For your interview, focus on these **key shell scripting topics** commonly used in DevOps, especially for automating tasks in CI/CD pipelines:

Topics to Focus On:

1. **File Operations**:
    - Reading, writing, and manipulating files (log files, configuration files, etc.).
    - File permissions and management (chmod, chown, etc.).
2. **Text Processing**:
    - Using tools like grep, sed, awk, and cut to parse, extract, and manipulate text (log files, CSV files, config files, etc.).
3. **Process Management**:
    - Managing and monitoring processes using ps, top, kill, and nohup.
4. **Automation**:
    - Automating tasks like package installations, deployments, service restarts, and monitoring.
    - Using cron for scheduling recurring jobs.
5. **System Monitoring and Alerts**:
    - Monitoring disk usage, CPU usage, memory consumption, and automating alerts using shell scripts.
6. **Networking**:
    - Working with ports, IP addresses, and protocols, especially using commands like netstat, ifconfig, and curl.
7. **Exit Codes and Error Handling**:
    - Using if, else, trap, and exit codes to handle errors and take actions accordingly.
8. **Database Interaction**:
    - Connecting to databases (e.g., MySQL, PostgreSQL) via shell scripts and performing backups, query executions, or health checks.
9. **Environment Variables**:
    - Using and manipulating environment variables inside scripts for configuration and sensitive data handling.
10. **CI/CD Pipeline Automation**:

- Automating stages of the pipeline like code builds, testing, deployment, and post-deployment monitoring using shell scripts.

Scenario-based Shell Scripts

1. **Logs & Password Rotation Script**: This script rotates logs and changes passwords for users periodically.

```
#!/bin/bash
# Log rotation
LOG_DIR="/var/log/myapp/"
ARCHIVE_DIR="/var/log/archive/"

mkdir -p $ARCHIVE_DIR

# Compress and move old logs
find $LOG_DIR -type f -name "*.log" -mtime +7 -exec gzip {} \;
mv $LOG_DIR/*.gz $ARCHIVE_DIR

# Password rotation (for local users)
for user in $(cat /etc/passwd | grep /home | cut -d':' -f1)
do
  echo "Rotating password for user: $user"
  NEW_PASS=$(openssl rand -base64 12)
  echo "$user:$NEW_PASS" | chpasswd
  echo "Password for $user has been changed"
done
```

2. **Automate Package Installation & Deployment**: This script installs packages and deploys an application.

```
#!/bin/bash
# Install packages
echo "Installing necessary packages..."
sudo apt-get update -y
sudo apt-get install -y nginx docker docker-compose

# Deploy app (docker-compose)
echo "Deploying the application..."
cd /path/to/app
docker-compose up -d
echo "Application deployed!"
```

3. **Parsing CSV Files**: Script to parse CSV and extract specific data.

```
#!/bin/bash
# Parsing CSV file
INPUT_FILE="data.csv"
IFS=','
```

```
while read -r id name email; do
  echo "User ID: $id"
  echo "Name: $name"
  echo "Email: $email"
done < $INPUT_FILE
```

4. **Connect to RDS-MySQL and Monitor Replica Lag**: This script connects to an AWS RDS MySQL instance, checks the replica lag, and sends an alert using AWS SES if the threshold is breached.

```
#!/bin/bash
# RDS MySQL Replica Lag Monitor
HOST="your-rds-endpoint"
USER="admin"
PASS="yourpassword"
DB="yourdatabase"
THRESHOLD=300
AWS_REGION="us-east-1"
SES_EMAIL="notify@example.com"
STAKEHOLDERS="stakeholder1@example.com, stakeholder2@example.com"

# Get replica lag
REPLICA_LAG=$(mysql -h $HOST -u $USER -p$PASS -D $DB -sse "SHOW SLAVE STATUS\G" | grep
'Seconds_Behind_Master' | awk '{print $2}')

if [ "$REPLICA_LAG" -ge "$THRESHOLD" ]; then
  echo "Replica lag is $REPLICA_LAG seconds, sending alert..."
  MESSAGE="Replica lag of $REPLICA_LAG seconds has exceeded the threshold of $THRESHOLD."
  aws ses send-email \
    --region $AWS_REGION \
    --from $SES_EMAIL \
    --destination "ToAddresses=$STAKEHOLDERS" \
    --message "Subject={Data=RDS Replica Lag Alert},Body={Text={Data=$MESSAGE}}"
else
  echo "Replica lag is under control: $REPLICA_LAG seconds."
Fi
```

This script can be scheduled to run every 15 minutes using a **crontab** entry like this:

```
*/15 * * * * /path/to/replica_monitor.sh
```

5. **Automating Backup and Sync to S3**: This script creates a backup and syncs it to an AWS S3 bucket.

```
#!/bin/bash
BACKUP_DIR="/backups"
S3_BUCKET="s3://your-bucket-name/"
TIMESTAMP=$(date +"%F")
BACKUP_FILE="backup-$TIMESTAMP.tar.gz"

# Create a backup of /var/www
tar -czf $BACKUP_DIR/$BACKUP_FILE /var/www

# Sync the backup to S3
aws s3 cp $BACKUP_DIR/$BACKUP_FILE $S3_BUCKET

echo "Backup complete and uploaded to S3."
```

Real-time DevOps CI/CD Shell Scenarios:

1. **Trigger Builds and Deployment**: Automate build and deployment stages by pulling code from Git, building the codebase, running tests, and deploying via shell scripts.
2. **System Monitoring**: Regularly check system resource usage (CPU, RAM, Disk) and send alerts if usage crosses a certain threshold.
3. **Server Health Check**: Periodically ping services or APIs running on your servers and alert if a service is down.