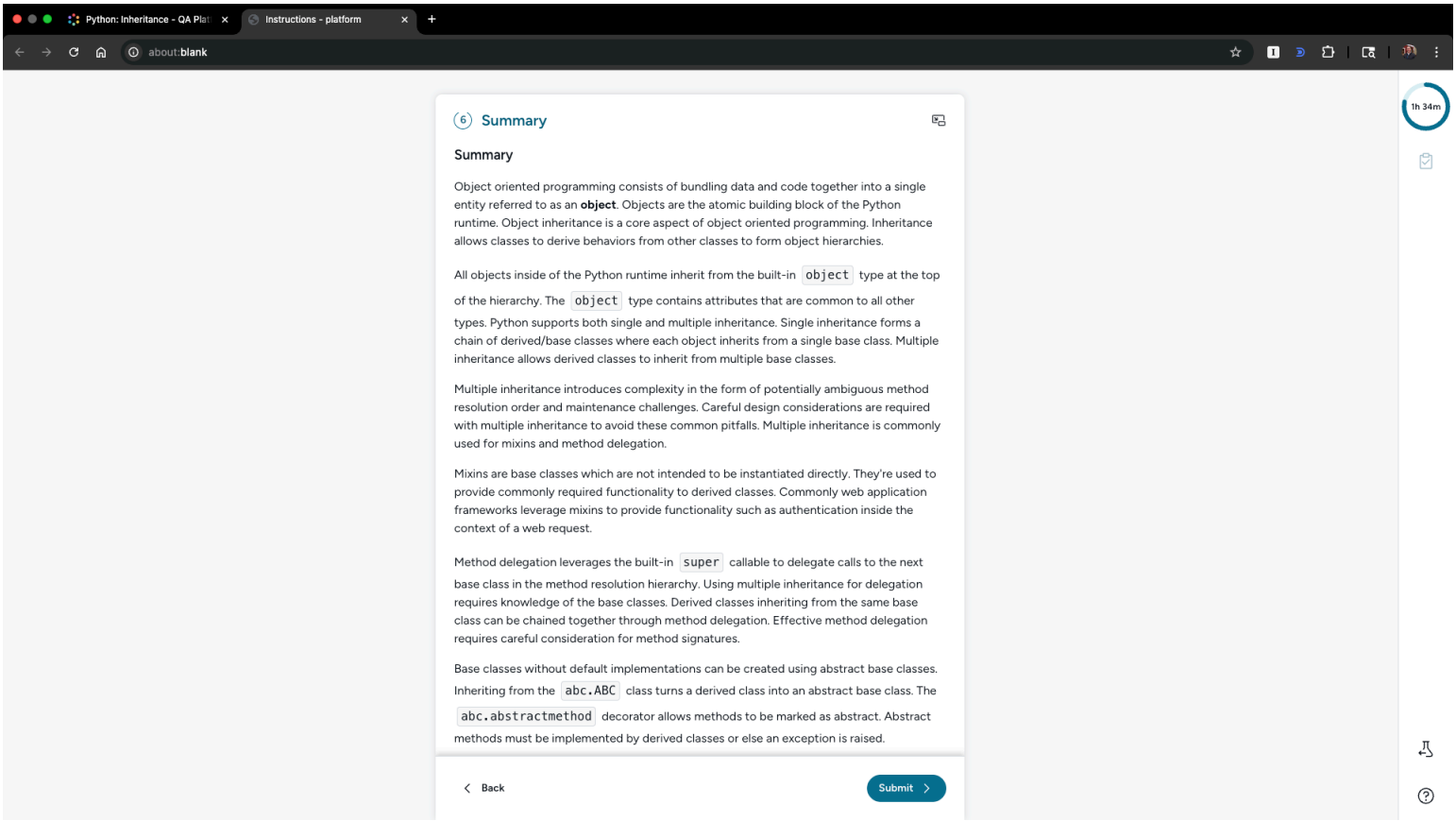
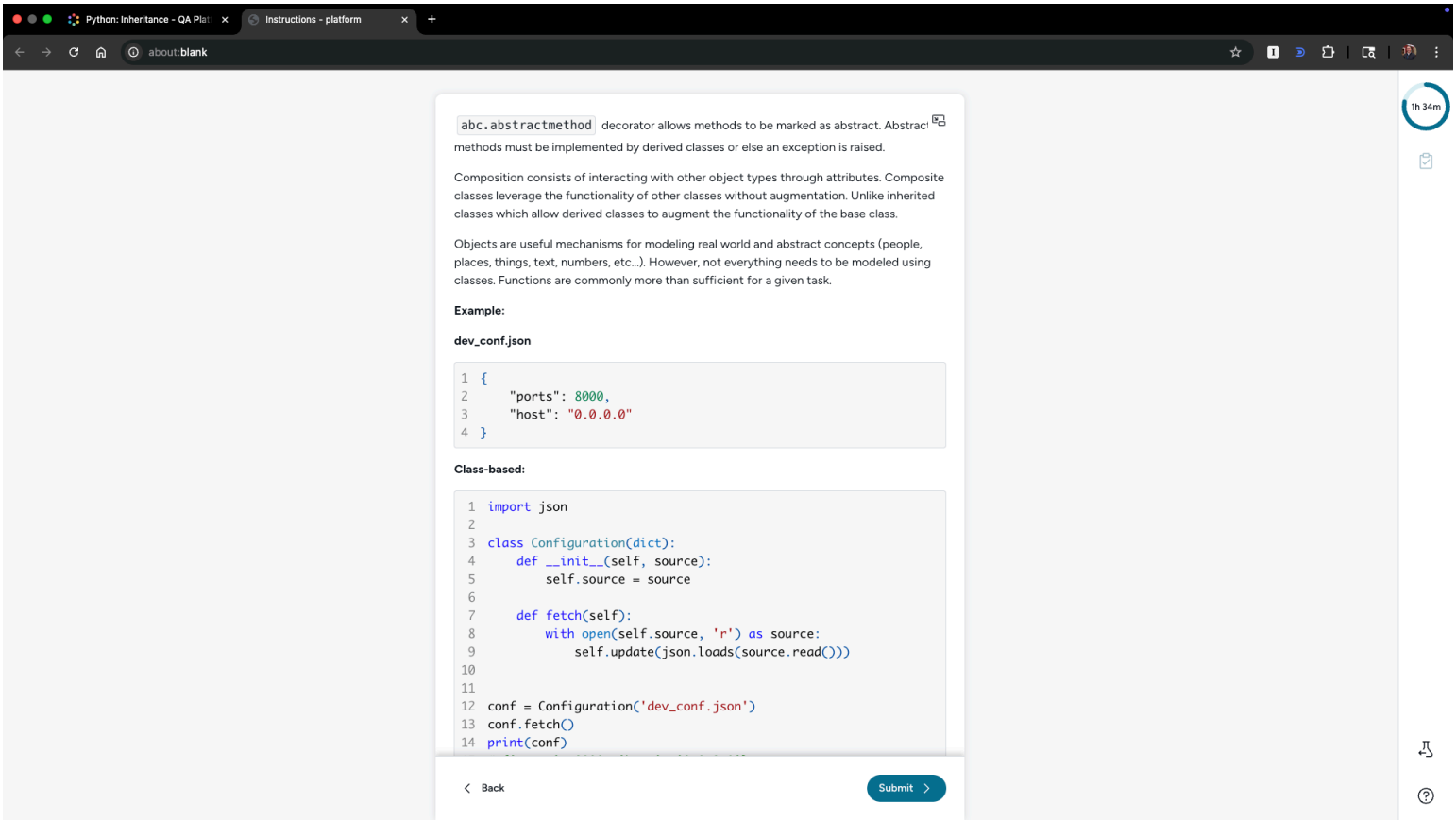
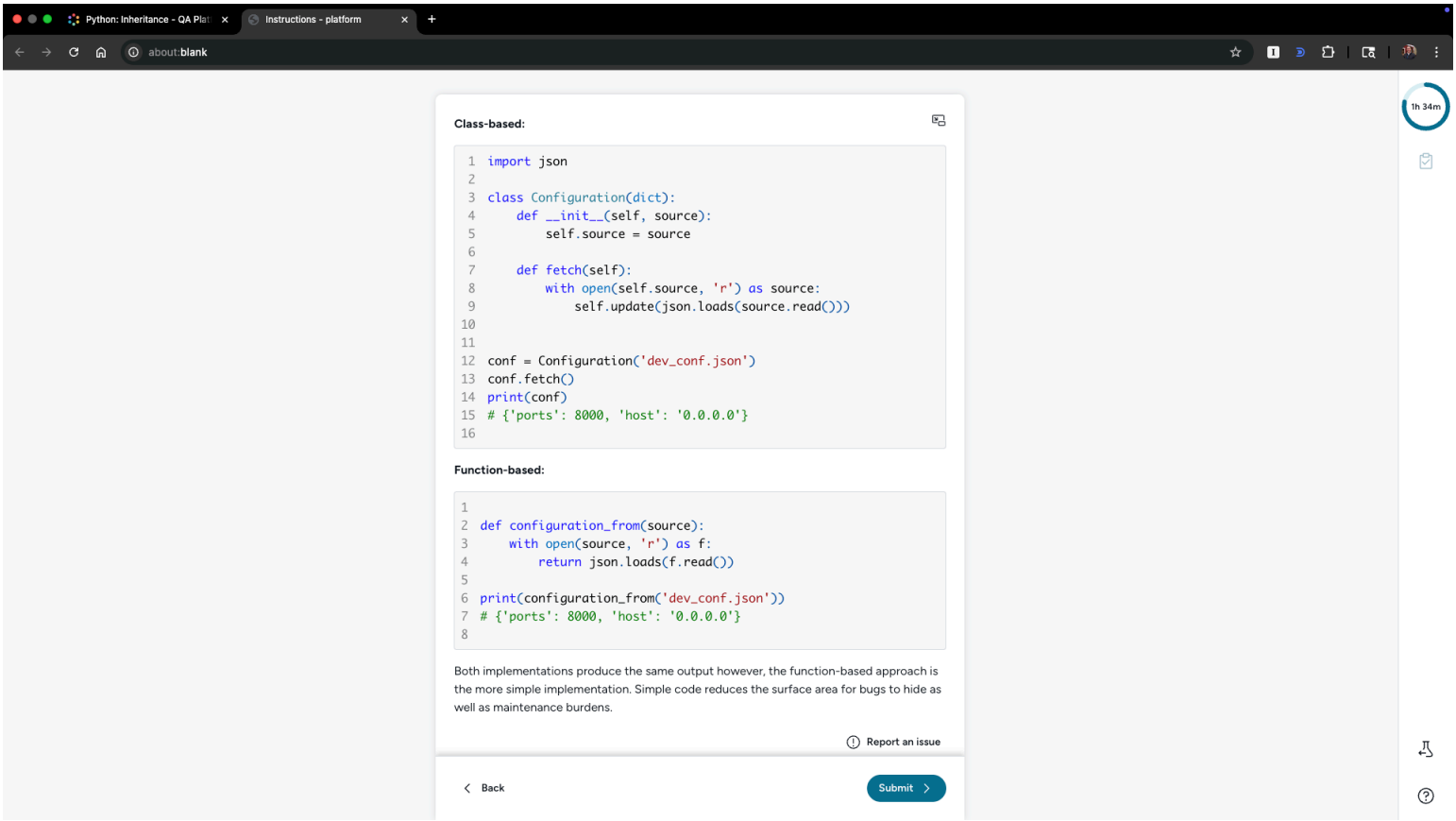

Python: Inheritance - QA Plat
Instructions - platform
about:blank
6 Summary
Summary
Summary
Object oriented programming consists of bundling data and code together into a single entity referred to as an **object**. Objects are the atomic building block of the Python runtime. Object inheritance is a core aspect of object oriented programming. Inheritance allows classes to derive behaviors from other classes to form object hierarchies.
All objects inside of the Python runtime inherit from the built-in `object` type at the top of the hierarchy. The `object` type contains attributes that are common to all other types. Python supports both single and multiple inheritance. Single inheritance forms a chain of derived/base classes where each object inherits from a single base class. Multiple inheritance allows derived classes to inherit from multiple base classes.
Multiple inheritance introduces complexity in the form of potentially ambiguous method resolution order and maintenance challenges. Careful design considerations are required with multiple inheritance to avoid these common pitfalls. Multiple inheritance is commonly used for mixins and method delegation.
Mixins are base classes which are not intended to be instantiated directly. They're used to provide commonly required functionality to derived classes. Commonly web application frameworks leverage mixins to provide functionality such as authentication inside the context of a web request.
Method delegation leverages the built-in `super` callable to delegate calls to the next base class in the method resolution hierarchy. Using multiple inheritance for delegation requires knowledge of the base classes. Derived classes inheriting from the same base class can be chained together through method delegation. Effective method delegation requires careful consideration for method signatures.
Base classes without default implementations can be created using abstract base classes. Inheriting from the `abc.ABC` class turns a derived class into an abstract base class. The `abc.abstractmethod` decorator allows methods to be marked as abstract. Abstract methods must be implemented by derived classes or else an exception is raised.
Back
Submit
1h 34m


Python: Inheritance - QA Plat
Instructions - platform
about:blank
1h 34m
abc.abstractmethod decorator allows methods to be marked as abstract. Abstrac
methods must be implemented by derived classes or else an exception is raised.
Composition consists of interacting with other object types through attributes. Composite classes leverage the functionality of other classes without augmentation. Unlike inherited classes which allow derived classes to augment the functionality of the base class.
Objects are useful mechanisms for modeling real world and abstract concepts (people, places, things, text, numbers, etc...). However, not everything needs to be modeled using classes. Functions are commonly more than sufficient for a given task.
Example:
dev_conf.json
{
"ports": 8000,
"host": "0.0.0.0"
}
Class-based:
import json
class Configuration(dict):
def __init__(self, source):
self.source = source
def fetch(self):
with open(self.source, 'r') as source:
self.update(json.loads(source.read()))
conf = Configuration('dev_conf.json')
conf.fetch()
print(conf)
Back
Submit


Python: Inheritance - QA Plat
Instructions - platform
about:blank
1h 34m
Class-based:
import json
class Configuration(dict):
def __init__(self, source):
self.source = source
def fetch(self):
with open(self.source, 'r') as source:
self.update(json.loads(source.read()))
conf = Configuration('dev_conf.json')
conf.fetch()
print(conf)
# {'ports': 8000, 'host': '0.0.0.0'}
Function-based:
def configuration_from(source):
with open(source, 'r') as f:
return json.loads(f.read())
print(configuration_from('dev_conf.json'))
# {'ports': 8000, 'host': '0.0.0.0'}
Both implementations produce the same output however, the function-based approach is the more simple implementation. Simple code reduces the surface area for bugs to hide as well as maintenance burdens.
Report an issue
Back
Submit