Q&A!

Up-to-date Questions and Answers from authentic resources to improve knowledge and pass the exam at very first attempt.
----- Guaranteed.

KILL EXAMS

PASS

Cisco

350-901

Developing Applications using Cisco Core Platforms and APIs (DEVCOR)

Which statement about microservices architecture is true?
A. Applications are written in a single unit.
B. It is a complex application composed of multiple independent parts.
C. It is often a challenge to scale individual parts.
D. A single faulty service can bring the whole application down.

Answer: B

Question: 304

Application sometimes store configuration as constants in the code, which is a violation of strict separation of configuration from code.

Where should application configuration be stored?
A. environment variables
B. YAML files
C. Python libraries
D. Dockerfiles
E. INI files

Answer: B

Question: 305

Which two methods are API security best practices? (Choose two.)
A. Use tokens after the identity of a client has been established.
B. Use the same operating system throughout the infrastructure.
C. Use encryption and signatures to secure data.
D. Use basic auth credentials over all internal API interactions.
E. Use cloud hosting services to manage security configuration.

Answer: AC

Question: 306

DRAG DROP

Drag and drop the steps from the left into the correct sequence on the right to describe how to use Git to maintain the current HEAD and revert back to a previous commit, while undoing all intermediate commits.

```
- name: Configure Interfaces
  with_items: "{{interfaces}}"
  netconf_config:
    <<: *host_info
    xml: |
      <config>
        <interfaces xmlns="urn:ietf:params:xml:ns:yang:ietf-interefaces">
          <interface>
            <name>{{item.interface_type}}{{item.interface_id}}</name>
            <description>{{item.description}}</description>
            <type xmlns:ianaift="urn:ietf:params:xml:ns:yang:iana-if-type">ianaift:ethernetCsmacd</type>
            <enabled>true</enabled>
            <ipv4 xmlns="urn:ietf:params:xml:ns:yang:ietf-ip">
              <address>
                <ip>{{item.ip_address}}</ip>
                <netmask>{{item.subnet_mask}}</netmask>
              </address>
            </ipv4>
          </interface>
        </interfaces>
      </config>
```

Answer:

```
import sys, requests

URL = "http://ios-xe-mgmt.cisco.com:9443"
USER - 'root'
PASS = 'C!isco0123'

url = URL + "/restconf/data/ietf-interfaces:interfaces-state"
headers = {'content-type': 'application/vang-data+json', 'accept':
           'application/yang-data+json'            }

try:
    result = requests.get(url, auth=(USER,PASS), headers=headers)
    r_json = result.json()
    flagDown = 0
    for record in r_json["ietf-interfaces:interfaces"]["interface"]:
        print("{0:<35}".format("interface: " + record["name"]), end="")
        print("{0:<5}".format("ip: "), end="")
        if('address' in record["ietf-ip:ipv4"]):
            print("{0:<15}".format(record["ietf-ip:ipv4"]["address"][0]["ip"]), end="")
        else:
            print("{0:<15}".format(record["No IPv4"], end="")
        print("{0:<9}".format("status: "), end="")
        print(str(record["enabled"]))
        if(record["enabled"]==False):
            flagDown=1
    print("")
    if(flagDown):
        print("At least one interface is down")
    else
        print("All interfaces are up")

except:
    print("Exception: " + str(sys.ecx_info()[0] + " " + str(sys.ecx_info()[1]))
    print("Error: " + str(result.status_code), result.text)
```

Question: 307

Refer to the exhibit.

```
curl --insecure -H "Accept: application/json" -H "Content-Type:
application/json" -d @token_data https://ast0072-
pod.cisco.com:33333//api/fdm/latest/fdm/token
```

The cURL POST request creates an OAuth access token for authentication with FDM API requests.

What is the purpose of the file "@token_data" that cURL is handling?
A. This file is a container to log possible error responses in the request.

B. This file is given as input to store the access token received from FD
D. This file is used to send authentication related headers.
E. This file contains raw data that is needed for token authentication.

Answer: B

Question: 308

A user is receiving a 429 Too Many Requests error.

Which scheme is the server employing that causes this error?
A. rate limiting
B. time outs
C. caching
D. redirection

Answer: A

Question: 309

Which two situations are flagged by software tools designed for dependency checking in continuous integration environments, such as OWASP? (Choose two.)
A. publicly disclosed vulnerabilities related to the included dependencies
B. mismatches in coding styles and conventions in the included dependencies
C. incompatible licenses in the included dependencies
D. test case failures introduced by bugs in the included dependencies
E. buffer overflows to occur as the result of a combination of the included dependencies

Answer: AE

Question: 310

Which two data encoding techniques are supported by gRPC? (Choose two.)
A. XML
B. JSON
C. ASCII
D. ProtoBuf
E. YAML

Answer: BE

Question: 311

An organization manages a large cloud-deployed application that employs a microservices architecture across multiple data centers. Reports have received about application slowness. The container orchestration logs show that faults have been raised in a variety of containers that caused them to fail and then spin up brand new instances.

Which two actions can improve the design of the application to identify the faults? (Choose two.)
A. Automatically pull out the container that fails the most over a time period.
B. Implement a tagging methodology that follows the application execution from service to service.
C. Add logging on exception and provide immediate notification.
D. Do a write to the datastore every time there is an application failure.
E. Implement an SNMP logging system with alerts in case a network link is slow.

Answer: BC

Question: 312

DRAG DROP

Drag and drop the git commands from the left into the correct order on the right to create a feature branch

from the master and then incorporate that feature branch into the master.

```
- name: Configure Interfaces
  with_items: "{{interfaces}}"
  netconf_config:
    <<: *host_info
    xml: |
        <config>
          <interfaces xmlns="urn:ietf:params:xml:ns:yang:ietf-interefaces">
            <interface>
                <name>{{item.interface_type}}{{item.interface_id}}</name>
                <description>{{item.description}}</description>
                <type xmlns:ianaift="urn:ietf:params:xml:ns:yang:iana-if-type">ianaift:ethernetCsmacd</type>
                <enabled>true</enabled>
                <ipv4 xmlns="urn:ietf:params:xml:ns:yang:ietf-ip">
                  <address>
                      <ip>{{item.ip_address}}</ip>
                      <netmask>{{item.subnet_mask}}</netmask>
                  </address>
                </ipv4>
            </interface>
          </interfaces>
        </config>
```

Answer:

Answer Area

git branch -d feature	git checkout -b feature master
git checkout -b feature master	git checkout master
git checkout master	git merge --no-ff feature
git push origin master	git push origin master
git merge --no-ff feature	git branch -d feature

Question: 313

DRAG DROP

Refer to the exhibit.

```
module: Cisco-IOS-XE-vlan
  augment /ios:native/ios:vlan:
   +--rw access-map* [name]
   |  +--rw name          string
   |  +--rw value?        uint16
   |  +--rw action?       enumeration
   |  +--rw match
   |     +--rw ipv6
   |     |  +--rw address*     string
   |     +--rw ip
   |        +--rw address*     string
   +--rw configuration* [vlan-id]
   |  +--rw vlan-id       union
   |  +--rw ip
   |  |  +--rw flow
   |  |     +--rw monitor* [flow-monitor]
   |  |        +--rw flow-monitor       string
   |  |        +--rw input?             empty
   |  |        +--rw output?            empty
   |  +--rw ipv6
   |  |  +--rw nd
   |  |  |  +--rw suppress!
   |  |  |     +--rw attach-policy?          string
   |  |  +--rw dhcp
   |  |     +--rw guard!
   |  |        +--rw attach-policy?          string
   |  +--rw member
   |     +--rw evpn-instance
   |     |  +--rw evpn-instance?    uint16
   |     |  +--rw vni?              uint32
   |     +--rw vni?                 uint32
   +--rw filter* [word]
```

```
https://ios-xe-mgmt.cisco.com:9443/<item 1>/<item 2>/<item 3>/<item 4>/
```

Drag and drop parts of the URL from the left onto the item numbers on the right that match the missing sections in the exhibit to create the appropriate RESTCONF URL to query the VLAN configuration given this YANG model. Not all URL parts are used.

Answer Area

vlan	<item 1>
restconf	<item 2>
interfaces	<item 3>
data	<item 4>
native	

Answer:

Answer Area

vlan	restconf
restconf	data
interfaces	native
data	vlan
native	

DRAG DROP

Drag and drop the expressions from below onto the code to implement error handling. Not all options are used.

Answer Area

```
base_url = "https://api.meraki.com/api/v0"
posturl = '%s/networks/%s/staticRoutes' % ((str(base_url), str(networked)))
headers = {
     'x-cisco-meraki-api-key': api_key,
     'Content-Type': 'application/json'
}
routes = [ {
     "subnet": "10.16.4.0/22",
     "gatewayIp": "10.1.0.20",
     "name": "ROUTE1",
     "enabled": true
   }
   {

     "subnet": "10.253.254.0/24",
     "gatewayIp": "10.1.0.20",
     "name": "ROUTE2",
     "enabled": true
   }
   {
"subnet": "10.168.0.0/21",
     "gatewayIp": "10.1.0.20",
     "name": "ROUTE3",
     "enabled": true
   } ]

for route in routes:
     print("Adding static: " + str(route['subnet']))
     response = requests.post(posturl, json=route, headers=headers)
     ┌─────────────────┐
     │                 │
     └─────────────────┘
     print("Done!")
     ┌─────────────────┐
     │                 │
     └─────────────────┘
     print("Failed to add static: " + str(route['subnet']) + "\n" + response.text)
```

| if response == 601: | else: | when: |
| if response == 201: | elif: | |

Answer Area

```
base_url = "https://api.meraki.com/api/v0"
posturl = '%s/networks/%s/staticRoutes' % ((str(base_url), str(networked)))
headers = {
     'x-cisco-meraki-api-key': api_key,
     'Content-Type': 'application/json'
}
routes = [ {
     "subnet": "10.16.4.0/22",
     "gatewayIp": "10.1.0.20",
     "name": "ROUTE1",
     "enabled": true
   }
   {
     "subnet": "10.253.254.0/24",
     "gatewayIp": "10.1.0.20",
     "name": "ROUTE2",
     "enabled": true
   }
   {
"subnet": "10.168.0.0/21",
     "gatewayIp": "10.1.0.20",
     "name": "ROUTE3",
     "enabled": true
   } ]

for route in routes:
     print("Adding static: " + str(route['subnet']))
     response = requests.post(posturl, json=route, headers=headers)
```
`if response == 201:`
```
     print("Done!")
```
`else:`
```
     print("Failed to add static: " + str(route['subnet']) + "\n" + response.text)
```

| if response == 601: | else: | when: |

| if response == 201: | elif: |

Question: 315

Refer to the exhibit.

```
name: VRFs
ios_vrf:
    vrfs: "{{ local_vrfs }}"
    state: present
    purge: yes
```

The YAML represented is using the ios_vrf module.

As part of the Ansible playbook workflow, what is the result when this task is run?
A. VRFs not defined in the host_vars file are removed from the device.
B. VRFs not defined in the host_vars file are added to the device, and any other VRFs on the device remain.
C. VRFs defined in the host_vars file are removed from the device.
D. VRFs are added to the device from the host_vars file, and any other VRFs on the device are removed.

Question: 316

User report that they can no longer process transactions with the online ordering application, and the logging dashboard is displaying these messages.

Fri Jan 10 19:37:31.123 EST 2020 [FRONTEND] INFO: Incoming request to add item to cart from user 45834534858

Fri Jan 10 19:37:31 247 EST 2020 [BACKEND] INFO: Attempting to add item to cart

Fri Jan 10 19:37:31 250 EST 2020 [BACKEND] ERROR: Failed to add item: MYSQLDB ERROR: Connection refused

What is causing the problem seen in these log messages?
A. The database server container has crashed.
B. The backend process is overwhelmed with too many transactions.
C. The backend is not authorized to commit to the database.
D. The user is not authorized to add the item to their cart.

Question: 317

Refer to the exhibit.

```
import sys, requests

URL = "http://ios-xe-mgmt.cisco.com:9443"
USER - 'root'
PASS = 'C!isco0123'

url = URL + "/restconf/data/ietf-interfaces:interfaces-state"
headers = {'content-type': 'application/vang-data+json', 'accept':
            'application/yang-data+json'        }

try:
   result = requests.get(url, auth=(USER,PASS), headers=headers)
   r_json = result.json()
   flagDown = 0
   for record in r_json["ietf-interfaces:interfaces"]["interface"]:
        print("{0:<35}".format("interface: " + record["name"]), end="")
        print("{0:<5}".format("ip: "), end="")
        if('address' in record["ietf-ip:ipv4"]):
            print("{0:<15}".format(record["ietf-ip:ipv4"]["address"][0]["ip"]), end="")
        else:
            print("{0:<15}".format(record["No IPv4"], end="")
        print("{0:<9}".format("status: "), end="")
        print(str(record["enabled"]))
        if(record["enabled"]==False):
            flagDown=1
   print("")
   if(flagDown):
        print("At least one interface is down")
   else
        print("All interfaces are up")

except:
   print("Exception: " + str(sys.ecx_info()[0] + " " + str(sys.ecx_info()[1]))
   print("Error: " + str(result.status_code), result.text)
```

What is the output of this IOS-XE configuration program?
A. interface operational status in IPv6 addresses
B. interface administrative status in IPv4 addresses
C. interface operational status in IPv4 addresses
D. interface administrative status in IPv6 addresses

Answer: D

SAMPLE QUESTIONS