KUBERNETES (ENGLISH)

1. forduló

A kategória támogatója: Nokia

Ismertető a feladatlaphoz

Please make sure you read the instructions below before starting the worksheet:

ALL questions have a correct answer.

There is NO question where all the answers are correct, if you do tick all the answers, you will automatically score 0 points for that question.

If there is more than one answer choice, marking an incorrect answer will result in a minus mark.

Questions with a radio button have one correct answer.

If the time limit for the task sheet expires, the system will AUTOMATICALLY submit the task sheet with the answers marked.

We recommend that you do NOT start the worksheets with attachments on your mobile phone, as this will be pointed out before the worksheets concerned.

The data-request exercises will NOT be marked out of the total, only the multiple-choice exercises.

Rankings will be shown after the 4th round, in percentage form: you will be in the top 20-40-60% in a given category.

If you open the same questionnaire from several browsers, several windows or several devices at the same time, we cannot be held responsible for any anomalies that may occur in the data saving!

The use of ChatGPT is not prohibited, but we do NOT accept comments referring to it!

Any questionnaire solved in a noticeably short time will be disqualified, in any other suspicious case we reserve the right to invalidate the round!

We wish you a good competition!

Let's get through the hard part! :)

Kubernetes workloads:

In Kubernetes, a "workload" is a term used to describe the type of application or task that you want to run on the cluster. Kubernetes provides several types of workloads to including various application deployment scenarios. These workloads define how applications are scheduled, scaled, and managed.

Before you start, please read the following hint:

KDiff3 is a free and open-source diff and merge tool that can be helpful during the solution of coding-related exercises.
Download link: https://sourceforge.net/projects/kdiff3/files/

Indítás utáni csatolmányok

1. feladat 1 pont

Extend the valid deployment skeleton with the following requirements:

- Deployment target namespace: ithon
- Create 3 replicas
- Add an environment variable with name ingress_port and value 30100.

deployment_skeleton.yaml

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nokia-app-deployment
  labels:
    app: nokia-web-app
spec:
  selector:
    matchLabels:
      app: nokia-web-app
  template:
    metadata:
      labels:
        app: nokia-web-app
    spec:
      containers:
        - name: nokia-container
          image: nokia-registry:5000/nokia-container-image:1.0.0
          ports:
            - containerPort: 80
```

Select the correct Deployment definition which contains all of the requirement above.

Válasz

```
apiVersion: apps/v1
kind: Statefulset
metadata:
  name: nokia-app-deployment
  labels:
    namespace: ithon
    app: nokia-web-app
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nokia-web-app
  template:
    metadata:
      labels:
        app: nokia-web-app
    spec:
      containers:
        - env:
            - name: ingress_port
              value: "30100"
```

```
        - name: nokia-container
          image: nokia-registry:5000/nokia-container-image:1.0.0
          ports:
            - containerPort: 80
```



```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nokia-app-deployment
  namespace: ithon
  labels:
    app: nokia-web-app
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nokia-web-app
  template:
    metadata:
      labels:
        app: nokia-web-app
    spec:
      containers:
        - name: nokia-container
          image: nokia-registry:5000/nokia-container-image:1.0.0
          ports:
            - containerPort: 80
          env:
            - name: ingress_port
              value: "30100"
```



```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nokia-app-deployment
  namespace: ithon
  labels:
    app: nokia-web-app
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nokia-web-app
```

```
  template:
    metadata:
      labels:
        app: nokia-web-app
    spec:
      containers:
        - env:
            name: egress_port
            value: "30100"
        - name: nokia-container
          image: nokia-registry:5000/nokia-container-image:1.0.0
          ports:
            - containerPort: 80
```



```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nokia-app-deployment
  namespace: megmerettetes
  labels:
    app: nokia-web-app
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nokia-web-app
  template:
    metadata:
      labels:
        app: nokia-web-app
    spec:
      containers:
        - name: nokia-container
          image: nokia-registry:5000/nokia-container-image:1.0.0
          ports:
            - containerPort: 80
          env:
            - name: ingress_port
              value: "30100"
```



```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nokia-app-deployment
```

```
  namespace: ithon
  labels:
    app: nokia-web-app
spec:
  replicas: 22
  selector:
    matchLabels:
      app: nokia-web-app
  template:
    metadata:
      labels:
        app: nokia-web-app
    spec:
      containers:
        - name: nokia-container
          image: nokia-registry:5000/nokia-container-image:1.0.0
          ports:
            - containerPort: 80
          env:
            - name: ingress_port
              value: "30100"
```



```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nokia-app-deployment
  namespace: ithon
  labels:
    app: nokia-web-app
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nokia-web-app
  template:
    metadata:
      labels:
        app: nokia-web-app
    spec:
      containers:
        - name: nokia-container
          image: nokia-registry:5000/nokia-container-image:1.0.0
          ports:
            - containerPort: 80
          env:
```

```
      - name: ingress_port
        value: 55000
```

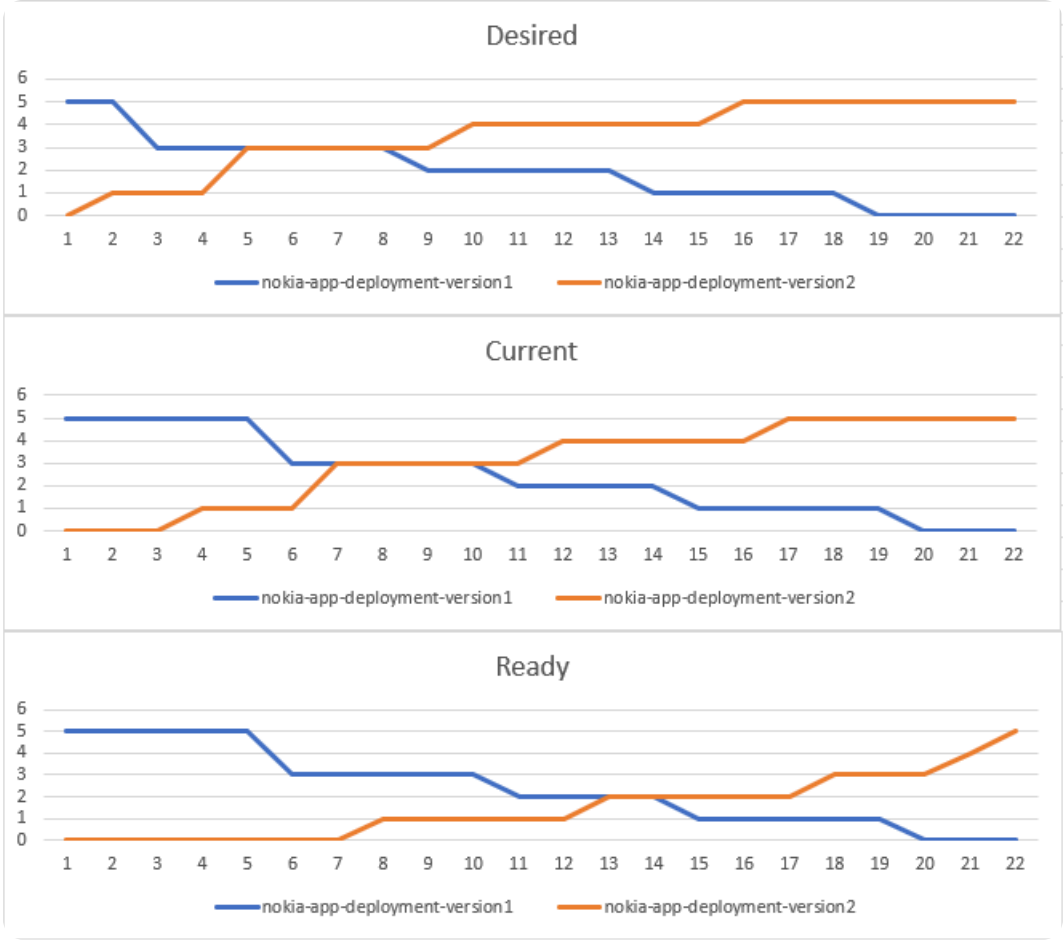
2. feladat 2 pont

We have a deployment with the following rolling upgrade specs:

```
metadata:
  name: nokia-update
spec:
  replicas: 5
  strategy:
    type: RollingUpdate
    rollingUpdate:
      maxSurge: <A>
      maxUnavailable: <B>
```

The order of state changes during an upgrade.

Time	nokia-app-deployment-version1			nokia-app-deployment-version2		
	desired	current	ready	desired	current	ready
1	5	5	5	0	0	0
2	5	5	5	1	0	0
3	3	5	5	1	0	0
4	3	5	5	1	1	0
5	3	5	5	3	1	0
6	3	3	3	3	1	0
7	3	3	3	3	3	0
8	3	3	3	3	3	1
9	2	3	3	3	3	1
10	2	3	3	4	3	1
11	2	2	2	4	3	1
12	2	2	2	4	4	1
13	2	2	2	4	4	2
14	1	2	2	4	4	2
15	1	1	1	4	4	2
16	1	1	1	5	4	2
17	1	1	1	5	5	2
18	1	1	1	5	5	3
19	0	1	1	5	5	3
20	0	0	0	5	5	3
21	0	0	0	5	5	4
22	0	0	0	5	5	5



What is the closest A-B value combination that allows the above upgrade scenario?

Válasz

- A: 25%, B: 20%
- A: 20%, B: 40%
- A: 40%, B: 40%
- A: 10%, B: 50%
- A: 25%, B: 25%
- A: 40%, B: 20%
- A: 25%, B: 75%
- A: 40%, B: 60%

3. feladat 3 pont

Extend the valid deployment skeleton with the following requirements:

Add liveness probe with HTTP request and set the `failureThreshold` to 3

Perform a liveness probe every 10 seconds but wait 15 seconds before performing the first probe

Add Readiness probe HTTP request and set the `failureThreshold` to 3

Perform a readiness probe every 15 seconds but wait 30 seconds before performing the first probe

Add resource Reqs

CPU: 100 millicore

MEMORY: 128 Mebibyte

Add resouce limits:

CPU: 500 millicore

MEMORY: 512 Mebibyte

deployment_skeleton.yaml

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nokia-app-deployment
  labels:
    app: nokia-web-app
spec:
  selector:
    matchLabels:
      app: nokia-web-app
  template:
    metadata:
      labels:
        app: nokia-web-app
    spec:
      containers:
        - name: nokia-container
          image: nokia-registry:5000/nokia-container-image:1.0.0
```

Select the correct Deployment definition extension which contains all of the requirement above.

Válasz

```
livenessProbe:
  httpGet:
    path: /health
    port: 8080
  initialDelaySeconds: 15
  periodSeconds: 10
readinessProbe:
  httpGet:
    path: /health
    port: 8080
  initialDelaySeconds: 30
```

```
    periodSeconds: 15
  resources:
    requests:
      cpu: "1"
      memory: "128Mi"
    limits:
      cpu: "5"
      memory: "512Mi"
```

```
  readinessProbe:
    httpGet:
      path: /health
      port: 8080
    initialDelaySeconds: 15
    periodSeconds: 10
  livenessProbe:
    httpGet:
      path: /health
      port: 8080
    initialDelaySeconds: 30
    periodSeconds: 15
  resources:
    requests:
      cpu: "500m"
      memory: "512Mi"
    limits:
      cpu: "100m"
      memory: "128Mi"
```

```
  livenessProbe:
    httpGet:
      path: /health
      port: 8080
    initialDelaySeconds: 15
    periodSeconds: 10
  readinessProbe:
    httpGet:
      path: /health
      port: 8080
    initialDelaySeconds: 30
    periodSeconds: 15
  resources:
    requests:
      cpu: "100m"
```

```
      memory: "128Mi"
    limits:
      cpu: "500m"
      memory: "512Mi"
```

```
livenessProbe:
  httpGet:
    path: /health
    port: 8080
  initialDelaySeconds: 10
  periodSeconds: 15
readinessProbe:
  httpGet:
    path: /health
    port: 8080
  initialDelaySeconds: 15
  periodSeconds: 30
resources:
  requests:
    cpu: "100m"
    memory: "128Mi"
  limits:
    cpu: "500m"
    memory: "512Mi"
```

```
livenessProbe:
  httpGet:
    path: /health
    port: 8080
  initialDelaySeconds: 15
  periodSeconds: 10
  failureThreshold: 10
readinessProbe:
  httpGet:
    path: /health
    port: 8080
  initialDelaySeconds: 30
  periodSeconds: 15
  failureThreshold: 10
resources:
  requests:
    cpu: "100m"
    memory: "128Mi"
  limits:
```

```
        cpu: "500m"
        memory: "512Mi"
```

Megoldások beküldése