Tarea 5 Enunciado: Prácticas con filtros

Ejercicio 1: Diseña una tarea en Ansible que trabaje con un diccionario definido en las variables del playbook.

- Deberá mostrar el contenido completo del diccionario.
- Transforma ese diccionario en una lista de pares clave-valor utilizando el filtro dict2items y muéstralo por pantalla.
- Recorrer esa lista con un bucle para imprimir un mensaje que indique la clave y el valor de cada elemento.

Ejemplo de variable entorno.

```
 entorno:
      proyecto: "Curso de Ansible"
      responsable: "Martín"
      personas: 20
      sistema: "Linux"
```

```
---
- name: Práctica de filtros
  hosts: srv1
  become: yes
  vars:
    entorno:
      proyecto: "Curso de Ansible"
      responsable: "Martín"
      personas: 20
      sistema: "Linux"

  tasks:
    - name: Mostrar datos de la variable
      ansible.builtin.debug:
        var: entorno

    - name: Mostrar datos del diccionario
      ansible.builtin.debug:
        msg: "{{ entorno | dict2items }}"

    - name: Bucle para recorrer el diccionario
      ansible.builtin.debug:
        msg: "La key es: {{ item.key }} y el valor es: {{ item.value }}"
      loop: "{{ entorno | dict2items }}"
```

Ejercicio 2: Realiza un playbook en Ansible que devuelva los usuarios del sistema que contienen la palabra "user" en el archivo /etc/passwd. Primero, ejecuta un comando para filtrar estas líneas con grep user. Luego, muestra cada línea resultante separada en una lista de campos usando el carácter ":" como separador. El

objetivo es practicar la ejecución de comandos remotos, el registro de resultados y el procesamiento de listas con filtros en Ansible.

```
---
- name: Devolver los usuarios del sistema separados por ":"
  hosts: srv1

  tasks:
    - name: Leer los usuarios del sistema
      ansible.builtin.shell: cat /etc/passwd | grep user
      register: passwd_content

    - name: Mostrar usuarios línea a línea
      ansible.builtin.debug:
        msg: "{{ item | split(':') }}"
      loop: "{{ passwd_content.stdout_lines }}"
```

Ejercicio 3: A través de ansible_facts recuperar el día de la semana en mayúsculas. Se encuentra en el diccionario **ansible_date_time**. Además, recuperar la primera IP de la máquina (array **ansible_all_ipv4_addresses**) y convertirlo en una lista.

```
---
- name: Recuperar datos de la máquina
  hosts: srv1

  tasks:
    - name: Día de la semana en mayúsculas
      ansible.builtin.debug:
        msg: "{{ ansible_date_time.weekday | upper }}"

    - name: Recuperar ip y convertirla en lista
      ansible.builtin.debug:
        msg: "{{ ansible_all_ipv4_addresses[0] | split('.') }}"
```

Ejercicio 4: Realiza un playbook en Ansible que instale el paquete GIT en equipos Debian cuyo valor del último octeto de su primera dirección IPv4 sea igual o superior a 6.

- Extrae la primera dirección IP del host y conviértela en una lista separada por puntos.
- Obtén el valor del último octeto para usarlo en una condición.
- Instala GIT usando el módulo apt, condicionando la instalación a que la distribución sea Debian y que el último octeto de la IP cumpla con la condición numérica indicada.

```
---
- name: Instalar GIT en los equipos con IP igual o superior a 6 que sean Debian
  hosts: all
  become: yes
```

```
  tasks:
    - name: Recuperar ip y convertirla en lista
      ansible.builtin.debug:
        msg: "{{ ansible_all_ipv4_addresses[0] | split('.') }}"
      register: valores

    - name: Recuperar la última posición de la IP
      ansible.builtin.debug:
        msg: "{{valores.msg[3]}}"
      register: ip

    - name: Instalar GIT en aquellos debian que tengan una ip superior a 6
      ansible.builtin.apt:
        name: git
        update_cache: yes
      when: ip.msg | int >= 6 and ansible_distribution | lower =='debian'
```

Podemos ver que al intentar instalar en el all el software *git* para control de versiones, como el srv2 su ip no cumple con la condición establecida tiene *skipped=1*.

```
ansible@ControlNode:~/practicas/pr9$ ansible-playbook tarea4.yaml

PLAY [Instalar GIT en los equipos con IP igual o superior a 6 que sean Debian]
***************

TASK [Gathering Facts]
*********************************************************************
ok: [srv1]
ok: [srv2]

TASK [Recuperar ip y convertirla en lista]
*************************************************
ok: [srv1] => {
    "msg": [
        "192",
        "168",
        "122",
        "6"
    ]
}
ok: [srv2] => {
    "msg": [
        "192",
        "168",
        "122",
        "5"
    ]
}

TASK [Recuperar la última posición de la IP]
*************************************************
```

```
ok: [srv2] => {
    "msg": "5"
}
ok: [srv1] => {
    "msg": "6"
}

TASK [Instalar GIT en aquellos debian que tengan una ip superior a 6]
************************
skipping: [srv2]
ok: [srv1]

PLAY RECAP
*********************************************************************************
*
srv1                       : ok=4    changed=0    unreachable=0    failed=0
skipped=0    rescued=0    ignored=0
srv2                       : ok=3    changed=0    unreachable=0    failed=0
skipped=1    rescued=0    ignored=0
```