## 1. Ansible

### Loops

- `loop:` — generic loop over items
- `with_items:` — older syntax, still works
- `with_dict:` — loop over dictionary items
- `with_list:` — loop over lists
- `with_nested:` — nested loops
- `with_fileglob:` — loop over files matching glob
- `until:` — retry loop until a condition is met
- `loop_control:` — for controlling loop labels, index, etc.

### Conditionals

- `when:` — execute task conditionally
- `failed_when:` — mark task failed based on condition
- `changed_when:` — mark task changed based on condition

### Handlers

- `handlers:` — define special tasks that only run when notified
- `notify:` — trigger a handler

### Notifications / Alerts

- Mostly via handlers (email, Slack, custom modules)
- External modules: `mail`, `slack`, `pagerduty`, `opsgenie`

### Keywords / Modules

- Common modules:
  - `file`, `copy`, `template`, `yum`, `apt`, `service`, `command`, `shell`, `git`, `user`, `cron`, `docker_container`, `docker_image`

- Keywords:

    - `hosts`, `tasks`, `vars`, `register`, `become`, `tags`, `delegate_to`, `ignore_errors`, `include_tasks`, `import_tasks`

## 2. Jenkins (Declarative Pipeline / Groovy)

### Loops

- `for`, `while` — standard Groovy loops
- `each`, `eachWithIndex` — Groovy collection iterators

### Conditionals

- `if`, `else`, `elseif` — standard Groovy
- `when` — pipeline declarative conditionals

### Notifications

- `emailext` — email notifications
- `slackSend` — Slack notifications
- `mail` — built-in email
- `step([$class: …])` — can call custom notifications

### Handlers / Triggers

- `triggers {}` — cron, GitHub webhook, SCM polling
- `post {}` — defines steps for `always`, `success`, `failure`, `unstable`, `changed`

## 3. Terraform

### Loops

- `for_each` — iterate over map or list
- `count` — create multiple resources
- `for` expressions — inline looping for variables
- `dynamic` blocks — loop inside resource blocks

### Conditionals

- `count = condition ? 1 : 0` — create resource conditionally
- `if`, `else` in expressions
- `lookup()` with defaults for conditional data

### Notifications

- Not native; usually via integrations (OpsGenie, Slack, AWS SNS, etc.)

## 4. Kubernetes (YAML)

### Loops & Conditionals

- Not natively supported in YAML — use Helm or Kustomize:
  - Helm templates: `{{ range .Values.items }}` … `{{ end }}`
  - `if`, `else`, `with` — conditional rendering in Helm

### Notifications

- Not native; handled externally:
  - Prometheus Alerts → Alertmanager → Slack/email/PagerDuty

### Handlers

- `postStart`, `preStop` hooks in pod spec

## 5. Python / Shell in DevOps Scripts

### Loops

- `for`, `while`
- `for index, item in enumerate(list):`
- List comprehensions: `[x for x in list if x%2==0]`

### Conditionals

- `if`, `elif`, `else`

- Exception handling: `try/except/finally`

**Notifications**

- `smtplib` for email
- `requests` to hit Slack webhook / Teams
- Custom logging: `logging` module

## 6. Common DevOps Concepts Across Tools

| Concept | Tools / Keywords |
|---|---|
| Loop | `loop`, `for_each`, `count`, `for`, `while`, `range` |
| Conditional | `when`, `if`, `else`, `elseif`, ternary operators |
| Handler / Trigger | `handlers`, `notify`, `post {}`, `triggers`, hooks |
| Notification | `slackSend`, `mail`, `emailext`, `smtplib`, `pagerduty` |
| Retry / Wait | `until`, `retries`, `delay`, `wait_for`, `sleep` |
| Include / Import | `include_tasks`, `import_tasks`, `include_role`, `source` |
| Variable / Register | `vars`, `set_fact`, `register`, `env`, `output`, `lookup` |

## DevOps Mega Cheat Sheet

| Concept | Ansible | Jenkins (Pipeline/ Groovy) | Terraform | Kubernetes / Helm | Python / Shell | Notes / Example Use |
|---|---|---|---|---|---|---|
| Loop | loop:<br>with_items:<br>with_dict:<br>with_nested:<br>with_fileglob:<br>until:<br>loop_control: | for, while<br>each, eachWithIndex | count<br>for_each<br>for expressions<br>dynamic blocks | {{ range .Values.items }} ... {{ end }} | for, while<br>List comprehensions<br>[x for x in list] | Repeat tasks or resources, iterate collections |
| Conditional | when:<br>failed_when:<br>changed_when: | if, else, elseif<br>when (pipeline declarative) | condition ? true : false<br>count = condition ? 1 : 0 | {{ if .Values.enableFeature }} ... {{ end }} | if, elif, else<br>try/except/finally | Execute tasks/ resources only if conditions are met |
| Handler / Trigger | handlers:<br>notify: | post {}<br>triggers {} | Not native; external triggers via webhooks | postStart, preStop hooks | Signal handling:<br>signal module<br>trap in Shell | Actions triggered by events, state changes, or notifications |
| Notification | mail, slack, pagerduty, opsgenie modules | mail, emailext, slackSend | Usually external integrations (AWS SNS, Slack webhook) | External: Prometheus → Alertmanager → Slack/email | smtplib, requests to hit webhook, logging | Alerts for success, failure, or monitoring |
| Retry / Wait | until:, retries:, delay: | retry, timeout steps | time_sleep in null_resource<br>depends_on logic | Helm: wait: true on deployment | time.sleep(), retry loops, backoff | Retry tasks/resources until success |
| Include / Import | include_tasks, import_tasks, include_role | load libraries / shared scripts | module, terraform_remote_state | Helm: include, required templates | import, source in Shell | Reuse tasks, templates, modules |
| Variables / Register | vars, set_fact, register | env, params, currentBuild | variable, output, locals | Helm: .Values.variable, .Release.Name | Python: variables, dicts<br>Shell: $VAR | Store and use data dynamically |
| File / Resource Management | file, copy, template, yum, apt, service, git, docker_container | sh, checkout scm, archiveArtifacts | resource, provider, data, output | Kubernetes: Deployment, Service, ConfigMap | Python: open(), os, shutil<br>Shell: cp, mv, mkdir | Create/manage files, directories, resources |
| Conditionally Changed / Failed | changed_when, failed_when | unstable, failure, success in post {} | count = condition ? 1 : 0 | Not native; use Helm templates | Python: raise Exception | Mark status based on custom conditions |
| Tags / Grouping | tags, roles | stage, parallel, matrix | module, for_each | Helm: labels, selectors | Python: functions, classes<br>Shell: scripts | Organize tasks/ stages/resources logically |