

Комплект оценочных материалов по дисциплине «Платформа.NET»

Задания закрытого типа

Задания закрытого типа на выбор правильного ответа

1. Выберите варианты правильного создания массива

A) `int[] array1 = new int[5];`

Б) `int[] array2 = [1, 2, 3, 4, 5, 6];`

В) `int[,] multiDimensionalArray1 = new int[2, 3];`

Г) `int[,] multiDimensionalArray2 = { { 1, 2, 3 }, { 4, 5, 6 } };`

Д) `int[][] jaggedArray = new int[6][];`

Е) `jaggedArray[0] = [1, 2, 3, 4];`

Правильный ответ: Все

Компетенции (индикаторы): ПК-2

2. Какой из следующих методов корректно принимает и выводит все элементы двумерного массива целых чисел?

A)

```
void PrintArray(int[,] array)
{
    for (int i = 0; i < array.Length; i++)
    {
        Console.WriteLine(array[i]);
    }
}
```

Б)

```
void PrintArray(int[,] array)
{
    for (int i = 0; i < array.GetLength(0); i++)
    {
        for (int j = 0; j < array.GetLength(1); j++)
        {
            Console.WriteLine(array[i, j]);
        }
    }
}
```

В)

```
void PrintArray(int[,] array)
{
    foreach (var item in array)
```

```

    {
        Console.WriteLine(item);
    }
}

```

Г)

```
void PrintArray(int[,] array)
```

```

{
    Console.WriteLine(array);
}

```

Правильный ответ: Б

Компетенции (индикаторы): ПК-2

3. Какой из следующих методов корректно использует массив параметров?

А) void MyMethod(int[] numbers) { }

Б) void MyMethod(params int[] numbers) { }

В) void MyMethod (string fmt, params object[] args) { }

Г) void MyMethod (params object[] args, string fmt) { }

Д) void MyMethod(params int numbers) { }

Е) void MyMethod(int numbers) { }

Правильный ответ: Б, В

Компетенции (индикаторы): ПК-2

4. Какой из следующих методов является методом экземпляра?

А) public static void MyStaticMethod() { }

Б) public void MyInstanceMethod() { }

В) public static int MyProperty { get; set; }

Г) private static void MyPrivateMethod() { }

Правильный ответ: Б

Компетенции (индикаторы): ПК-2

5. Что такое абстрактный класс?

А) Класс, который нельзя создать.

Б) Класс, который содержит только абстрактные методы.

В) Класс, который может содержать абстрактные методы и методы с реализацией.

Г) Класс, который не может наследоваться.

Правильный ответ: В

Компетенции (индикаторы): ПК-2

6. Какой из следующих методов не может быть переопределён?

```
class BaseClass
```

```

{
    public virtual void MyMethod()

```

```

    {
        // Some implementation
    }
}
class DerivedClass : BaseClass
{
    public sealed override void MyMethod()
    {
        // Another implementation
    }
}

```

A) Метод в классе BaseClass

Б) Метод в классе DerivedClass

В) Метод в классе, который наследует DerivedClass

Правильный ответ: В

Компетенции (индикаторы): ПК-2

7. Какое сочетание модификаторов корректно?

A) sealed virtual

Б) sealed abstract

В) sealed override

Г) все вышеперечисленные

Правильный ответ: В

Компетенции (индикаторы): ПК-2

8. Посмотрите на код ниже и выберите верное утверждение:

```

abstract class Person
{
    public abstract string Name { get; set; }
}
abstract class Manager : Person
{
}

```

A) Производный класс всегда обязан реализовать все абстрактные члены базового класса

Б) Производный класс может не реализовать все абстрактные члены базового класса, если производный класс определен как абстрактный

В) Абстрактный класс не может быть наследником другого абстрактного класса

Правильный ответ: Б

Компетенции (индикаторы): ПК-2

9. Выберите верные утверждения:

А) Один и тот же интерфейс может быть реализован в двух классах по-разному.

Б) Интерфейс представляет собой именованный набор абстрактных членов

В) В одном классе может быть реализовано любое количество интерфейсов

Г) Абстрактные классы могут иметь несколько родительских классов

Д) Один интерфейс не может наследовать другой

Правильный ответ: А, Б, В

Компетенции (индикаторы): ПК-2

Задания закрытого типа на установление соответствия

1. Установите правильное соответствие.

Определение		Значение	
1) Лямбда-выражения	А)	могут содержать реализацию	
2) Делегаты	Б)	не могут содержать никакой реализации	
3) Интерфейсы	В)	представляют такие объекты, которые указывают на методы	
4) Абстрактные классы	Г)	представляют упрощенную запись анонимных методов	

Правильный ответ:

1	2	3	4
Г	В	Б	А

Компетенции (индикаторы): ПК-2

2. Поставьте в соответствие методы и свойства класса FileInfo:

Определение		Значение	
1) Свойство Directory	А)	получает родительский каталог в виде объекта DirectoryInfo	
2) Свойство DirectoryName	Б)	получает полный путь к родительскому каталогу	
3) Свойство Extension	В)	получает расширение файла	

Правильный ответ:

1	2	3
А	Б	Б

Компетенции (индикаторы): ПК-2

3. XPath представляет язык запросов в XML. Он позволяет выбирать элементы, соответствующие определенному селектору. Поставьте в селектор в соответствие выполняемому действию.

Определение	Значение
1) . – выбор текущего узла;	А) выбор текущего узла;

- 2) .. – выбор родительского узла; Б) выбор родительского узла;
 3) * – выбор всех дочерних узлов В) выбор всех дочерних узлов
 текущего узла; текущего узла;
 4) @* – выбор всех атрибутов Г) выбор всех атрибутов текущего
 текущего узла; узла;

Правильный ответ:

1	2	3	4
А	Б	В	Г

Компетенции (индикаторы): ПК-2

Задания закрытого типа на установление правильной последовательности

1. Для сериализации (сохранения в форма xml) применяется метод Serialize(). Установите правильную последовательность передачи параметров в метод

- А) передается поток Stream
 Б) передается объект, который будет сохраняться в формат xml

Правильный ответ: А, Б

Компетенции (индикаторы): ПК-2

2. Установите правильную последовательность создания делегата его вызова?

- А) Action
 Б) <int>
 В) myDelegate
 Г) =
 Д) new
 Е) (Method)
 Ж) myDelegate(5)

Правильный ответ: А, Б, В, Г, Д, Е, Ж

Компетенции (индикаторы): ПК-2

Задания открытого типа

Задания открытого типа на дополнение

1. Тип класса, для которого объявлены параметры типа, называется _____ класса.

Правильный ответ: универсальным типом

Компетенции (индикаторы): ПК-2

2. Универсальный тип, для которого указаны аргументы типа, например, `Pair<int, string>`, называется _____.

Правильный ответ: сконструированным типом

Компетенции (индикаторы): ПК-2

_____ методом называется виртуальный метод без реализации.

Правильный ответ: Абстрактным

Компетенции (индикаторы): ПК-2

3. Для работы с каталогами в пространстве имен `System.IO` предназначены сразу два класса:_____.

Правильный ответ: `Directory` и `DirectoryInfo`

Компетенции (индикаторы): ПК-2

4. _____ – это процесс преобразования объекта в поток байтов для сохранения или передачи в память, в базу данных или в файл.

Правильный ответ: Сериализация

Компетенции (индикаторы): ПК-2

Задания открытого типа с кратким свободным ответом

1. Вопрос: Какой модификатор доступа делает члены класса доступными только внутри самого класса?

Правильный ответ: `private`

Компетенции (индикаторы): ПК-2

2. Вопрос: Какой метод используется для преобразования строки в целое число?

Правильный ответ: `int.Parse()`

Компетенции (индикаторы): ПК-2

3. Какой модификатор доступа позволяет членам класса быть доступными в пределах той же сборки и в производных классах?

Правильный ответ: `protected internal`

Компетенции (индикаторы): ПК-2

4. Что выведет следующий листинг кода?

```
public class Person
{
    public int age;
    public string name;
}
class TestPerson
{
```

```

static void Main()
{
    var person = new Person();
    Console.WriteLine($"Name: {person.name}, Age: {person.age}");
    Console.ReadKey();
}
}

```

Правильный ответ: Name: , Age: 0
 Компетенции (индикаторы): ПК-2

Задания открытого типа с развернутым ответом

1. Создайте класс, который будет использовать делегаты и события.
 Привести расширенное решение.
 Время выполнения – 30 мин.

Ожидаемый результат:

// MyClass.cs

using System;

public class MyClass

```

{
    public delegate void MyDelegate(string message);
    public event MyDelegate MyEvent;
    public void TriggerEvent(string message)
    {
        MyEvent?.Invoke(message);
    }
}

```

// MyClassTests.cs

using Microsoft.VisualStudio.TestTools.UnitTesting;

[TestClass]

public class MyClassTests

```

{
    private MyClass _myClass;
    private string _receivedMessage;
    [TestInitialize]
    public void Setup()
    {
        _myClass = new MyClass();
        _receivedMessage = string.Empty;
        // Подписка на событие с использованием анонимного метода
        _myClass.MyEvent += delegate (string message)
        {
            _receivedMessage = message;
        };
    }
}

```

```

    }
    [TestMethod]
    public void TestEventTriggered()
    {
        // Arrange
        string testMessage = "Hello, World!";
        // Act
        _myClass.TriggerEvent(testMessage);
        // Assert
        Assert.AreEqual(testMessage, _receivedMessage);
    }
    [TestMethod]
    public void TestEventNotTriggered()
    {
        // Arrange
        string initialMessage = _receivedMessage;
        // Act
        // Не вызываем событие
        // Assert
        Assert.AreEqual(initialMessage, _receivedMessage);
    }
}

```

Пояснение.

MyClass: Класс с делегатом и событием. Метод `TriggerEvent` вызывает событие, если на него есть подписчики.

MyClassTests: Класс тестов, который использует `MSTest` для проверки функциональности.

В методе `Setup` мы подписываемся на событие с помощью анонимного метода, который сохраняет полученное сообщение.

В методе `TestEventTriggered` мы проверяем, что событие корректно передает сообщение.

В методе `TestEventNotTriggered` мы проверяем, что если событие не вызывается, то сообщение остается пустым.

Критерии оценивания:

- создание класса `MyClass` с делегатом `MyDelegate(string message)` и событием для передачи сообщения;
- создание класса `MyClassTests`, который использует `MSTest` для проверки функциональности;
- создание метода `TestEventTriggered` для проверки корректной передачи сообщения;
- создание метода `TestEventNotTriggered` для проверки вызова события.

Компетенции (индикаторы): ПК-2

2. Создайте класс Person, который будет представлять человека, и затем выполните запись и извлечение объекта этого класса в XML.

Привести расширенное решение.

Время выполнения – 30 мин.

Ожидаемый результат:

Шаг 1: Определение класса

```
using System;
using System.IO;
using System.Xml.Serialization;
[Serializable] // Указываем, что класс может быть сериализован
public class Person
{
    public string Name { get; set; }
    public int Age { get; set; }
}
```

Шаг 2: Сериализация объекта в XML

```
public static void SerializeToXml(Person person, string filePath)
{
    XmlSerializer serializer = new XmlSerializer(typeof(Person));
    using (StreamWriter writer = new StreamWriter(filePath))
    {
        serializer.Serialize(writer, person);
    }
}
```

Шаг 3: Десериализация объекта из XML

```
public static Person DeserializeFromXml(string filePath)
{
    XmlSerializer serializer = new XmlSerializer(typeof(Person));
    using (StreamReader reader = new StreamReader(filePath))
    {
        return (Person)serializer.Deserialize(reader);
    }
}
```

Шаг 4: Пример использования

```
class Program
{
    static void Main(string[] args)
    {
        // Создаем объект Person
        Person person = new Person { Name = "Alice", Age = 30 };
        // Сериализуем объект в XML
        string filePath = "person.xml";
        SerializeToXml(person, filePath);
        Console.WriteLine("Объект сериализован в XML.");
    }
}
```

```

        // Десериализуем объект из XML
        Person deserializedPerson = DeserializeFromXml(filePath);
        Console.WriteLine($"Имя:    {deserializedPerson.Name},    Возраст:
{deserializedPerson.Age}");
    }
}

```

Пояснение.

Определение класса: Мы создаем класс Person с двумя свойствами: Name и Age. Атрибут [Serializable] указывает, что класс может быть сериализован.

Сериализация: Метод SerializeToXml принимает объект Person и путь к файлу, в который будет записан XML. Мы используем XmlSerializer для сериализации объекта в XML-формат.

Десериализация: Метод DeserializeFromXml читает XML из файла и преобразует его обратно в объект Person.

Пример использования: В методе Main мы создаем объект Person, сериализуем его в XML, а затем десериализуем его обратно и выводим на консоль.

Критерии оценивания:

- создание класса Person со свойствами и атрибутом, указывающим, что класс может быть сериализован;
- создание файла person.xml;
- использование метода SerializeToXml для сериализации объекта в XML-формат;
- использование метода DeserializeFromXml, который читает XML из файла и преобразует его обратно в объект Person;

Компетенции (индикаторы): ПК-2

Экспертное заключение

Представленный комплект оценочных материалов по дисциплине «Платформа.NET» соответствует требованиям ФГОС ВО.

Предлагаемые оценочные материалы адекватны целям и задачам реализации основной профессиональной образовательной программы по направлению подготовки 09.03.04 Программная инженерия.

Виды оценочных средств, включенные в представленный фонд, отвечают основным принципам формирования ФОС.

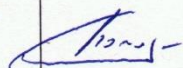
Разработанные и представленные для экспертизы оценочные материалы рекомендуются к использованию в процессе подготовки обучающихся по указанному направлению.

Председатель учебно-методической комиссии
института компьютерных систем
и информационных технологий



Ветрова Н.Н.

Лист изменений и дополнений

№ п/п	Виды дополнений и изменений	Дата и номер протокола заседания кафедры (кафедр), на котором были рассмотрены и одобрены изменения и дополнения	Подпись (с расшифровкой) заведующего кафедрой (заведующих кафедрами)
1.	Дополнен комплектом оценочных материалов	протокол заседания кафедры компьютерных систем и сетей № <u>8</u> от <u>10.03.2025</u>	 С.В. Попов