1- How many Namespaces exist on the system?

```
amr@amrgomaa:~$ kubectl get ns
NAME              STATUS   AGE
default           Active   13d
kube-node-lease   Active   13d
kube-public       Active   13d
kube-system       Active   13d
amr@amrgomaa:~$
```

2-How many pods exist in the kube-system namespace?

```
amr@amrgomaa:~$ kubectl get pods -n kube-system
NAME                               READY   STATUS    RESTARTS        AGE
coredns-565d847f94-bg4th           1/1     Running   4 (3m20s ago)   13d
etcd-minikube                      1/1     Running   4 (3m20s ago)   13d
kube-apiserver-minikube            1/1     Running   4 (3m20s ago)   13d
kube-controller-manager-minikube   1/1     Running   4 (3m20s ago)   13d
kube-proxy-xzqx5                   1/1     Running   4 (3m20s ago)   13d
kube-scheduler-minikube            1/1     Running   4 (3m20s ago)   13d
storage-provisioner                1/1     Running   8 (2m22s ago)   13d
amr@amrgomaa:~$
```

3- create a Deployment with
name= deployment-1
image= busybox
replicas= 3

```
apiVersion: apps/v1
kind: Deployment
metadata:
  labels:
    app: deployment1
  name: deployment1
spec:
  replicas: 3
  selector:
    matchLabels:
      app: deployment1
  strategy: {}
  template:
    metadata:
      labels:
        app: deployment1
    spec:
      containers:
      - image: busybox
        name: busybox
        resources: {}
        tty: true
status: {}
"rep1.yml" 23L, 375B                                    22,17         All
```

```
amr@amrgomaa:~/k8s$ kubectl apply -f rep1.yml
deployment.apps/deployment1 created
```

4- How many Deployments and ReplicaSets exist on the system now?

```
amr@amrgomaa:~/k8s$ kubectl get deployment
NAME          READY   UP-TO-DATE   AVAILABLE   AGE
deployment1   3/3     3            3           2m16s
```

```
amr@amrgomaa:~/k8s$ kubectl get rs
NAME                     DESIRED   CURRENT   READY   AGE
deployment1-85bb456674   3         3         3       2m17s
```

5- How many pods are ready with the deployment-1?

3 pods are ready, using tty = true or add a command will keep the pods ready and running

6- Update deployment-1 image to nginx then check the ready pods again

```
amr@amrgomaa:~/k8s$ kubectl set image deployment/deployment1 busybox=nginx
deployment.apps/deployment1 image updated
amr@amrgomaa:~/k8s$
```

7- Run kubectl describe deployment deployment-1 and check events

What is the deployment strategy used to upgrade the deployment-1?

```
StrategyType:           RollingUpdate
```

```
Normal  ScalingReplicaSet  18m   deployment-controller  Scaled up replica set deployment1-5cf59dfbd9 to 3
Normal  ScalingReplicaSet  16m   deployment-controller  Scaled up replica set deployment1-85bb456674 to 1
Normal  ScalingReplicaSet  16m   deployment-controller  Scaled down replica set deployment1-5cf59dfbd9 to 2 from 3
Normal  ScalingReplicaSet  16m   deployment-controller  Scaled up replica set deployment1-85bb456674 to 2 from 1
Normal  ScalingReplicaSet  16m   deployment-controller  Scaled down replica set deployment1-5cf59dfbd9 to 1 from 2
Normal  ScalingReplicaSet  16m   deployment-controller  Scaled up replica set deployment1-85bb456674 to 3 from 2
Normal  ScalingReplicaSet  16m   deployment-controller  Scaled down replica set deployment1-5cf59dfbd9 to 0 from 1
Normal  ScalingReplicaSet  71s   deployment-controller  Scaled up replica set deployment1-55c75769d8 to 1
Normal  ScalingReplicaSet  67s   deployment-controller  Scaled down replica set deployment1-85bb456674 to 2 from 3
Normal  ScalingReplicaSet  67s   deployment-controller  Scaled up replica set deployment1-55c75769d8 to 2 from 1
Normal  ScalingReplicaSet  63s   deployment-controller  Scaled down replica set deployment1-85bb456674 to 1 from 2
Normal  ScalingReplicaSet  63s   deployment-controller  Scaled up replica set deployment1-55c75769d8 to 3 from 2
Normal  ScalingReplicaSet  59s   deployment-controller  Scaled down replica set deployment1-85bb456674 to 0 from 1
```

8- Rollback the deployment-1

What is the used image with the deployment-1?

```
amr@amrgomaa: ~/k8s
amr@amrgomaa:~/k8s$ kubectl rollout history deploy deployment1
deployment.apps/deployment1
REVISION  CHANGE-CAUSE
1         <none>
2         <none>

amr@amrgomaa:~/k8s$ kubectl rollout undo deploy deployment1
deployment.apps/deployment1 rolled back
amr@amrgomaa:~/k8s$
```

```
amr@amrgomaa:~/k8s$ kubectl describe deploy deployment1 | grep Image:
    Image:        busybox
amr@amrgomaa:~/k8s$
```

10- Create a deployment with
Name: dev-deploy
Image: redis
Replicas: 2
Namespace: dev
Resources Requests:
CPU: .5 vcpu
Mem: 1G
Resources Limits:
CPU: 1 vcpu
Mem: 2G

```
amr@amrgomaa:~/k8s$ kubectl create ns dev
namespace/dev created
amr@amrgomaa:~/k8s$
```

```
amr@amrgomaa:~/k8s$ kubectl apply -f dev.yml
deployment.apps/dev-deploy created
amr@amrgomaa:~/k8s$
```

```
amr@amrgomaa:~/k8s$ kubectl apply -f dev.yml
deployment.apps/dev-deploy created
amr@amrgomaa:~/k8s$ kubectl get pods -n dev
NAME                          READY   STATUS    RESTARTS   AGE
dev-deploy-977c688d5-dml6j    1/1     Running   0          25s
dev-deploy-977c688d5-n9ljz    1/1     Running   0          25s
amr@amrgomaa:~/k8s$
```

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  labels:
    app: dep-redis
  name: dev-deploy
  namespace: dev
spec:
  replicas: 2
  selector:
    matchLabels:
      app: app1
  strategy: {}
  template:
    metadata:
      labels:
        app: app1
    spec:
      containers:
      - image: redis
        name: redis
        resources:
          requests:
            memory: "1Gi"
            cpu: "500m"
          limits:
            memory: "2Gi"
            cpu: "1000m"
status: {}
~
~
```