Laboratorijas darbs par Delegātu izmantošanu C# programmēšanas valodā

2025

Saturs

[1. Uzdevums “Delegāti” 2](#_Toc191289911)

[1.1 Programmas kods 3](#_Toc191289912)

[1.2 Programmas izpilde 4](#_Toc191289913)

[1.2.1 Delegate – Saskaitit() 4](#_Toc191289914)

[1.2.2 Delegate – Atnemt() 4](#_Toc191289915)

[1.2.3 Delegate – Dalit() 5](#_Toc191289916)

[1.2.4 Delegate – Reizinat() 5](#_Toc191289917)

[2. Uzdevums - “Anonīmās funkcijas” 6](#_Toc191289918)

[2.1 Programmas izpilde 6](#_Toc191289919)

[2.2 Programmas kods 7](#_Toc191289920)

[3.Uzdevums “Anonīmās funkcijas” 8](#_Toc191289921)

[3.1 Programmas kods 9](#_Toc191289922)

[3.1 Programmas izpilde 11](#_Toc191289923)

[Laboratorijas darba aizstavešanas jautājumi un atbildes 12](#_Toc191289924)

# Uzdevums “Delegāti”

Uzdevuma kritēriji, kuri tika izpildīti laboratorijas darba izstrādes brīdī:

1. Izveidot jaunu konsoles aplikācijas projektu.
2. Definēt jaunu delegāta tipu ar nosaukumu Matematika, kurš strādā ar funkciju, kas saņem divus double tipa mainīgos un atgriež double tipa rezultātu
3. Klasē Program realizēt:
   1. Funkciju Saskaitit(), kas saņem divus double tipa mainīgos un atgriež to sumu (double tipa)
   2. Funkciju Atnemt(), kas saņem divus double tipa mainīgos un atgriež to starpibu (double tipa)
   3. Funkciju Reizinat(), kas saņem divus double tipa mainīgos un atgriež to reizinājumu (double tipa)
   4. Funkciju Dalit(), kas saņem divus double tipa mainīgos un atgriež to dalījumu (double tipa)
4. Klases Program funkcijā Main:
   1. Definēt delegāta Matemātika tipa mainīgo ar nosaukumu del un inicializēt to ar vērtību null.
   2. Izvadīt paziņojumu.

Izvelies darbību 1- saskaitīt, 2 - atņemt, 3 - dalīt, 4 –reizināt

* 1. Nolasīt lietotāja ievadīto vērtību
  2. Atkarībā no lietotāja ievadītās vērtības piešķirt funkciju delegātam (ja 1 tad Saskaitit() , ja 3 tad Dalit() utt.)
  3. Pieprasīt, lai lietojās ievada divus skaitļus.
  4. Lietojot ievadītos skaitļus izpildīt delegātu.
  5. Izvadīt uz ekrāna rezultātu;

## Programmas kods

|  |
| --- |
| 1. using System; 2. //Definējam delegātu ar nosaukumu "Matematika", kas saņem 2 doublu tipa mainīgos kā parametrus 3. //{modifier} delegate [return\_type] [delegate\_name] ([parameter\_list]) 4. public delegate double Matematika(double sk1, double sk2); 5. class Program 6. { 7. //Funkcija atgriež summu no 2 skaitļiem 8. static double Saskaitit(double x, double y) => x + y; 9. //Funkcija atgriež starpibu no 2 skaitļiem 10. static double Atnemt(double x, double y) => x - y; 11. //Funkcija atgriež reizinājumu no 2 skaitļiem 12. static double Reizinat(double x, double y) => x \* y; 13. //Funkcija atgriež dalijumu no 2 skaitļiem 14. static double Dalit(double x, double y) => x / y; 15. static void Main() 16. { 17. //Definē instanci delegāta "Matematika", ar nosaukumu "del" un vertibu null 18. Matematika del = null; 19. //Pieprasam, lai lietotājs veic izvēli darbībai 20. Console.WriteLine("Izvelies darbibu 1 - saskaitit, 2 - atnemt, 3 - dalit, 4 - reizinat"); 21. var izvele = int.Parse(Console.ReadLine()); 22. //Pec izveles izveles pieškiram delegatam funkciju 23. if(izvele == 1) 24. { 25. del = Saskaitit; 26. } 27. else if(izvele == 2) 28. { 29. del = Atnemt; 30. } 31. else if (izvele == 3) 32. { 33. del = Dalit; 34. } 35. else if (izvele == 4) 36. { 37. del = Reizinat; 38. } 39. //Pieprasam lai lietotājs ievada 2 skaitļus, kuri tiks padoti delegātam 40. Console.WriteLine("Ievadi skaitli 1: "); 41. var sk1 = double.Parse(Console.ReadLine()); 42. Console.WriteLine("Ievadi skaitli 2: "); 43. var sk2 = double.Parse(Console.ReadLine()); 44. //ar skaitļiem izpildam delegātu 45. double result = del(sk1, sk2); 46. Console.WriteLine("Delegats ir izpildits, seit ir rezultats: {0}",result); 47. } 48. } |

## 1.2 Programmas izpilde

### 1.2.1 Delegate – Saskaitit()

|  |
| --- |
| Izvelies darbibu 1 - saskaitit, 2 - atnemt, 3 - dalit, 4 - reizinat  1  Ievadi skaitli 1:  3  Ievadi skaitli 2:  5  Delegats ir izpildits, seit ir rezultats: 8  C:\Users\adria\Desktop\Program\_tehnologijas\1.Laboratorijas darbs\1.Uzdevums\_Delegati\App1\bin\Debug\App1.exe (process 41964) exited with code 0 (0x0).  To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops.  Press any key to close this window . . . |

Programma pieprasa izvēlēties operāciju un ievadīt 2 skaitļus, kas tiek piešķirti mainīgajiem, kuri tiek padoti delegāta instancei. Pēc izvēles programma piešķir delegātam funkciju Saskaitit().

### 1.2.2 Delegate – Atnemt()

|  |
| --- |
| Izvelies darbibu 1 - saskaitit, 2 - atnemt, 3 - dalit, 4 - reizinat  2  Ievadi skaitli 1:  6  Ievadi skaitli 2:  2  Delegats ir izpildits, seit ir rezultats: 4  C:\Users\adria\Desktop\Program\_tehnologijas\1.Laboratorijas darbs\1.Uzdevums\_Delegati\App1\bin\Debug\App1.exe (process 47304) exited with code 0 (0x0).  To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops.  Press any key to close this window . . . |

Programma pieprasa izvēlēties operāciju un ievadīt 2 skaitļus, kas tiek piešķirti mainīgajiem, kuri tiek padoti delegāta instancei. Pēc izvēles programma piešķir delegātam funkciju Atnemt().

### 1.2.3 Delegate – Dalit()

|  |
| --- |
| Izvelies darbibu 1 - saskaitit, 2 - atnemt, 3 - dalit, 4 - reizinat  3  Ievadi skaitli 1:  9  Ievadi skaitli 2:  3  Delegats ir izpildits, seit ir rezultats: 3  C:\Users\adria\Desktop\Program\_tehnologijas\1.Laboratorijas darbs\1.Uzdevums\_Delegati\App1\bin\Debug\App1.exe (process 26668) exited with code 0 (0x0).  To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops.  Press any key to close this window . . . |

Programma pieprasa izvēlēties operāciju un ievadīt 2 skaitļus, kas tiek piešķirti mainīgajiem, kuri tiek padoti delegāta instancei. Pēc izvēles programma piešķir delegātam funkciju Atnemt().

### 1.2.4 Delegate – Reizinat()

|  |
| --- |
| Izvelies darbibu 1 - saskaitit, 2 - atnemt, 3 - dalit, 4 - reizinat  4  Ievadi skaitli 1:  9  Ievadi skaitli 2:  5  Delegats ir izpildits, seit ir rezultats: 45  C:\Users\adria\Desktop\Program\_tehnologijas\1.Laboratorijas darbs\1.Uzdevums\_Delegati\App1\bin\Debug\App1.exe (process 23816) exited with code 0 (0x0).  To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops.  Press any key to close this window . . . |

Programma pieprasa izvēlēties operāciju un ievadīt 2 skaitļus, kas tiek piešķirti mainīgajiem, kuri tiek padoti delegāta instancei. Pēc izvēles programma piešķir delegātam funkciju Reizinat().

# 2. Uzdevums - “Anonīmās funkcijas”

Uzdevuma kritēriji, kuri tika izpildīti laboratorijas darba izstrādes brīdī:

Pārveidot uzdevumu „Delegāti”, tā lai funkcijas Saskaitit(), Atņemt(), Reizinat(), Dalit(), būtu realizētas tikai delegāta definēšanas brīdi (anonīmi).

## 2.1 Programmas izpilde

Programmas izpilde ir tāda pati kā 1.2 (skat. 1.2 nodaļu).

## 2.2 Programmas kods

|  |
| --- |
| using System;  //Definējam delegātu ar nosaukumu "Matematika", kas saņem 2 doublu tipa mainīgos kā parametrus  //{modifier} delegate [return\_type] [delegate\_name] ([parameter\_list])  public delegate double Matematika(double sk1, double sk2);  class Program  {  public static void LietotajaOperacijas(Matematika del)  {  //Pieprasam lai lietotājs ievada 2 skaitļus, kuri tiks padoti delegātam  Console.WriteLine("Ievadi skaitli 1: ");  var sk1 = double.Parse(Console.ReadLine());  Console.WriteLine("Ievadi skaitli 2: ");  var sk2 = double.Parse(Console.ReadLine());  //ar skaitļiem izpildam delegātu  double result = del(sk1, sk2);  //Izvadam rezultātu  Console.WriteLine("Delegats ir izpildits, seit ir rezultats: {0}", result);  }  static void Main()  {  //Pieprasam, lai lietotājs veic izvēli darbībai  Console.WriteLine("Izvelies darbibu 1 - saskaitit, 2 - atnemt, 3 - dalit, 4 - reizinat");  var izvele = int.Parse(Console.ReadLine());  //Pec izveles pieškiram delegatam funkciju  if (izvele == 1)  {  Matematika del = delegate (double a, double b)  { return a + b; };  LietotajaOperacijas(del);  }  else if (izvele == 2)  {  Matematika del = delegate (double a, double b)  { return a - b; };  LietotajaOperacijas(del);  }  else if (izvele == 3)  {  Matematika del = delegate (double a, double b)  { return a / b; };  LietotajaOperacijas(del);  }  else if (izvele == 4)  {  Matematika del = delegate (double a, double b)  { return a \* b; };  LietotajaOperacijas(del);  }  }  } |

# 3.Uzdevums “Anonīmās funkcijas”

Uzdevuma kritēriji, kuri tika izpildīti laboratorijas darba izstrādes brīdī:

1. Izveidot jaunu konsoles aplikācijas projektu
2. Klases Program funkcijā Main:
   1. Izveidot veselu skaitļu sarakstu un aizpildīt to ar 10 gadījuma skaitļiem.
   2. Izvadīt saraksta vērtības uz ekrāna
   3. Definēt anonīmu funkciju izmantojot predikāta Predicate<T> tipa delegātu, kas atgriež true, ja saņemtā mainīgā vērtība ir pirmskaitlis (dalās tikai ar 1 un sevi pašu) un false citos gadījumos.
   4. Lietojot iepriekš izveidoto anonīmo funkciju testēt saraksta funkcijas, FindAll(), Find(), Remove(), RemoveAll();
   5. Rezultātu izvadīt uz ekrāna

## 3.1 Programmas kods

|  |
| --- |
| using System;  using System.Collections.Generic;  class Program  {  static void PrintList(List<int> numbers)  {  foreach (int number in numbers)  {  Console.Write(number + " ");  }  Console.WriteLine("\n");  }  static bool IsPrime(int number)  {  if (number < 2) return false;  for (int i = 2; i \* i <= number; i++)  {  if (number % i == 0) return false;  }  return true;  }  static void FindAllPrimes(List<int> numbers, Predicate<int> isPrime)  {  List<int> primeNumbers = numbers.FindAll(isPrime);  Console.WriteLine("Pirmskaitli saraksta:");  PrintList(primeNumbers);  }  static void FindFirstPrime(List<int> numbers, Predicate<int> isPrime)  {  int firstPrime = numbers.Find(isPrime);  if (firstPrime != 0)  {  Console.WriteLine("Pirmais atrastais pirmskaitlis: " + firstPrime);  }  else  {  Console.WriteLine("Neviens pirmskaitlis netika atrasts.");  }  }  static void RemoveFirstPrime(List<int> numbers, Predicate<int> isPrime)  {  int firstPrime = numbers.Find(isPrime);  if (firstPrime != 0 && numbers.Remove(firstPrime))  {  Console.WriteLine($"Nonemam pirmo pirmskaitli: {firstPrime}");  }  else  {  Console.WriteLine("Nav pirmskaitlu, ko nonemt.");  }  }  static void RemoveAllPrimes(List<int> numbers, Predicate<int> isPrime)  {  int removedCount = numbers.RemoveAll(isPrime);  Console.WriteLine($"Nonemam {removedCount + 1} pirmskaitlus no saraksta.");  }  static void Main()  {  List<int> randomNumbers = new List<int>();  Random random = new Random();  for (int i = 0; i < 10; i++)  {  int randomNumber = random.Next(1,101);  randomNumbers.Add(randomNumber);  }    Console.WriteLine("Generetie gadijuma skaitli:");  foreach (int number in randomNumbers)  {  Console.WriteLine(number);  }  Console.WriteLine("\n");  Predicate<int> isPrime = IsPrime;  FindAllPrimes(randomNumbers, isPrime);  FindFirstPrime(randomNumbers, isPrime);  RemoveFirstPrime(randomNumbers, isPrime);  RemoveAllPrimes(randomNumbers, isPrime);  Console.WriteLine("\nSaraksts pec darbibam:");  PrintList(randomNumbers);  }  } |

## 3.1 Programmas izpilde

|  |
| --- |
| Generetie gadijuma skaitli:  83  78  85  65  21  62  74  58  19  93  Pirmskaitli saraksta:  83 19  Pirmais atrastais pirmskaitlis: 83  Nonemam pirmo pirmskaitli: 83  Nonemam 2 pirmskaitlus no saraksta.  Saraksts pec darbibam:  78 85 65 21 62 74 58 93  C:\Users\adria\Desktop\Program\_tehnologijas\1.Laboratorijas darbs\3.Uzdevums\_Anonimas\_funkcijas\ConsoleApp1\ConsoleApp1\bin\Debug\ConsoleApp1.exe (process 18644) exited with code 0 (0x0).  To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops.  Press any key to close this window . . . |

Programma ģenerē sarakstu ar gadījuma skaitļiem, ietvarā (1 - 100), tad izmantojot kodā redzamās funkcijas:

* + - 1. Atrodam visus pirmskaitļus sarakstā – **FindAll()**;
      2. Atrodam pirmo pirmskaitli sarakstā – **Find()**;
      3. Noņemam pirmo pirmskaitli sarakstā – **Remove()**;
      4. Noņemam visus pirmskaitļus sarakstā – **RemoveAll()**;
      5. Izvadām atjaunoto sarakstu – **PrintList()**;

# Git repository izveidei un projekta pārsūtīšanai uz GitHub kontu

1. No termināli, dodamies uz Home folderi:

*cd "Desktop\Program\_tehnologijas\Laboratorijas darbs\_Delegāti"*

1. Inicializējam tukšu Git repository:

*git init*

1. Padodam mūsu git repositry visus failus no Home folder:

git commit -m “Initial commit”

1. Dodamies uz galveno atzarojumu:

git branch -M main

1. Pievienojam GitHub izveidoto repository:

git remote add origin <https://github.com/AdriansFreimanis/Labratory_work_Delegates.git>

1. Saglabājam veiktās izmaiņas:

git push -u origin main

* Lai veiktu izmaiņas un apstiprinātu tās:

*git add .*

git commit -m “Changes in code documentation”

git branch -M main

git push -u origin main

* Laba prakse ir pievienot arī *.gitignore* failu, to var izdarīt sekojoši:

*echo. > .gitignore*

* + Tad failā iekopējam šo tekstu, kas paradzēts C# kodam:

|  |
| --- |
| # Ignore Visual Studio & build files  .vs/  bin/  obj/  \*.user  \*.suo  \*.sln.docstates  \*.log  # Ignore user secrets  appsettings.json  appsettings.Development.json  # Ignore IDE configurations  .idea/  \*.iml  # Ignore other unnecessary files  \*.tmp  \*.cache |

* + Saglabājam izmaiņas un aizveram failu

# Laboratorijas darba aizstavešanas jautājumi un atbildes

1. **Kas ir Delegāts, programmēšanas valodā C#?**

Delegāts ir pārstāvis, kas rada atsauci uz kādu funkciju. Veidojot Delegāta instanci to var saistīt ar kādu funkciju, kas kaut ko atgriež. Funkciju var izsaukt, izmantojot, Delegāta instanci.

***Piemērs –*** “Latvijas Republikas vēstnesis Eiropas Parlamentam, dodas nodot jaunākos ziņojumus par Latvijas stāvokli un nostāju, par Krievijas un Ukrainas karu, pēc Krievu un ASV sarunām. Šajā gadījumā vēstnesis ir kā delegāts **Main()** funkcijā, mūsu kodā, kas nodod ziņojumu no Latvijas Parlamenta – **Funkcija()”**

1. **Ko var izdarīt ar Delegātu un nevar izdarīt ar funkciju?**

Ar Delegātu ir iespējams padot funkcijas kā parametrus, Delegāti var saņemt un uzkrāt vairākas funkcijas, programmas izpildes brīdī ir iespējams piešķirt vai mainīt delegātam pieejamās funkcijas.

1. **Ar, ko anonīmā funkcija atšķiras no parastās funkcijas?**

Anonīmām funkcijām netiek piešķirts nosaukums, kamēr parastām tiek, anonīmas funkcijas izmanto “delegate” vai “lambda” izteiksmes. Parastās funkcijas nevar pa tiešo tikt izmantotas ar “Events” un “Delegates”, kamēr anonīmās funkcijas var. Anonīmās funkcijas lielākoties izmanto īsās koda daļās, kur nav nepieciešams atkārtot operāciju.

1. **Kādos gadījumos un ar kādu mērķi tiek lietotas anonīmās funkcijas?**

Anonīmās funkcijas tiek lietotas, kadi ir nepieciešams tikai vienreiz izsaukt šo funkciju vai arī , ja ir nepieciešamība padot funkciju kā parametru. Lielākotiess starp programmētājiem anonīmās funkcijas tiek lietotas ar mērķi neradīt nevajadzīgu kodu, kā piemēram nedefinēt veselu funkciju, lai tikai to vienreiz izsauktu.