Tasks:

1. **What does YAML stand for, and what is its role in Kubernetes resource definition? Provide a brief explanation.**
2. **List the key components of a typical YAML file for defining Kubernetes resources. Explain the purpose of each component.**
3. **Review the provided YAML example for a Kubernetes Pod (as shown in the blog). Break down the YAML structure, highlighting the apiVersion, kind, metadata, and spec sections.**
4. **Create a simple YAML file for defining a Kubernetes Service. The Service should have the following properties: apiVersion of v1, kind of Service, metadata with the name "my-service," and a spec section that exposes a service for port 80, targeting a pod named "my-pod."**
5. **Using the provided YAML file above one apply it to your Kubernetes cluster using the kubectl apply command. Ensure that the service is created successfully.**

1. **YAML**

- YAML: "YAML Ain't Markup Language" or "Yet Another Markup Language."
- YAML is a human-readable data serialization format.
- It serves as a language for configuration files and data exchange between different platforms and languages.
- In Kubernetes, YAML is the preferred language for defining resources, providing a readable and expressive way to express the desired state of the cluster.

2. **Key components of a typical YAML file for defining Kubernetes resources:**

- apiVersion:
  - Specifies the API version of the Kubernetes resource.
  - Indicates the version of the Kubernetes API the resource uses.

- kind:
  - Defines the type of Kubernetes resource (e.g., Pod, Service, Deployment).
  - Specifies the category or kind of object being created.

- metadata:
  - Contains information about the resource.
  - Includes fields like name, labels, and annotations.

- spec:
  - Describes the desired state and configuration of the resource.
  - Contains specifications such as containers, volumes, and other resource-specific details.

3. **Review of the provided YAML example for a Kubernetes Pod:**

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: my-pod
spec:
  containers:
    - name: my-container
      image: nginx:latest
```

This YAML configuration defines a simple Kubernetes Pod.

- **apiVersion:** Specifies the API version for the Kubernetes resource, in this case, 'v1'.
- **kind:** Defines the type of resource, which is a 'Pod' in this configuration.
- **metadata:**
  - **name:** Sets the name of the pod to 'my-pod'.
- **spec:**
  - **containers:** Describes the containers running in the pod.
    - **name:** Specifies the name of the container, which is set to 'my-container'.
    - **image:** Specifies the Docker image for the container, in this case, 'nginx:latest'.

4. **Creating a simple YAML file for defining a Kubernetes Service.**

i. **Pod Configuration ('my-pod.yaml'):**

my-pod.yaml × my-service.yaml

my-pod.yaml > ...

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: my-pod
spec:
  containers:
    - name: my-container
      image: nginx:latest
```

ii. **Service Configuration ('my-service.yaml'):**

my-pod.yaml my-service.yaml ×

my-service.yaml > {} spec > {} selector

```yaml
apiVersion: v1
kind: Service
metadata:
  name: my-service
spec:
  ports:
    - port: 80
  selector:
    app: my-pod
```

- **apiVersion:** Specifies the API version for the Kubernetes resource, in this case, **'v1'**.
- **kind:** Defines the type of resource, which is a **'Service'** in this configuration.
- **metadata:**
  - **name:** Sets the name of the service to **'my-service'**.
- **spec:**
  - **ports:** Describes the ports configuration for the service.
    - **port:** Specifies that the service should expose port 80.

- **selector:** Specifies the label selector to determine which pods the service should target.
    - **app: my-pod:** Defines the label selector to target pods with the label **'app'** set to **'my-pod'**.

Applying Configurations:

kubectl apply -f my-pod.yaml

kubectl apply -f my-service.yaml

```
PS C:\Users\HP\Desktop\K8s-Assignment-3> kubectl apply -f my-pod.yaml
>>
pod/my-pod created
PS C:\Users\HP\Desktop\K8s-Assignment-3> kubectl apply -f my-service.yaml
>>
service/my-service created
```

After applying, check the status and details:

```
PS C:\Users\HP\Desktop\K8s-Assignment-3> kubectl get pod my-pod
NAME     READY   STATUS    RESTARTS   AGE
my-pod   1/1     Running   0          13s
PS C:\Users\HP\Desktop\K8s-Assignment-3> kubectl get service my-service
NAME         TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)   AGE
my-service   ClusterIP   10.105.63.113   <none>        80/TCP    24s
```