

Документ подписан простой электронной подписью
 Информация о владельце:
 ФИО: Косенок Сергей Михайлович
 Должность: РПНТОЗ
 Дата подписания: 18.06.2024 18:22:55
 Уникальный программный ключ:
 e3a68f3eaa1e62674b54f4998099d3d6bfdcf836

Тестовое задание для диагностического тестирования по дисциплине:

Программирование на языках 4 GL, 5 семестр

Код, направление подготовки	09.03.01 Информатика и вычислительная техника
Направленность (профиль)	Автоматизированные системы обработки информации и управления
Форма обучения	Очная, заочная, очно-заочная
Кафедра разработчик	Автоматизированных систем обработки информации и управления
Выпускающая кафедра	Автоматизированных систем обработки информации и управления

Проверяемая компетенция	Задание	Варианты ответов	Тип сложности вопроса	Кол-во баллов за правильный ответ
ОПК-2.1 ОПК-2.2 ОПК-2.3 ОПК-8.1 ОПК-8.2 ОПК-8.3 ПК-3.1 ПК-3.2 ПК-3.3 ПК-6.1 ПК-6.2 ПК-6.3 ПК-7.1 ПК-7.2 ПК-7.3 ПК-11.1 ПК-11.2 ПК-11.3	Класс - это ...		Низкий	2

ОПК-2.1 ОПК-2.2 ОПК-2.3 ОПК-8.1 ОПК-8.2 ОПК-8.3 ПК-3.1 ПК-3.2 ПК-3.3 ПК-6.1 ПК-6.2 ПК-6.3 ПК-7.1 ПК-7.2 ПК-7.3 ПК-11.1 ПК-11.2 ПК-11.3	Объект - это ...		Низкий	2
ОПК-2.1 ОПК-2.2 ОПК-2.3 ОПК-8.1 ОПК-8.2 ОПК-8.3 ПК-3.1 ПК-3.2 ПК-3.3 ПК-6.1 ПК-6.2 ПК-6.3 ПК-7.1 ПК-7.2 ПК-7.3 ПК-11.1 ПК-11.2 ПК-11.3	Какой язык является объектно- ориентированным	1. Assembler 2. Prolog 3. C 4. C++	Низкий	2

<p>ОПК-2.1 ОПК-2.2 ОПК-2.3 ОПК-8.1 ОПК-8.2 ОПК-8.3 ПК-3.1 ПК-3.2 ПК-3.3 ПК-6.1 ПК-6.2 ПК-6.3 ПК-7.1 ПК-7.2 ПК-7.3 ПК-11.1 ПК-11.2 ПК-11.3</p>	<p>Объектно-ориентированное программирование - это ...</p>	<p>1. методология программирования, основанная на представлении программы в виде совокупности логических функций 2. методология программирования, основанная на представлении программы в виде совокупности моделей, каждый из которых является экземпляром определённого шаблона, а шаблоны образуют иерархию наследования 3. методология программирования, основанная на представлении программы в виде модулей 4. методология программирования, основанная на представлении программы в виде совокупности объектов, каждый из которых является экземпляром определённого класса, а классы образуют иерархию наследования</p>	<p>Низкий</p>	<p>2</p>
---	--	---	---------------	----------

ОПК-2.1 ОПК-2.2 ОПК-2.3 ОПК-8.1 ОПК-8.2 ОПК-8.3 ПК-3.1 ПК-3.2 ПК-3.3 ПК-6.1 ПК-6.2 ПК-6.3 ПК-7.1 ПК-7.2 ПК-7.3 ПК-11.1 ПК-11.2 ПК-11.3	В объектно-ориентированном программировании число является	1. объектом 2. типом 3. переменной 4. полем	Низкий	2
ОПК-2.1 ОПК-2.2 ОПК-2.3 ОПК-8.1 ОПК-8.2 ОПК-8.3 ПК-3.1 ПК-3.2 ПК-3.3 ПК-6.1 ПК-6.2 ПК-6.3 ПК-7.1 ПК-7.2 ПК-7.3 ПК-11.1 ПК-11.2 ПК-11.3	Максимальное количество деструкторов в классе		Средний	5
ОПК-2.1 ОПК-2.2 ОПК-2.3 ОПК-8.1 ОПК-8.2 ОПК-8.3 ПК-3.1 ПК-3.2 ПК-3.3	Сопоставьте ключевые слова в C#	1. class <=> virtual 2. method <=> operator 3. static <=> abstract	Средний	5

ПК-6.1 ПК-6.2 ПК-6.3 ПК-7.1 ПК-7.2 ПК-7.3 ПК-11.1 ПК-11.2 ПК-11.3				
ОПК-2.1 ОПК-2.2 ОПК-2.3 ОПК-8.1 ОПК-8.2 ОПК-8.3 ПК-3.1 ПК-3.2 ПК-3.3 ПК-6.1 ПК-6.2 ПК-6.3 ПК-7.1 ПК-7.2 ПК-7.3 ПК-11.1 ПК-11.2 ПК-11.3	Основные термины объектно- ориентированного программирования	1. класс 2. граф 3. сеть 4. объект	Средний	5

<p>ОПК-2.1 ОПК-2.2 ОПК-2.3 ОПК-8.1 ОПК-8.2 ОПК-8.3 ПК-3.1 ПК-3.2 ПК-3.3 ПК-6.1 ПК-6.2 ПК-6.3 ПК-7.1 ПК-7.2 ПК-7.3 ПК-11.1 ПК-11.2 ПК-11.3</p>	<p>... - это концепция объектно-ориентированного программирования, согласно которой абстрактный тип данных может наследовать данные и функциональность некоторого существующего типа, способствуя повторному использованию компонентов программного обеспечения</p>		Средний	5
<p>ОПК-2.1 ОПК-2.2 ОПК-2.3 ОПК-8.1 ОПК-8.2 ОПК-8.3 ПК-3.1 ПК-3.2 ПК-3.3 ПК-6.1 ПК-6.2 ПК-6.3 ПК-7.1 ПК-7.2 ПК-7.3 ПК-11.1 ПК-11.2 ПК-11.3</p>	<p>... - это использование только тех характеристик объекта, которые с достаточной точностью представляют его в данной системе</p>		Средний	5

<p>ОПК-2.1 ОПК-2.2 ОПК-2.3 ОПК-8.1 ОПК-8.2 ОПК-8.3 ПК-3.1 ПК-3.2 ПК-3.3 ПК-6.1 ПК-6.2 ПК-6.3 ПК-7.1 ПК-7.2 ПК-7.3 ПК-11.1 ПК-11.2 ПК-11.3</p>	<p>Абстракция - это ...</p>	<p>1. использование всех характеристик объекта, которые представлены в данной системе 2. использование только эффективных характеристик объекта, которые имеются в данной системе 3. использование только не эффективных характеристик объекта, которые имеются в данной системе 4. использование только тех характеристик объекта, которые с достаточной точностью представляют его в данной системе</p>	<p>Средний</p>	<p>5</p>
--	-----------------------------	--	----------------	----------

<p>ОПК-2.1 ОПК-2.2 ОПК-2.3 ОПК-8.1 ОПК-8.2 ОПК-8.3 ПК-3.1 ПК-3.2 ПК-3.3 ПК-6.1 ПК-6.2 ПК-6.3 ПК-7.1 ПК-7.2 ПК-7.3 ПК-11.1 ПК-11.2 ПК-11.3</p>	<p>Инкапсуляция - это ...</p>	<p>1. механизм сокрытия, позволяющий разграничивать доступ к различным частям компонента 2. механизм переадресации, позволяющий осуществлять доступ к различным компонентам 3. правила или утверждения, позволяющий разграничивать доступ к различным частям компонента 4. правила сокрытия, позволяющий разграничивать доступ к различным частям компонента</p>	<p>Средний</p>	<p>5</p>
---	-------------------------------	--	----------------	----------

<p>ОПК-2.1 ОПК-2.2 ОПК-2.3 ОПК-8.1 ОПК-8.2 ОПК-8.3 ПК-3.1 ПК-3.2 ПК-3.3 ПК-6.1 ПК-6.2 ПК-6.3 ПК-7.1 ПК-7.2 ПК-7.3 ПК-11.1 ПК-11.2 ПК-11.3</p>	<p>Наследование - это ...</p>	<p>1. правила объектно-ориентированного программирования, согласно которой абстрактный тип данных может наследовать данные и функциональность некоторого существующего типа, способствуя повторному использованию компонентов программного обеспечения 2. механизм объектно-ориентированного программирования, согласно которой абстрактный тип данных может наследовать данные и функциональность некоторого существующего типа, способствуя повторному использованию компонентов программного обеспечения 3. концепция объектно-ориентированного программирования, согласно которой абстрактный тип данных может наследовать данные и функциональность некоторого существующего типа, способствуя</p>	<p>Средний</p>	<p>5</p>
---	-----------------------------------	---	----------------	----------

		<p>повторному использованию компонентов программного обеспечения</p> <p>4. концепция или механизм объектно- ориентированного программирования, согласно которой абстрактный тип данных может наследовать данные и функциональность некоторого существующего типа, способствуя повторному использованию компонентов программного обеспечения</p>		
--	--	---	--	--

<p>ОПК-2.1 ОПК-2.2 ОПК-2.3 ОПК-8.1 ОПК-8.2 ОПК-8.3 ПК-3.1 ПК-3.2 ПК-3.3 ПК-6.1 ПК-6.2 ПК-6.3 ПК-7.1 ПК-7.2 ПК-7.3 ПК-11.1 ПК-11.2 ПК-11.3</p>	<p>Полиморфизм - это ...</p>	<p>1. способность функции обрабатывать данные разных типов 2. способность функции или предиката обрабатывать данные разных типов 3. способность функции обрабатывать данные разных подтипов 4. способность предиката обрабатывать данные разных типов</p>	<p>Средний</p>	<p>5</p>
--	-----------------------------------	--	----------------	----------

<p>ОПК-2.1 ОПК-2.2 ОПК-2.3 ОПК-8.1 ОПК-8.2 ОПК-8.3 ПК-3.1 ПК-3.2 ПК-3.3 ПК-6.1 ПК-6.2 ПК-6.3 ПК-7.1 ПК-7.2 ПК-7.3 ПК-11.1 ПК-11.2 ПК-11.3</p>	<p>Расшифруйте аббревиатуру SOLID</p>	<p>1. single data, open– closed, Liskov substitution, interface segregation и dependency inversion 2. single responsibility, open– closed, Liskov substitution, interface segregation и dependency inversion 3. single responsibility, open– closed, Liskov substitution, interface segregation и dependency injection 4. single responsibility, open– connect, Liskov substitution, interface segregation и dependency inversion</p>	<p>Средний</p>	<p>5</p>
--	---	---	----------------	----------

<p>ОПК-2.1 ОПК-2.2 ОПК-2.3 ОПК-8.1 ОПК-8.2 ОПК-8.3 ПК-3.1 ПК-3.2 ПК-3.3 ПК-6.1 ПК-6.2 ПК-6.3 ПК-7.1 ПК-7.2 ПК-7.3 ПК-11.1 ПК-11.2 ПК-11.3</p>	<p>К принципам SOLID относиться</p>	<p>1. Liskov substitution, interface segregation и dependency inversion 2. single responsibility, open-closed 3. Liskov substitution, interface segregation и dependency injection 4. single data, open-closed</p>	<p>Высокий</p>	<p>8</p>
<p>ОПК-2.1 ОПК-2.2 ОПК-2.3 ОПК-8.1 ОПК-8.2 ОПК-8.3 ПК-3.1 ПК-3.2 ПК-3.3 ПК-6.1 ПК-6.2 ПК-6.3 ПК-7.1 ПК-7.2 ПК-7.3 ПК-11.1 ПК-11.2 ПК-11.3</p>	<p>Расставьте фрагменты кода на C# в правильном порядке</p>	<p>1. Car 2. class 3. public 4. } 5. { 6. public Car() 7. } 8. {</p>	<p>Высокий</p>	<p>8</p>

<p>ОПК-2.1 ОПК-2.2 ОПК-2.3 ОПК-8.1 ОПК-8.2 ОПК-8.3 ПК-3.1 ПК-3.2 ПК-3.3 ПК-6.1 ПК-6.2 ПК-6.3 ПК-7.1 ПК-7.2 ПК-7.3 ПК-11.1 ПК-11.2 ПК-11.3</p>	<p>Расставьте фрагменты кода на С# в правильном порядке</p>	<ol style="list-style-type: none"> 1. public 2. { 3. void 4. Run() 5. static 6. } 	<p>Высокий</p>	<p>8</p>
<p>ОПК-2.1 ОПК-2.2 ОПК-2.3 ОПК-8.1 ОПК-8.2 ОПК-8.3 ПК-3.1 ПК-3.2 ПК-3.3 ПК-6.1 ПК-6.2 ПК-6.3 ПК-7.1 ПК-7.2 ПК-7.3 ПК-11.1 ПК-11.2 ПК-11.3</p>	<p>Расставьте фрагменты кода на С# в правильном порядке</p>	<ol style="list-style-type: none"> 1. { 2. Cat 3. } 4. {} 5. private void 6. internal 7. Jump() 8. class 	<p>Высокий</p>	<p>8</p>

ОПК-2.1 ОПК-2.2 ОПК-2.3 ОПК-8.1 ОПК-8.2 ОПК-8.3 ПК-3.1 ПК-3.2 ПК-3.3 ПК-6.1 ПК-6.2 ПК-6.3 ПК-7.1 ПК-7.2 ПК-7.3 ПК-11.1 ПК-11.2 ПК-11.3	Расставьте фрагменты кода на С# в правильном порядке	1. } 2. string Name {get;set;} 3. { 4. private 5. IAnimal 6. interface	Высокий	8
---	---	--	---------	---