1. **В чем различие между классом и объектом?**

|  |
| --- |
|  |

1. **Верно ли утверждение, что класс не может быть создан без полей и методов?**
2. да
3. нет
4. **Напишите формулу объявления класса в Java.**

|  |
| --- |
|  |

1. **Исправьте код ниже, если это необходимо для запуска приложения.**

|  |
| --- |
| **public** **class** MyClass  {  **private** String someStr = "String";    // остальной код класса...  } |
| **public** **class** Main  {  **public** **static** **void** main(String[] args)  {  System.***out***.println(MyClass.someStr());  }  } |

1. **Какой оператор используется для возврата из метода? Приведите примеры его использования.**

|  |
| --- |
|  |

1. **Какие существуют виды инициализации полей?**
2. Инициализация полей в месте использования
3. Инициализация в конструкторе
4. Инициализация в статическом блоке
5. Инициализация в нестатическом блоке
6. Иницилизация «по умолчанию»
7. **Сколько конструкторов может быть в классе?**
8. Только один
9. Один и более
10. Ни одного
11. **Нужно ли вызывать конструктор суперкласса из класса-потомка? Если да, то в каких случаях?**

|  |
| --- |
|  |

1. **В чем состоит суть отношения «is-a»?**

|  |
| --- |
|  |

1. **Как организовать наследование в Java?**

|  |
| --- |
|  |

1. **Перечислите модификаторы, отвечающие за уровень доступа.**

|  |
| --- |
|  |

1. **Можно ли изменять модификатор доступа метода в «родственных» классах?**
2. да
3. нет
4. **Выберите верное суждение:**
5. Интерфейсы могут быть наследниками классов
6. По умолчанию все интерфейсы «public»
7. Интерфейсы могут организовывать иерархию наследования
8. Все методы по умолчанию «abstract»
9. Интерфейс может не содержать методов вообще
10. **Отредактируйте следующий код, если это необходимо для успешной компиляции программы.**

|  |
| --- |
| **package** com.tsystems.entity;  **public** **interface** Readable  {  read();  } |
| **package** com.tsystems.entity;  **public** **class** ParentClass  {  **protected** String name;  **public** ParentClass(String aName)  {  name=aName;  }  } |
| **package** com.tsystems;  **public** **class** ChildClass **extends** ParentClass **implements** Readable  {  **public** ChildClass()  {  **super**(aName);  }  **public** ChildClass(String aName)  {}  } |

1. **Зачем нужно динамическое и статическое связывание?**

|  |
| --- |
|  |

1. **В каких случаях требуется явное приведение ссылочных типов, а в каких – нет?**

|  |
| --- |
|  |

1. **???????Проанализируйте код из вопроса 14: каким типам можно привести объект типа** ChildClass**?**

|  |
| --- |
|  |

1. **???????В сигнатуру метода входят:**
2. Имена параметров
3. Типы параметров
4. Имя класса
5. Имя метода
6. Тип возвращаемого значения
7. Модификатор доступа метода
8. **???????Укажите верный вариант перегрузки методов:**

|  |
| --- |
| **package** com.tsystems.entity;  **public** **class** ParentClass  {  **protected** String name;  **public** ParentClass(String aName)  {  name=aName;  }  **public** String read()  {  **return** **null**;  }  // вставить вариант перегрузки сюда  } |

|  |
| --- |
| // код класса ParentClass    **public** String read(String fileName)  {  **return** **null**;  }  **public** String read(String otherFileName)  {  **return “**otherFileName**”**;  }  // код класса ParentClass |

1. **???????**

|  |
| --- |
| // код класса ParentClass    **public** String read(String fileName)  {  **return** **read(**filename, “NoneFolder”**)**;  }  **private** String read(String otherFileName, String fileFolder)  {  **return “**otherFileName**”**;  }  // код класса ParentClass |

1. **???????**

|  |
| --- |
| // код класса ParentClass  **public** String read(**int** value)  {  **return** **read()**;  }  // код класса ParentClass |

1. **Возможно ли переопределить метод внутри одного и того же класса?**
2. да
3. нет
4. **Чем переопределение отличается от перегрузки?**

|  |
| --- |
|  |

1. **Возможно ли перегрузить интерфейсный метод в классе, реализующем данный интерфейс?**
2. да
3. нет
4. **???????Возможно ли перегрузить методы внутри интерфейса?**
5. да
6. нет