

**Тестовое задание для диагностического тестирования по дисциплине**

**Объектно-ориентированное программирование**

Семестр 3

Код, направление подготовки	09.03.04 Программная инженерия
Направленность (профиль)	Программное обеспечение компьютерных систем
Форма обучения	очная
Кафедра-разработчик	автоматики и компьютерных систем
Выпускающая кафедра	автоматики и компьютерных систем

№	Проверяемая компетенция	Тип вопроса	Задание	Варианты ответов	Тип сложности вопроса
1.	ОПК-2.1 ОПК-6.1 ОПК-6.3	Множественный выбор	Каких типов данных не существует в языке C++?	<ol style="list-style-type: none"> <li>1. unsigned char</li> <li>2. long double</li> <li>3. long char</li> <li>4. short int</li> <li>5. short float</li> <li>6. unsigned double</li> </ol>	низкий
2.	ОПК-2.1 ОПК-6.1 ОПК-6.3	Вставить слово	Зарезервированное слово в C++ является _____(1–4) и его использование _____(5–6) в каком-либо ином смысле, кроме определенного в _____(7–9).	<ol style="list-style-type: none"> <li>1. ключевым словом</li> <li>2. идентификатором</li> <li>3. оператором</li> <li>4. константой</li> <li>5. не допускается</li> <li>6. допускается</li> <li>7. данном языке</li> <li>8. данной программе</li> <li>9. данном классе</li> </ol>	низкий
3.	ОПК-2.1 ОПК-6.1 ОПК-6.3	Один из	Что в C++ означает следующее выражение: cout << x << endl;	<ol style="list-style-type: none"> <li>1. значение переменной x находится строго между значениями переменных cout и endl</li> <li>2. значение переменной x будет выведено на экран с последующим переходом на новую строку</li> <li>3. двоичный сдвиг значений переменных cout и x на endl разрядов</li> <li>4. такое выражение не имеет смысла в C++</li> </ol>	низкий
4.	ОПК-2.1 ОПК-6.1 ОПК-6.3	Один из	Какое определение термина «лексема» является правильным?	<ol style="list-style-type: none"> <li>1. минимальная единица языка, имеющая самостоятельный смысл</li> <li>2. законченное действие в программе</li> <li>3. новый тип данных, определенный разработчиком в тексте программы</li> <li>4. элемент общедоступного интерфейса класса</li> </ol>	низкий

5.	ОПК-2.1 ОПК-6.1 ОПК-6.3	Один из	Какую функцию должна содержать любая программа на языке С++?	<ol style="list-style-type: none"> <li>1. main()</li> <li>2. main++()</li> <li>3. cpp_main()</li> <li>4. public static void main()</li> </ol>	низкий
6.	ОПК-2.1 ОПК-6.1 ОПК-6.3	Все или ничего	Каких модификаторов доступа не существует в С++?	<ol style="list-style-type: none"> <li>1. protected</li> <li>2. public</li> <li>3. included</li> <li>4. private</li> <li>5. priority</li> </ol>	средний
7.	ОПК-2.1 ОПК-6.1 ОПК-6.3	Вставить слово	<p>Вставьте пропущенные строки в следующей программе на С++:</p> <pre> _____ (1-3) _____ (4-6) int main() {     cout &lt;&lt; "Hello, World!" &lt;&lt; endl;     _____ (7-9) } </pre>	<ol style="list-style-type: none"> <li>1. #include &lt;stdio.h&gt;</li> <li>2. #include &lt;cstdio&gt;</li> <li>3. #include &lt;iostream&gt;</li> <li>4. using std library;</li> <li>5. using namespace std;</li> <li>6. import std.*</li> <li>7. return 0;</li> <li>8. end main;</li> <li>9. break;</li> </ol>	средний
8.	ОПК-2.1 ОПК-6.1 ОПК-6.3	Один из	Что такое конструктор в С++?	<ol style="list-style-type: none"> <li>1. специальный метод класса, который выделяет память при создании объекта</li> <li>2. специальный метод класса, который предназначен для инициализации полей данных объекта некоторыми начальными значениями</li> <li>3. набор элементов, из которых конструируются объекты классов</li> <li>4. способ описания новых типов данных</li> </ol>	средний

9.	ОПК-2.1 ОПК-6.1 ОПК-6.3	Один из	<p>В программе массив был создан динамически следующим образом:  <code>int *p = new int[1000];</code>          Как правильно его уничтожить?</p>	<ol style="list-style-type: none"> <li>1. <code>delete p;</code></li> <li>2. <code>delete[] p;</code></li> <li>3. <code>delete p[];</code></li> <li>4. <code>delete p[1000]</code></li> <li>5. <code>delete int[1000]</code></li> <li>6. <code>free(p);</code></li> <li>7. он уничтожится автоматически после использования</li> <li>8. такой массив нельзя уничтожить</li> </ol>	средний
10.	ОПК-2.1 ОПК-6.1 ОПК-6.3	Один из	<p>Что такое инкапсуляция?</p>	<ol style="list-style-type: none"> <li>1. Свойство языка программирования, позволяющее объединить данные и методы в одном объекте и защитить их от внешнего вмешательства</li> <li>2. Возможность использовать одно и то же имя для нескольких объектов или функций</li> <li>3. Обобщения множества классов и выделение общего базового класса</li> <li>4. Защита текста программы от наблюдения и взлома</li> </ol>	средний

11.	ОПК-2.1 ОПК-6.1 ОПК-6.3	Вставить слово	<p>Как правильно вызвать метод Out() объекта класса Something в каждом из случаев:</p> <pre> class Something {     ...     public:     void Out() const;     void Do     {         _____     } }; void func(const Something &amp; a) {     _____ } int main() {     Something *a = new Something;     _____     return 0; } </pre>	<ol style="list-style-type: none"> <li>1. a.Out();</li> <li>2. a-&gt;Out();</li> <li>3. Out();</li> <li>4. a::Out();</li> <li>5. здесь метод Out() недоступен</li> </ol>	средний
12.	ОПК-2.1 ОПК-6.1 ОПК-6.3	Один из	Сколько параметров можно передать в деструктор?	<ol style="list-style-type: none"> <li>1. любое количество</li> <li>2. ровно один параметр</li> <li>3. 0 или 1 параметр в зависимости от ситуации</li> <li>4. ни одного, деструктор не имеет параметров</li> </ol>	средний

13.	ОПК-2.1 ОПК-6.1 ОПК-6.3	Один из	Что такое полиморфизм?	<ol style="list-style-type: none"> <li>1. Механизм, который объединяет данные и код в одном объекте и защищающий их от внешнего вмешательства</li> <li>2. Механизм, позволяющий единообразно обрабатывать объекты разных типов за счет наличия у них одного и того же интерфейса</li> <li>3. Механизм, позволяющий использовать одно и то же имя для нескольких объектов или функций</li> <li>4. Механизм, который позволяет обращаться к одному объекту, используя разные имена</li> </ol>	средний
14.	ОПК-2.1 ОПК-6.1 ОПК-6.3	Один из	Перегрузкой функций называют:	<ol style="list-style-type: none"> <li>1. вызов функции с передачей ей большего количества параметров, чем она может принять</li> <li>2. использование одного имени для нескольких функций, отличающихся списком формальных параметров</li> <li>3. слишком частые вызовы одной функции, из-за чего она начинает давать сбои</li> <li>4. реализацию в одной функции слишком сложного алгоритма, который требует много времени для его выполнения</li> </ol>	средний
15.	ОПК-2.1 ОПК-6.1 ОПК-6.3	Один из	Что такое виртуальный метод?	<ol style="list-style-type: none"> <li>1. Метод, который не имеет тела (отсутствует реализация метода)</li> <li>2. Метод базового класса, который может быть переопределен (замещен) в производном классе</li> <li>3. Любой метод, который был унаследован от базового класса</li> <li>4. Метод, имеющий такое же имя, как сам класс, и поэтому вызываемый неявно</li> </ol>	средний

16.	ОПК-2.1 ОПК-6.1 ОПК-6.3	Все или ничего	Для некоторого класса Number необходимо перегрузить бинарную операцию «+» («плюс», сложение), которая не изменяет значения своих операндов. Какие из приведенных вариантов являются правильными?	<ol style="list-style-type: none"> <li>1. как функцию: <code>Number operator+ (const Number &amp; b);</code></li> <li>2. как функцию: <code>Number operator+ (const Number &amp; b) const;</code></li> <li>3. как функцию: <code>Number operator+ (const Number &amp; a, const Number &amp; b);</code></li> <li>4. как функцию: <code>Number operator+ (const Number &amp; a, const Number &amp; b) const;</code></li> <li>5. как метод: <code>Number operator+ (const Number &amp; b);</code></li> <li>6. как метод: <code>Number operator+ (const Number &amp; b) const;</code></li> <li>7. как метод: <code>Number operator+ (const Number &amp; a, const Number &amp; b);</code></li> <li>8. как метод: <code>Number operator+ (const Number &amp; a, const Number &amp; b) const;</code></li> </ol>	ВЫСОКИЙ
17.	ОПК-2.1 ОПК-6.1 ОПК-6.3	Все или ничего	Чему может соответствовать следующий прототип: <code>Number operator- (Number &amp; x);</code>	<ol style="list-style-type: none"> <li>1. перегрузка унарной операции «-» в виде функции</li> <li>2. перегрузка унарной операции «-» в виде метода класса</li> <li>3. перегрузка бинарной операции «-» в виде функции</li> <li>4. перегрузка бинарной операции «-» в виде метода класса</li> </ol>	ВЫСОКИЙ

18.	ОПК-2.1 ОПК-6.1 ОПК-6.3	Один из	<p>В каком (каких) из блоков будет обработано исключение в следующем фрагменте кода:</p> <pre> try {     throw 1; } catch (Error &amp; e) {     ...    // 1 } catch (int e) {     ...    // 2 } catch (...) {     ...    // 3 } </pre>	<ol style="list-style-type: none"> <li>1. в блоке 1</li> <li>2. в блоке 2</li> <li>3. в блоке 3</li> <li>4. во всех по очереди</li> <li>5. ни в каком из приведенных</li> <li>6. здесь не будет исключения</li> </ol>	высокий
19.	ОПК-2.1 ОПК-6.1 ОПК-6.3	Один из	<p>Если в описании класса не указан ни один модификатор доступа, то все его элементы будут являться:</p>	<ol style="list-style-type: none"> <li>1. частными (закрытыми)</li> <li>2. защищенными</li> <li>3. общедоступными (открытыми)</li> <li>4. абстрактными</li> <li>5. это приведет к ошибке компиляции</li> </ol>	высокий
20.	ОПК-2.1 ОПК-6.1 ОПК-6.3	Один из	<p>В каких случаях на языке C++ правильно объявлен класс Derive, производный от некоторых классов А и В?</p>	<ol style="list-style-type: none"> <li>1. Derive = class(A, B) { ... };</li> <li>2. class Derive : public A, public B { ... };</li> <li>3. class Derive extends A, B { ... };</li> <li>4. в C++ нет множественного наследования, поэтому объявить такой класс невозможно</li> </ol>	высокий