**Делегаты**

|  |  |
| --- | --- |
|  | Вопрос 1 |
|  | Выберите статические методы делегата :  1) Target 2) Method 3) Combine 4) EndInvoke 5) Remove |
|  | 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: +++  delegate void Meow(ref int a, out int b, params int[] q); class Program { static void Main(string[] args) { int a = 0, b = 0; int[] q = new int[5]; Meow meow = new Meow ((ref int aa, out int bb, int[] qq)  { bb = -7; a--; }); Meow anotherMeow = new Meow (delegate(ref int aa, out int bb, int[] qq) { bb = 2; a+=5; for (int i=0; i<qq.Length; i++) { qq[i] = i + i - bb \* i-1; } }); meow(ref a, out b, q); anotherMeow(ref a, out b, q); Console.WriteLine(a.ToString() + ";" + b.ToString() + ";" + q[2].ToString()); } } |
|  | Как объявлять делегат? а) delegate void(); b)delegate void MyDelegate(); c)static delegate int MyDelegate(ref int x, ref int y); d) protected delegate double MyDelegate(out int x, out double z); |
|  |  |
|  | Что будет выведено после выполнения данной программы (\*\*\* - ошибка компиляции)? using System; namespace testdelegate { delegate int A(); class MainClass { public static void Main(string[] args) { A [] a = new A[5]; for (int i = 0; i < 5; i++) { a[i] = () => { return i; }; } for (int i = 0; i < 5; i++) { Console.Write(a[i]()); }  } } } |
|  | В результате выполнения следующей программы: class Program { delegate int Del(params int[] p); static void Main(string[] args) { Del del1 = delegate (int[] p) { int k = 0; for (int i = p.Length - 1; i >= 0; i--) { k += p[i]; } return k; }; Console.WriteLine(del1(1, 2, 5)); } } на экран будет выведено:  Примечание: Если возникнет ошибка компиляции, введите: \*\*\* Если на экран не выведется ничего, введите: --- Если возникнет ошибка исполнения или исключение, введите: +++ |
|  |  |
|  | В результате исполнения данной команды на экран будет выведено: delegate int Del(ref int a);  class Program { static void Main(string[] args) { Del del1 = delegate (ref int a) { return a \* 2; }; int b = 1; del1(ref b); Console.WriteLine(b); } } |
|  | Если один из методов вызывает неперехваченное исключение, то следующие методы в списке вызова делегата: 1. Исполняются 2. Не исполняются |
|  | Что будет выведено в результате выполнения следующего фрагмента кода: namespace Test1 { delegate void Test(); class Program { static void Main(string[] args) { int x = 2; Test d = () => { for (int i = 1; i <= 3; i++) { x\*= 2+i; x--; } } d(); Console.WriteLine(x); Console.ReadKey(); } } }  Если возникнет ошибка компиляции, напишите: \*\*\* Если ошибок и исключений нет, но на экран не выведется ничего, напишите: ---  Если возникнет ошибка исполнения или исключение, напишите: +++ |
|  | Если вызвать .Target от элемента массива делегатов, который ссылается на статический метод, то какое значение вернет программа? |
|  | Что выведет программа? delegate int Del(ref int x);  static int Func(ref int a) { a \*= 2; return a\*2; }  static void Main(string[] args) { Del d = delegate (ref int x) { return x - 2; }; d += (ref int x)=> { x += -2; return x; }; d += Func;  int a = 10;  Console.WriteLine(d(ref a)); } |
|  | Делегат-тип можно объявить внутри 1) класса 2) пространства имен 3) метода 4) перечисления 5) структуры 6) глобального пространства имен |
|  | Укажите правильный синтаксис объявления массива делегатов этого делегата типа: public delegate int Del(int a); 1) public delegate int[] MyDelegate (int a); 2) public delegate [] Del = new delegate [](int b); 3) delegate int [] Del = new delegate int [] (int c); 4) Del[] methods = new DelFigure1[] (); 5) Del[] methods = new Del[] {b => b+1}; |
|  | Верно, что контравариантность в языке С# позволяет |
|  | Что возвращает метод GetlnvokationList()? |
|  | Могут ли параметры делегата идти с модификатором ref или out? |
|  | Если возникнет ошибка компиляции, напишите: \*\*\* Если ошибок и исключений нет, но на экран не выведется ничего, напишите: ---  Если возникнет ошибка исполнения или исключение, напишите: +++ Что выведет программа?  namespace ConsoleApp1 { delegate int Del1(ref int x); class Program { static void Main(string[] args) { int x = 10; Del1 d= (ref a) => 6 \* a \* a; Console.WriteLine(d(ref x)); }  } } |
|  | Требуется ли точное соответствие методов типу делегата? |
|  | Лямбда :  Class Length{ public int x;}  delegate int DEL(Point i);  static void Main(string[] args)  {  DEL del1= x=> x \* x;  Console.WriteLine(del1(10));  //Результат работа? } |
|  | Каким образом объявляется делегат(общая форма)? На какой метод может ссылаться экземпляр делегата? |
|  | Написание какого из следующих методов можно опустить и он будет вызван автоматически? 1. RemoveAll() 2. BeginInvoke() 3. EndInvoke() 4. Invoke() 5. GetlnvokationList() |
|  | Верно, что delegate void Del (int x); отвечает за: 1) Создание объекта делегата типа Del 2)вызов анонимного метода, связанного с делегатом MyDel 3)описание типа делегата с именем Del |
|  | Для добавления метода в список вызова делегата допустимо использовать следущий код: 1. multiDel = hiDel + byeDel; 2. multiDel = hiDel \* byeDel; 3. multiDel = System.Delegate.Combine(hiDel, byeDel); 4. multiDel = hiDel & byeDel; |
|  |  |
|  | Выберите верные утверждения.  Что такое делегат? 1) тип, который представляет ссылки на методы с определенным списком параметров и типом возвращаемого значения 2) абстрактный класс 3) аналог указателя на функцию 4) статический класс, инкапсулирующий ссылку на метод 5) тип |
|  | Какие методы и свойства НЕ входят в класс MulticastDelegate? 1)Method() 2)GetlnvokatingList() 3)Method 4)Delete() 5)RemoveAll() |
|  | Напишите \*\*\* - если программа не скомпилируется +++ - если ничего не будет на экране --- - если вылетит эксепшен Что выведет данный фрагмент программы: delegate int MyDelegate(int x); MyDelegate myDel = (a)=> { Console.WriteLine(a); return a\*a\*a; } myDel(15); |
|  | Можно ли делегат передавать как параметр в метод? |
|  | Что будет выведено после выполнения данной программы (\*\*\* - ошибка компиляции)? using System; namespace testdelegate { delegate int A(); class MainClass { public static void Main(string[] args) { A a ; for (int i = 0; i < 5; i++) { a = () => { return i; }; } Console.WriteLine(a());  } } } |
|  | Выберите правильные утверждения о делегатах: 1) Делегаты можно передавать в качестве параметра метода 2) В сигнатуру делегатов возвращаемое значение никогда не входит. 3) Метод можно вызвать (активировать) с помощью экземпляра делегата. 4) Методы, не возвращающие значения, не могут быть вызваны делегатом. 5) Методы, на которые ссылаются делегаты, должны иметь те же параметры и тот же тип возвращаемого значения. |
|  | Три делегата: delegate uint MyDel1(int a) delegate int MyDel2(int a) delegate uint MyDel(uint a)  в каких строчках есть ошибка компиляции? 1) MyDel1 del = i => i + 1; 2) MyDel2 del = i => i + 1; 3) MyDel3 del = i => i + 1; |
|  | Отметьте верные утверждения: 1. Метод, который передается как параметр делегата, должен иметь такую же сигнатуру, что и объявление делегата; 2. Экземпляр делегата может инкапсулировать только метод экземпляра. 3. С помощью оператора "-" можно удалить делегат, входящий в состав многоадресного делегата. 4. Допускается использование goto, break или continue внутри блока анонимного метода, если цель перехода располагается за пределами блока. 5. Анонимные методы нельзя использовать в левой части оператора is. |
|  | Допустимые варианты удаления методов из списка вызова делегата: 1. allMethodsDelegate -= d1; 2. allMethodsDelegate/d1; 3. Del oneMethodDelegate = allMethodsDelegate - d1; 4. Del oneMethodDelegate = allMethodsDelegate/d1; |
|  |  |
|  |  |
|  | В каких лямбда-выражениях допущены ошибки компиляции? delegate int Del(int x);  1) Del del1 = x => { return x; }; 2) Del del2 = int x => { return x; }; 3) Del del3 = x = 0 => { return x; }; 4) Del del4 = (x) => { return x; }; 5) Del del5 = (int x = 0) { return x; }; |
|  | Что будет выведено на экран? namespace ConsoleApplication1 { delegate void Del1(int x);  class A { public static void Main() { Del1 del1 = i => { Console.Write(i + 3); }; del1 += del1; del1.Invoke(2); del1 = (Del1)Delegate.Combine(del1, del1); del1(1); del1 = (Del1)Delegate.Remove(new Del1((int x) => { Console.WriteLine(4); }),del1 ); del1(0); } } } |
|  | Какие операторы используются для обращения к методам Combine и Remove в сокращенной нотации (перегружены для многоадресных делегатов) 1) \*= 2) ! 3) + 4) += 5) - 6) -= 7) && 8) ()  9) != 10) % |
|  |  |
|  | Что делает метод Remove()? |
|  | входит ли в сигнатуру делегата возвращаемое значение? |
|  | Если возникнет ошибка компиляции, напишите: \*\*\* Если ошибок и исключений нет, но на экран не выведется ничего, напишите: ---  Если возникнет ошибка исполнения или исключение, напишите: +++ Что выведет программа?  namespace ConsoleApp1 { delegate string Sum(int number);  class Program { static Sum SomeVar() { string s = "";   Sum del = delegate (int number) { for (int i = 0; i <= number; i++) s += i.ToString(); return s; }; return del; }  static void Main() { Sum del1 = SomeVar();  for (int i = 1; i <= 2; i++) { Console.Write(del1(i)); }  Console.ReadLine(); } } } |
|  | Могут ли модификаторы доступа быть применены к делегату? |
|  | Как много типов делегата в C#? |
|  | Для чего используется метод Invoke() ? |
|  | Укажите верные утверждения Делегат... 1. может служить для вызова любого метода с соответствующей сигнатурой и возвращаемым типом 2. позволяет вызывать метод, на который он ссылается 3. автоматически снабжается способностью вызывать свои методы синхронно или асинхронно 4. используется для передачи метода в качестве аргумента к другим методам |
|  | В результате выполнения следующей программы: delegate double MyDel(int par); class Program { static void Main() { int d = 20; MyDel a = x => X \* 20; Console.WriteLine(a(d)); } } на экран будет выведено: 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: +++ |
|  | Примечание: Если возникнет ошибка компиляции, выведите: \*\*\* Если ошибок и исключений нет, но на экран не выведется ничего, введите: --- Если возникнет ошибка исполнения или исключение, введите: +++  В результате выполнения следующей программы: using System; class Program { delegate void Del(double[] ar); static void Main() { double[] dar = { 5.6, 44.9 }; Del print = delegate (double[] ar) { foreach (double d in ar) Console.Write((int)d / 10); }; print(dar); } } на экран будет выведено: |