

**Комплект оценочных материалов по дисциплине  
«Технология.NET»**

**Задания закрытого типа**

**Задания закрытого типа на выбор правильного ответа**

1. Какой из методов класса File используется для записи текстовой строки в файл?

- А) File.WriteAllText()
- Б) File.AppendText()
- В) File.SaveText()
- Г) File.WriteText()

Правильный ответ: А

Компетенции (индикаторы): ОПК-4

2. Какой метод класса File можно использовать для проверки, существует ли файл?

- А) File.Exists()
- Б) File.IsExists()
- В) File.FileExists()
- Г) File.CheckExists()

Правильный ответ: А

Компетенции (индикаторы): ОПК-4

3. Какой из следующих конструкторов класса FileStream открывает файл для чтения и записи?

- А) new FileStream("file.txt", FileMode.Create)
- Б) new FileStream("file.txt", FileMode.Open, FileAccess.ReadWrite)
- В) new FileStream("file.txt", FileMode.Append)
- Г) new FileStream("file.txt", FileMode.Truncate)

Правильный ответ: Б

Компетенции (индикаторы): ОПК-4

4. Какой класс в C# используется для записи текстовых данных в файл?

- А) FileReader
- Б) FileWriter
- В) StreamReader
- Г) StreamWriter

Правильный ответ: Г

Компетенции (индикаторы): ОПК-4

5. Как добавить новый элемент Book с атрибутом id в существующий XmlDocument?

- A) document.Add(new XElement("Book", new XAttribute("id", 1)));
- Б) document.Element("Books").Add(new XElement("Book", new XAttribute("id", 1)));

В) document.CreateElement("Book", new XAttribute("id", 1));

Правильный ответ: Б

Компетенции (индикаторы): ОПК-4

6. Как выглядит пример кода для создания простого XML-документа с корневым элементом Library и одним подэлементом Book?

A) XmlDocument document = new XmlDocument(new XElement("Library", new XElement("Book")));

Б) XmlDocument document = new XmlDocument("Library", "Book");

В) XmlDocument document = new XmlDocument(new XElement("Library"), new XElement("Book"));

Правильный ответ: А

Компетенции (индикаторы): ОПК-4

7. Как добавить несколько атрибутов к элементу в LINQ to XML?

A) new XElement("Book", new XAttribute("id", 1), new XAttribute("title", "C# Basics"))

Б) new XElement("Book", {"id": 1, "title": "C# Basics"})

В) new XElement("Book").Add(new XAttribute("id", 1), new XAttribute("title", "C# Basics"))

Правильный ответ: А

Компетенции (индикаторы): ОПК-4

### **Задания закрытого типа на установление соответствия**

1. Установите правильное соответствие.

Определение	Значение
1) Класс, используемый для работы с текстовыми файлами в C#.	А) File.ReadAllLines()
2) Метод для чтения всех строк из текстового файла.	Б) FileNotFoundException
3) Метод для записи строки в текстовый файл, создавая новый файл, если его не существует.	В) StreamWriter
4) Исключение, которое может быть выброшено при попытке доступа к несуществующему файлу.	Г) File.CreateText()
5) Класс, предоставляющий методы	Д) FileStream

для асинхронной работы с файлами.

Правильный ответ:

1	2	3	4	5
В	А	Г	Б	Д

Компетенции (индикаторы): ОПК-4

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

Определение	Значение
1) <code>..</code> – выбор текущего узла;	А) выбор текущего узла;
2) <code>..</code> – выбор родительского узла;	Б) выбор родительского узла;
3) <code>*</code> – выбор всех дочерних узлов текущего узла;	В) выбор всех дочерних узлов текущего узла;
4) <code>@*</code> – выбор всех атрибутов текущего узла;	Г) выбор всех атрибутов текущего узла;

Правильный ответ:

1	2	3	4
А	Б	В	Г

Компетенции (индикаторы): ОПК-4

3. Установите правильное соответствие.

Определение	Значение
1) Класс, представляющий набор данных, полученных из базы данных	А) ExecuteReader()
2) Выполняет запрос и возвращает Reader для чтения данных	Б) ExecuteNonQuery
3) Метод используется для открытия соединения с базой данных	В) DataSet
4) Выполняет sql-выражение и возвращает количество измененных записей	Г) Open()

Правильный ответ:

1	2	3	4
В	А	Г	Б

Компетенции (индикаторы): ОПК-4

**Задания закрытого типа на установление правильной последовательности**

1. Установите правильную последовательность шагов для подключения к базе данных с использованием ADO.NET.

- А) Создать объект SqlConnection с строкой подключения.
- Б) Создать объект SqlCommand для выполнения команд SQL.
- В) Открыть соединение с базой данных, вызвав метод Open().
- Г) Закрыть соединение, вызвав метод Close().

Правильный ответ: А, В, Б, Г

Компетенции (индикаторы): ОПК-4

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): ОПК-4

3. Установите правильную последовательность шагов для чтения данных из базы данных с использованием ADO.NET.

- А) Вызвать метод ExecuteReader() у SqlCommand.
- Б) Создать объект SqlDataReader.
- В) Пройтись по данным с помощью метода Read().
- Г) Закрыть SqlDataReader.

Правильный ответ: Б, А, В, Г

Компетенции (индикаторы): ОПК-4

4. Установите правильную последовательность шагов для работы с транзакциями в ADO.NET.

- А) Начать транзакцию, вызвав метод BeginTransaction() у SqlConnection.
- Б) Зафиксировать транзакцию, вызвав метод Commit().
- В) Выполнить команды с помощью ExecuteNonQuery().
- Г) Создать объект SqlCommand и установить для него свойство Transaction.
- Д) Закрыть соединение.

Правильный ответ: А, Г, В, Б, Д,

Компетенции (индикаторы): ОПК-4

**Задания открытого типа**

### **Задания открытого типа на дополнение**

1. \_\_\_\_\_ – класс реализует поток, который сжимает и распаковывает данные с помощью алгоритма DEFLATE

Правильный ответ: DeflateStream

Компетенции (индикаторы): ОПК-4

2. \_\_\_\_\_ – класс позволяет сжимать и распаковывать данные в формате GZIP.

Правильный ответ: GZipStream

Компетенции (индикаторы): ОПК-4

3. \_\_\_\_\_ – это процесс преобразования объекта в поток байтов для сохранения или передачи в память, в базу данных или в файл.

Правильный ответ: Сериализация

Компетенции (индикаторы): ОПК-4

4. В ADO.NET классы: SqlConnection, SqlCommand, SqlDataAdapter используются для работы с \_\_\_\_\_?

Правильный ответ: SQL Server

Компетенции (индикаторы): ОПК-4

### **Задания открытого типа с кратким свободным ответом**

1. Какой метод класса File позволяет переместить файл из одной директории в другую?

Правильный ответ: File.Move()

Компетенции (индикаторы): ОПК-4

2. Какой метод класса FileInfo позволяет получить размер файла в байтах?

Правильный ответ: Length

Компетенции (индикаторы): ОПК-4

3. Какое пространство имен необходимо импортировать для работы с LINQ to XML в C#?

Правильный ответ: System.Xml.Linq

Компетенции (индикаторы): ОПК-4

4. Какой объект ADO.NET используется для выполнения SQL-запросов?

Правильный ответ: SqlCommand

Компетенции (индикаторы): ОПК-4

## Задания открытого типа с развернутым ответом

1. Создайте пример сериализации и десериализации в JSON-файл класса со свойствами Name и Age, с использованием пространства имен System.Text.Json..

Привести расширенное решение.

Время выполнения – 30 мин.

Ожидаемый результат:

Шаг 1: Подготовка класса для сериализации и десериализации

Сначала создадим класс, который будет представлять данные, которые будем записывать в JSON файл.

```
public class Person
{
    public string Name { get; set; }
    public int Age { get; set; }
}
```

Шаг 2: Запись объекта в JSON-файл

Создадим объект Person, затем сериализуем его в JSON и запишем в файл.

```
using System;
using System.IO;
using System.Text.Json;
class Program
{
    static void Main()
    {
        // Создаем новый объект Person
        var person = new Person
        {
            Name = "John Doe",
            Age = 30
        };
        // Задаем путь к файлу
        string filePath = "person.json";
        // Сериализуем объект в JSON и записываем в файл
        string jsonString = JsonSerializer.Serialize(person);
        File.WriteAllText(filePath, jsonString);
        Console.WriteLine("Данные записаны в файл: " + filePath);
    }
}
```

Шаг 3: Чтение объекта из JSON-файла

Прочитаем данные из указанного файла и десериализуем их обратно в объект Person.

```
using System;
```

```

using System.IO;
using System.Text.Json;
class Program
{
    static void Main()
    {
        // Задаем путь к файлу
        string filePath = "person.json";
        // Читаем содержимое файла
        string jsonString = File.ReadAllText(filePath);
        // Десериализуем JSON строку обратно в объект Person
        var person = JsonSerializer.Deserialize<Person>(jsonString);
        // Выводим данные на консоль
        Console.WriteLine("Имя: " + person.Name);
        Console.WriteLine("Возраст: " + person.Age);
    }
}

```

Пояснение.

Для работы с System.Text.Json необходимо подключить библиотеку System.Text.Json, которая входит в .NET Core и .NET 5/6/7.

Убедитесь, что файл, указанный в filePath, доступен для записи и чтения в вашем проекте.

1. Создание класса:

– Класс Person содержит два свойства: Name и Age. Эти свойства будут сериализованы и десериализованы в формате JSON.

2. Запись в файл:

– Создаем объект Person и заполняем его данными.

– Используем JsonSerializer.Serialize для преобразования объекта в строку JSON.

– С помощью File.WriteAllText записываем строку JSON в файл с указанным именем.

3. Чтение из файла:

– Используем File.ReadAllText для чтения содержимого файла.

– Десериализуем строку JSON обратно в объект Person с помощью JsonSerializer.Deserialize.

– Выводим данные на консоль.

Критерии оценивания:

– создание класса Person со свойствами Name и Age;

– создание объекта Person и заполнение его данными;

– использование JsonSerializer.Serialize для преобразования объекта в строку JSON;

– запись строки JSON в файл (File.WriteAllText);

– чтение содержимого файла (File.ReadAllText);

– десериализация строки JSON обратно в объект Person (JsonSerializer.Deserialize);

– вывод данных на консоль.

Компетенции (индикаторы): ОПК-4

2. Создать пример записи и чтения в/из XML файла информации о книгах (название, автор и год издания).

Привести расширенное решение.

Время выполнения – 30 мин.

Ожидаемый результат:

Запись XML файла.

using System;

using System.Xml;

class Program

{

static void Main()

{

// Создаем XmlDocument

XmlDocument xmlDoc = new XmlDocument();

// Создаем корневой элемент

XmlElement root = xmlDoc.CreateElement("Books");

xmlDoc.AppendChild(root);

// Создаем элементы книги

for (int i = 0; i < 3; i++)

{

XmlElement book = xmlDoc.CreateElement("Book");

XmlElement title = xmlDoc.CreateElement("Title");

title.InnerText = "Book Title " + (i + 1);

book.AppendChild(title);

XmlElement author = xmlDoc.CreateElement("Author");

author.InnerText = "Author " + (i + 1);

book.AppendChild(author);

XmlElement year = xmlDoc.CreateElement("Year");

year.InnerText = (2000 + i).ToString();

book.AppendChild(year);

// Добавляем книгу в корневой элемент

root.AppendChild(book);

}

// Сохраняем XML в файл

xmlDoc.Save("books.xml");

Console.WriteLine("XML файл успешно создан: books.xml");

}

}



Пояснение.:

Создание экземпляра XmlDocument – это основной класс для работы с XML в C#.

Создание корневого элемента – создаем корень и добавляем его в документ.

Создание элементов – в цикле добавляем несколько элементов <Book>, затем добавляем подэлементы <Title>, <Author> и <Year>.

Сохранение XML — использует метод Save, чтобы записать все в файл books.xml.

Чтения XML файла.

Теперь давайте прочитаем созданный XML файл и выведем информацию о книгах на консоль:

```
using System;
using System.Xml;
class Program
{
    static void Main()
    {
        // Загружаем XML файл
        XmlDocument xmlDoc = new XmlDocument();
        xmlDoc.Load("books.xml");
        // Получаем корневой элемент
        XmlElement root = xmlDoc.DocumentElement;
        // Перебираем все элементы Book
        foreach (XmlElement book in root.GetElementsByTagName("Book"))
        {
            string title = book["Title"].InnerText;
            string author = book["Author"].InnerText;
            string year = book["Year"].InnerText;
            // Выводим информацию о книге на консоль
            Console.WriteLine($"Title: {title}, Author: {author}, Year: {year}");
        }
    }
}
```

Пояснение чтения.

Загрузка XML файла – используем метод Load для чтения файла books.xml.

Получаем корневой элемент – так же, как и при записи, получаем корень документа.

Чтение элементов – используем метод GetElementsByTagName для получения всех элементов <Book> и перебираем их в цикле.

Вывод информации — для каждого элемента книги читаем значение подэлементов и выводим их на консоль.

Критерии оценивания:

- создание экземпляра класса XmlDocument;
- создание корневого элемента;
- создание элементов;
- сохранение XML;
- загрузка XML файла;
- получение корневого элемента;
- чтение элементов;
- вывод данных на консоль.

Компетенции (индикаторы): ОПК-4

### Экспертное заключение

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

Предлагаемые оценочные материалы адекватны целям и задачам реализации основной профессиональной образовательной программы по направлению подготовки 09.04.04 Программная инженерия.

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

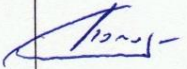
Разработанные и представленные для экспертизы оценочные материалы рекомендуются к использованию в процессе подготовки обучающихся по указанному направлению.

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