TestpassportQ&A

Bessere Qualität , bessere Dienstleistungen!

We offer free update service for one year

Http://www.testpassport.ch

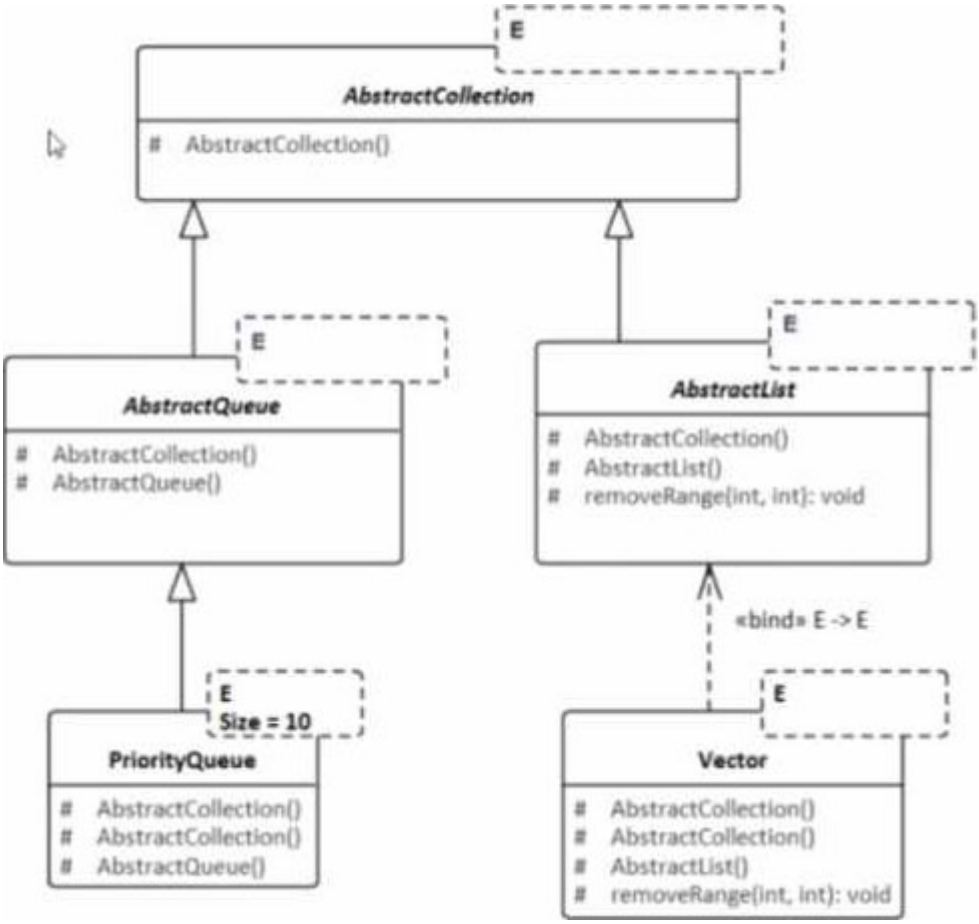
Exam : **OMG-OCUP2-ADV300**

Title : OMG Certified UML Professional 2 (OCUP 2) - Advanced Level

Version : DEMO

1.Which feature of a UML model element could NOT be adapted by a Stereotype?
A. Notation could be changed.
B. Constraints could be added
C. Constraints can be removed.
D. Attributes and Operations could be added.
Answer: C
Explanation:
A stereotype in UML is a powerful extension mechanism that allows developers to tailor UML models for particular domains or platforms. Stereotypes can adapt UML model elements by adding constraints, changing notation, and adding attributes and operations. However, they cannot remove existing constraints from a model element12.
Notation Change (A): Stereotypes can indeed change the notation of a model element to make it more expressive or domain-specific. For example, a stereotype could be used to visually distinguish between different kinds of classes in a class diagram1.
Adding Constraints (B): Stereotypes can add new constraints to a model element to specify additional rules or requirements that are not defined by the standard UML1.
Removing Constraints ©: This is not a capability of stereotypes. Constraints define rules that must be followed, and once they are part of a model element, they cannot be removed by a stereotype. Instead, they are an intrinsic part of the model's definition1.
Adding Attributes and Operations (D): Stereotypes can be used to add attributes and operations to a model element, which allows for the specification of additional properties or behaviors that are not part of the standard UML metamodel1.
In summary, while stereotypes are versatile and can extend the capabilities of UML model elements in various ways, they do not have the ability to remove constraints that are already applied to those elements. This is because constraints are considered fundamental rules that govern the integrity of the model, and removing them would potentially violate the model's correctness or completeness1.

2.A framework developer has been given a requirement to create an extensible utility for solution developers to use to create collections.
The framework developer has submitted the following diagram fragment for review:



Which review comment is valid and applicable?

A. The upper limit of 10 for the Size of PhontyQueue is too small and should be increased to at least 20 to accommodate special cases.

B. The «bind» relationship between the concrete class Vector and the abstract class AbstractList is incorrect. It should be a Generalization relationship.

C. The Generalization relationship between the concrete class PriorityOueue and the abstract class AbstractQueue is incorrect. It should be a «bmd» relationship.

D. The template parameter Size cannot be added to a specialized class. It needs to be moved to the top of the hierarchy and added to AbstractCollection and AbstractQueue.

Answer: B

Explanation:

In UML, the «bind» relationship is used to specify that a class is a template instantiation of a template class, where actual parameters are bound to the formal parameters of the template class1. However, in the case of the relationship between a concrete class like Vector and an abstract class like AbstractList, the correct relationship should be Generalization, not «bind».

Generalization is a taxonomic relationship between a more general classifier and a more specific classifier. Each instance of the specific classifier is also an instance of the general classifier1.

Thus, Vector being a concrete implementation of AbstractList, should inherit from AbstractList, which is correctly represented by a Generalization relationship in UML.

The other options can be evaluated as follows:

Option A: The upper limit for the size of PhontyQueue is a design decision that should be based on the requirements and use cases of the application. It is not inherently incorrect in UML to have a specific

upper limit.
Option C: The Generalization relationship is correctly used
between PriorityQueue and AbstractQueue as it represents inheritance in UML.
Option D: While it is true that template parameters are typically defined at the top of the hierarchy, the statement is too absolute. In UML, template parameters can be added to specialized classes, but it depends on the specific design and requirements. Therefore, without additional context, this statement cannot be deemed universally valid.
In conclusion, the most applicable and valid review comment is option B, which correctly identifies the misuse of the «bind» relationship in the context of the class diagram provided.

3.Which statement should be taken into consideration when extending a UML metaclass with a stereotype?
A. UML recommends to start extending the metaclass "Class" and then other metaclasses depending on the expected qualities of the profile.
B. The choice of the extended metaclass is not that important since tools can always apply a profile's filtering rules to hide unneeded metaclasses.
C. The metaclass and the stereotype that extends it should be semantically related to each other to avoid having to constrain the metamodel excessively.
D. UML specifies rules on how the mapping between stereotypes and UML metaclasses should be done; these rules must be followed to identify the best metaclasses.
Answer: C
Explanation:
When extending a UML metaclass with a stereotype, it is critical to ensure that the metaclass and the stereotype are semantically related. This is because a stereotype is a way to extend the UML metamodel to create new kinds of model elements that can include additional semantics and constraints, but still adhere to the base behavior defined by the metaclass. The stereotype should be a meaningful specialization of the metaclass and not contradict its fundamental semantics. By keeping them semantically related, there is less need for additional constraints on the metamodel, and the resulting profile is more intuitive for users. This is consistent with the principles described in the UML 2 Specification, particularly in the sections on profiles and stereotypes.

4.An organization has determined that they want to add the capability to create and add requirement elements to their UML models. They also want to create a unique relationship for tracing requirements to other model elements.
What is the appropriate approach to do this?
A. Use the requirement element and relationship defined in the UML specification.
B. Create a profile that stereotypes Class as requirement and Dependency as the relationship.
C. Create a new MOF metamodel that includes UML and adds the desired requirement element and relationship.
D. Assign tag values that ascribe the desired requirement type to a UML Requirement and Dependency relationship.
Answer: B
Explanation:
UML allows the introduction of new concepts that are not part of the standard UML metamodel by

creating a profile. To add capabilities for modeling requirements and tracing relationships in UML, a profile can be created where a Class is stereotyped to represent a requirement, and a Dependency is stereotyped to represent the trace relationship. This approach is both practical and conforms to the UML standard's mechanisms for extending the language. It is a common practice to create such profiles for requirements engineering within the UML framework. This conforms to the UML 2 Superstructure Specification, which provides guidelines on creating and applying profiles and stereotypes.

5.Where does UML explicitly intend String Expression elements to be used?
A. as (he ValueSpecifications for the nameExpressions of ParameterableElements within Template specifications
B. as the model the author chooses for the specification of custom dynamically-generated names for any NamedElement
C. whenever an OpaqueExpression form of a ValueSpecification needs to specify an expression that operates on String instances
D. The specification has no metaclass StnngExpression and so no use of StringExpression is explicitly intended.
Answer: D
Explanation:
In UML, ValueSpecifications are used to specify the value of an element. The UML 2 Specification does not define a metaclass named StringExpression. Instead, it provides a metaclass named OpaqueExpression, which can be used when an expression is written in a language that is not directly interpretable by the model. Since there is no metaclass called StringExpression in the UML 2 Specification, there is no explicitly intended use for it within the UML metamodel. The absence of this metaclass suggests that any use of a concept called "StringExpression" would not conform to standard UML 2 practices and would likely be part of an extension or profile, not the core UML metamodel.