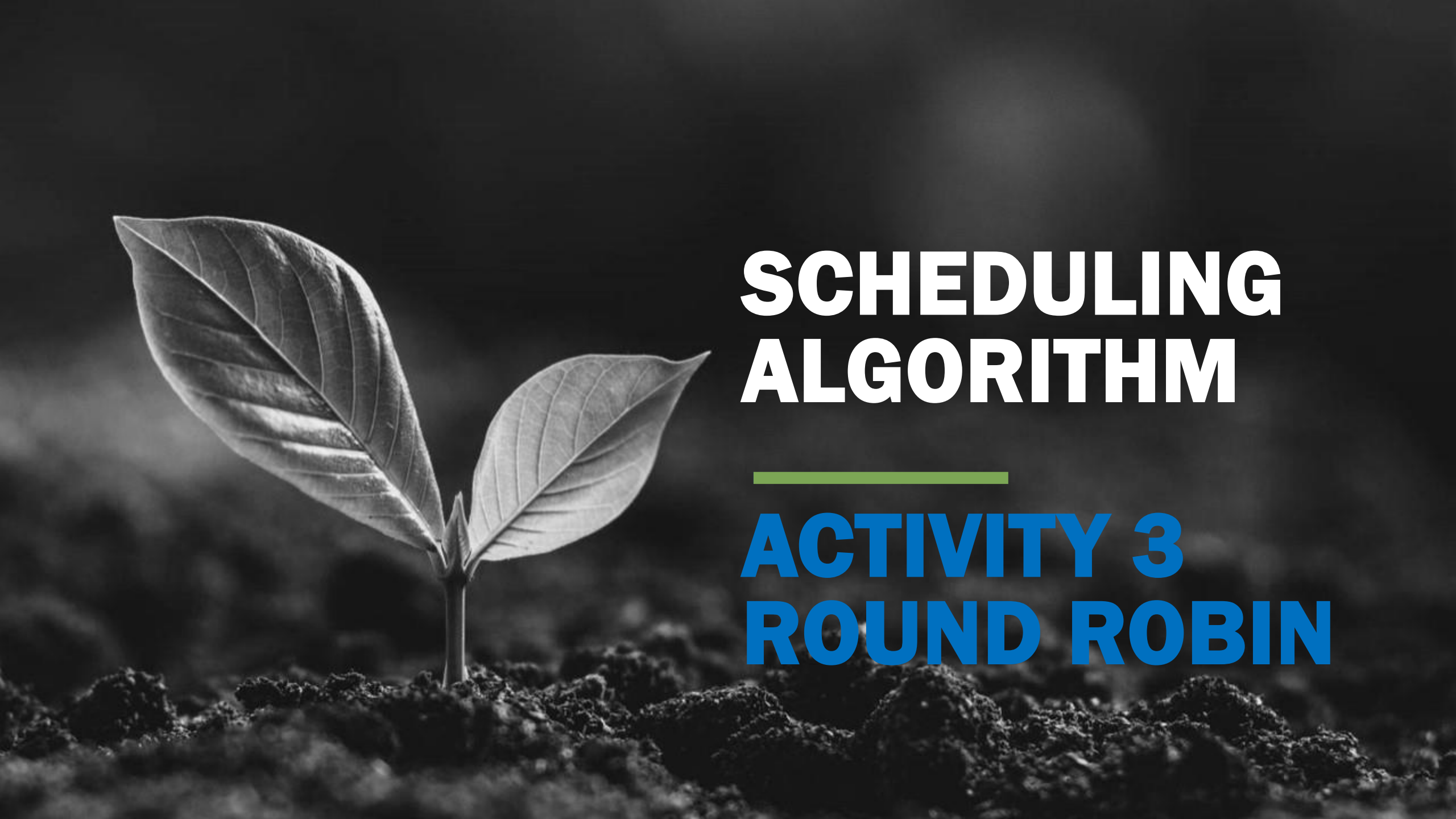
SCHEDULING ALGORITHM

ROUND ROBIN



SCHEDULING ALGORITHM

ACTIVITY 3
ROUND ROBIN

RR ALGORITHM

- **Round Robin (RR) Scheduling** is a **CPU scheduling algorithm** commonly used in **time-sharing systems**. The algorithm works by allocating a fixed time slice or **quantum** to each process in the ready queue. When a process's time quantum expires, the CPU is preempted and the next process in the queue gets a turn.

How Round Robin Scheduling Works:

1. **Process Queue**:
    1. All processes are placed in a **queue**. When the CPU becomes idle, the first process in the queue is selected to use the CPU.
2. **Time Quantum (or Time Slice)**:
    1. Each process is assigned a fixed **time slice** or **quantum** (e.g., 10 ms). The process will run for the duration of the time quantum, or until it completes its execution, whichever comes first.

RR ALGORITHM

- **Context Switching**:
- When the time quantum expires, the **CPU is switched** to the next process in the queue, regardless of whether the current process is finished. The current process is then put back at the end of the queue.

- This is called a **context switch**, and the process continues its execution in the next round if it hasn't completed.

- **Fairness**:
- All processes receive an equal opportunity to execute, which means **Round Robin** is considered **fair** because each process gets a turn to use the CPU.

RR ALGORITHM

Preemption:

- **Round Robin** is a **preemptive scheduling algorithm**, meaning a process can be interrupted and replaced by another process before it finishes, based on the time quantum.
- **Time Quantum** is critical to Round Robin's performance. A smaller time quantum results in more context switching, leading to inefficiency. A larger quantum can lead to longer wait times for other processes.
- **Fairness:** Every process gets an equal share of CPU time.
- **Preemption:** If a process doesn't finish within its time slice, it’s preempted and placed back in the queue.
- **CPU Utilization:** As long as there are processes in the queue, the CPU is utilized.

RR ALGORITHM

Consider the set of 6 processes whose arrival time and burst time are given below: **Time Quantum (TQ) = 4 units**

PROCESS	AT	BT
P1	0	5
P2	1	6
P3	2	3
P4	3	1
P5	4	5
P6	6	4

Queueing:

Time 0: P1 P2 P3 P4 P5 → P1 (5-4 = 1 units left BT)
Time 4: P2 P3 P4 P5 P1 P6 → P2 (6-4=2 units left BT)
Time 8: P3 P4 P5 P1 P6 P2 → P3 (3-0 units left BT 3 qt used)
Time 11: P4 P5 P1 P6 P2 → P4 (1-0 units left BT 1 qt used)
Time 12: P5 P1 P6 P2 → P5 (5-4=1 unit BT)
Time 16: P1 P6 P2 P5 → P1 (1-0 units BT 1 qt used)
Time 17: P6 P2 P5 → P6 (4-4=0 units BT)
Time 21: P2 P5 → P2 (0 units BT 2 qt used)
Time 23: P5 → P5 (1-0=0 units BT 1 qt used)

RR ALGORITHM

Gantt Chart

Time Units →	**0**	**4**	**8**	**11**	**12**	**16**	**17**	**21**	**23**	**24**
Process →	P1	P2	P3	P4	P5	P1	P6	P2	P5	
Execution	4	4	3	1	4	1	4	2	1	

ACTIVITY 3

RR ALGORITHM

ACTIVITY 3: RR ALGORITHM

INSTRUCTIONS

1. Make a Java program that will compute for the following.
   a. Turn Around Time = Completion Time – Arrival Time.
   b. Waiting Time = Turn Around Time – Burst Time
   c. Average Turn Around Time.
   d. Average Waiting Time.
   e. Gantt Chart.
2. The user will have the choice to input 3 to 5 process, including Arrival Time and Burst time.
3. The program will ask the user if he/she wants to try again?
4. Upload the java file using the filename format: SNFN#.java
5. Upload a sample output of the program.
6. Java file and sample output will be uploaded at teams.

Criteria: 50 points

1. Complete and running program no errors. **50 points**.
2. Incomplete but running program no errors. **35 points**.
3. Running but no correct output **15 points**. (for the effort)
4. 2nd checking must be complete and working no errors **20 points**.
5. Programs to be checked should be error free. No checking of program if error persists.
6. Once I checked the program it is recorded. If you request for 2nd checking you be in the last priority.

PROGRAM OUTPUT

ACTIVITY 3:
RR ALGORITHM

ACTIVITY 3: FCFS ALGORITHM

Title: RR SCHEDULING ALGORITHM

Enter no. of process: (IN)

Quantum Time: (IN)

Process	AT	BT	CT	TAT	WT
P1	(IN)	(IN)	(O)	(O)	(O)
P2	(IN)	(IN)	(O)	(O)	(O)
P3	(IN)	(IN)	(O)	(O)	(O)
P4	(IN)	(IN)	(O)	(O)	(O)
P5	(IN)	(IN)	(O)	(O)	(O)

Average TAT: (O)

Average WT: (O)

Gannt Chart: (Example)

| P1 | P2 | P3 | P4 | P5 |

0 4 7 8 10 14

EOA

SAMPLE OUTPUT FOR CHECKING PURPOSE ONLY

Activity 3: Round Robin Algorithm

Quantum Time: 4

Process	Arrival Time (AT)	Burst Time (BT)	Completion Time (CT)	Turnaround Time (TAT)	Waiting Time (WT)
P1	0	5	17	17	12
P2	1	6	23	22	16
P3	2	3	11	9	6
P4	3	1	12	9	8
P5	4	5	24	20	15

AVERAGE TAT = 11.33
AVERAGE WT = 15.33

Gantt Chart: (Example)

Time Units →	0		4		8		11		12		16		17		21		23		24
Process →		P1		P2		P3		P4		P5		P1		P6		P2		P5	
Execution		4		4		3		1		4		1		4		2		1	