1. **Вопрос:**

Какую структуру можно передавать в качестве callback метода (функции)? (Передавать как аргумент, к примеру, методу Array.Sort)

**Ответ:** Метод, лямбда-выражение, анонимный метод.

**Ссылки на источники:**

1. **Вопрос:**

Выберите верные утверждения:

1) реализация callback'a возможна только с помощью делегата

2) callback это тип

3) callback это интерфейс

4) callback это функция

**Ответ:** 1, 4

**Ссылки на источники:**

1. **Вопрос:**

Что выведет предоставленный кусок кода?

Если возникнет ошибка компиляции, напишите: \*\*\*

Если ошибок и исключений нет, но на экран не выведется ничего, напишите: —-

Если возникнет ошибка исполнения или исключение, напишите: +++

static void PrintArray(int[] a)

{

Console.WriteLine();

for (int i = 0; i < a.Length; i++)

{

Console.Write(a[i].ToString("") + " ");

}

}

static void Main(string[] args)

{

int[] a = { 1, 3, -4, 7, 5 };

Array.ForEach(a, x => { x = 5 \* x;});

PrintArray(a);

}

**Ответ:** Сначала пустую строку,

затем перейдёт на следующую строку и выведет 1, 3, -4, 7, 5.

Чтобы элементы изменились надо использовать ConvertAll.

**Ссылки на источники:**

1) Метод ConvertAll: <https://docs.microsoft.com/ru-ru/dotnet/api/system.array.convertall?view=netframework-4.8>

2) Метод ForEach: <https://docs.microsoft.com/ru-ru/dotnet/api/system.array.foreach?view=netframework-4.8>

1. **Вопрос:**

namespace Test

{

public delegate int ConvertRule(int numb);

class Converter

{

public int Convert(int numb , ConvertRule Conv)

{ return Conv(numb); }

}

class Program

{

public static double Rule(int numb)

{

if (numb % 3 == 0) return numb / 3;

else return numb - (numb % 3);

}

static void Main(string[] args)

{

ConvertRule crMethod = Rule;

Converter testConverter = new Converter();

Console.WriteLine(testConverter.Convert(5, Rule));

}

}

}

Примечание:

Если возникнет ошибка компиляции, введите: \*\*\*

Если ошибок и исключений нет, но на экран не выведется ничего, введите: --- Если возникнет ошибка исполнения или исключение, введите: +++

**Ответ:** \*\*\*, т.к. ковариантность в данном случае не выполняется.

**Ссылки на источники:**

1. Ковариантность в делегатах: <https://docs.microsoft.com/ru-ru/dotnet/csharp/programming-guide/concepts/covariance-contravariance/using-variance-in-delegates>
2. **Вопрос:**

Что будет выведено на экран после выполнения данного кода? (\*\*\* - ошибка компиляции)

using System;

namespace test2

{

delegate int A(int x);

public class MainClass

{

public static int x(int x)

{ return (int)Math.Pow(x, 5); }

public static int y(int x, A a)

{ return a(x); }

public static void Main(string[] args)

{

int p = 2;

A a = new A(x);

Console.WriteLine(y(p,a));

}

}

}

**Ответ: \*\*\*,** т.к. модификатор доступа у делегата “меньше” модификатора доступа у метода.

**Ссылки на источники:**

1. Ограничения на использование уровней доступности: <https://docs.microsoft.com/ru-ru/dotnet/csharp/language-reference/keywords/restrictions-on-using-accessibility-levels>
2. **Вопрос:**

Отметьте верные утверждения:

**1.** Все функции DLL, вызываемые в вызове неуправляемого кода, требуют для своего выполнения наличия в управляемом коде функции обратного вызова.

**2.** Метод Array.Sort требует обязательного наличия функции обратного вызова среди переданных ему аргументов.

**3.** Метод Array.ConvertAll требует обязательного наличия функции обратного вызова среди переданных ему аргументов.

**4**. Вызов неуправляемого кода автоматически преобразует экземпляр делегата в знакомый формат обратного вызова.

**Ответ:** 3, 4

**Ссылки на источники:**

1. Реализация функций обратного вызова: <https://docs.microsoft.com/ru-ru/dotnet/framework/interop/how-to-implement-callback-functions>
2. Array.Sort: <https://docs.microsoft.com/ru-ru/dotnet/api/system.array.sort?view=netframework-4.8>
3. Array.ConvertAll: <https://docs.microsoft.com/ru-ru/dotnet/api/system.array.convertall?view=netframework-4.8>
4. Функции обратного вызова: <https://docs.microsoft.com/ru-ru/dotnet/framework/interop/callback-functions>
5. **Вопрос:**

Что выведет данный фрагмент кода?

public static void Main()

{

PointF[] apf = { new PointF(27.8F, 32.62F),

new PointF(99.3F, 147.273F),

new PointF(7.5F, 1412.2F) }; //system.drawings

Point[] ap = Array.ConvertAll(apf,

new Converter(PointFToPoint));

foreach( Point p in ap )

{

Console.WriteLine(p.X + " " + p.Y );

}

}

public static Point PointFToPoint(PointF pf)

{ return new Point(((int) pf.X), ((int) pf.Y));

}

Примечание: Если возникнет ошибка компиляции, введите: \*\*\* Если на экран не выведется ничего, введите: --- Если возникнет ошибка исполнения или исключение, введите: +++

**Ответ:** 27 32

99 147

7 1412

**Ссылки на источники:**

1. Point : <https://docs.microsoft.com/ru-ru/dotnet/api/system.drawing.point?view=netframework-4.8>
2. PointF : <https://docs.microsoft.com/ru-ru/dotnet/api/system.drawing.pointf?view=netframework-4.8>
3. **Вопрос:**

Вызов обработчиков события является частью самого события?

**1** да

**2** нет

**Ответ:** 2

**Ссылки на источники:**

1. Делегаты и события в .NET: <https://habr.com/ru/post/198694/>
2. **Вопрос:**

Что обычно возвращают функции обратного вызова сообщая о сбое?

1)1

2)-1

3)0

4)false

5)null

**Ответ:** 3

**Ссылки на источники:**

1. Реализация функций обратного вызова: <https://docs.microsoft.com/ru-ru/dotnet/framework/interop/how-to-implement-callback-functions>
2. **Вопрос:**

Выберите верные утверждения для Callback-a:

**1)** Обратный вызов позволяет в функции исполнять код, который задаётся в аргументах при её вызове.

**2)** Этот код может быть определён в других контекстах программного кода и быть недоступным для прямого вызова из этой функции.

**3)** Этот НЕ код может быть определён в других контекстах программного кода и быть недоступным для прямого вызова из этой функции.

**Ответ:** 1, 2

**Ссылки на источники:**

1. Callback: <https://www.sbup.com/wiki/Callback_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)>
2. Callback: <https://www.sbup.com/wiki/Callback_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)>