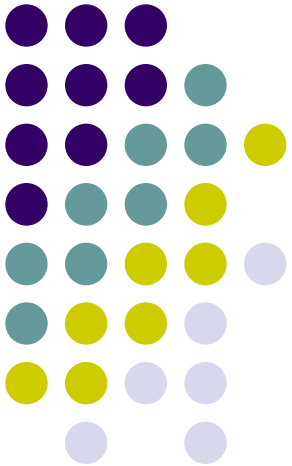
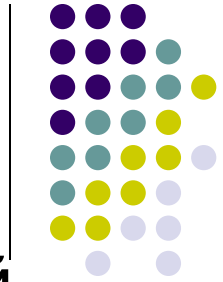
Архітектура .NET
Основи C#

Common Language Runtime
Intermediate Language.
Assembly
Класи .NET Framework
.NET Application by C#
Змінні. Типи. Керування. Масиви.
Простори імен

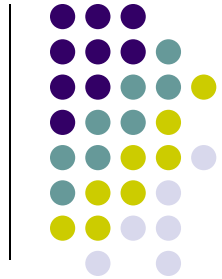


Значення .NET і C#



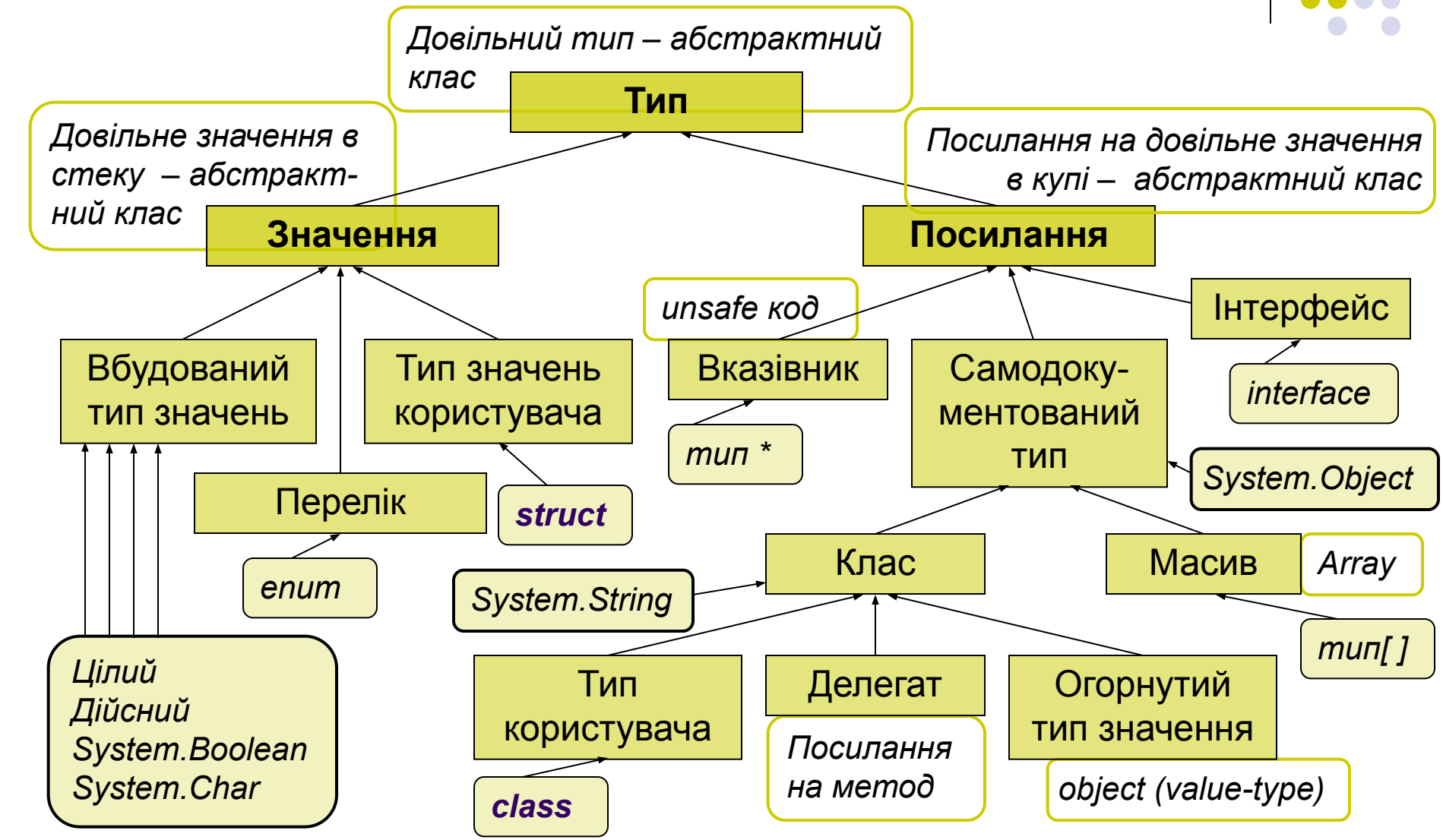
- Новий каркас (середовище), новий системний інтерфейс для програмування
  - Об’єктно-орієнтований підхід
  - Зрозумілий дизайн базових класів
  - Незалежність від мови програмування, досконалі засоби взаємодії
  - Підтримка динамічних web-сторінок, Web-служб
  - Ефективний доступ до даних
  - Assembly VS DLL
  - Підвищена безпека, спрощена інсталяція
- Нова мова програмування для середовища .NET
  - ООП, звичайне наслідування, інтерфейси
  - Безпека типів, автоматичне збирання сміття
  - Підтримка всіх можливостей середовища та сучасних технологій

Спільне середовище виконання мов CLR. Проміжна мова IL

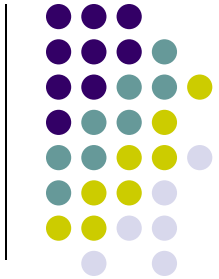


- Двокрокова компіляція:
  - Звичайний компілятор “C# → IL(байт-код)”;
  - JIT-компілятор “IL → машинна мова”;
- Підвищення продуктивності: зменшення обсягу компіляції, краща оптимізація
- Взаємодія мов: наслідування класів, включення і передавання об’єктів, надсилання повідомлень, налагодження
- Тип значення зберігає дані (стек), посилання зберігає адресу (керована купа)
- Common Type System

Система типів IL

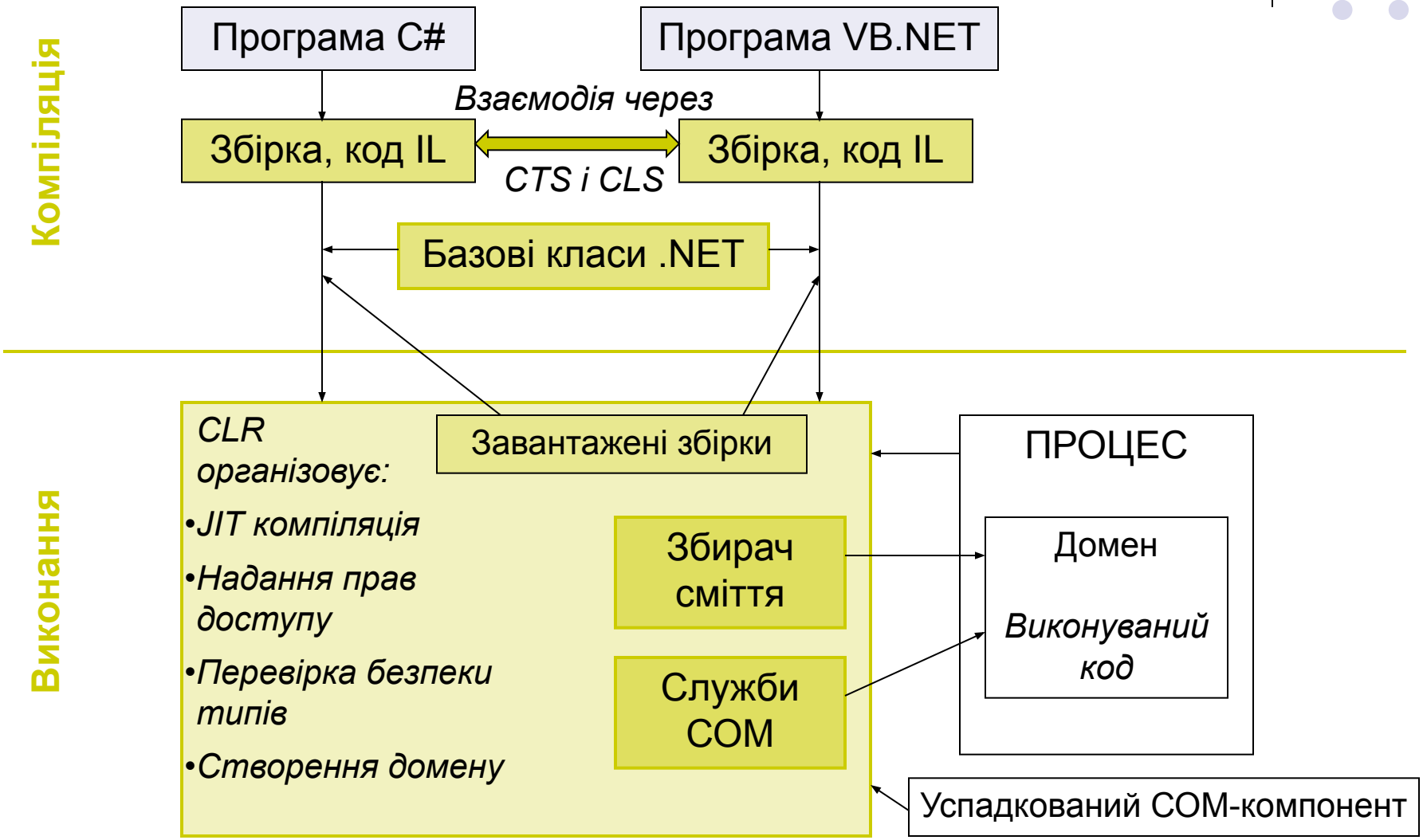


Особливості внутрішньої мови

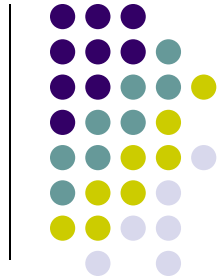


- Common Language Specification:
  - мінімальні вимоги до .NET-орієнтованих компіляторів
  - вимоги сумісності до відкритих методів і класів
- “Збирання сміття” – недетерміноване виявлення і знищення об’єктів, на які нема посилань
- Домени аплікацій – спосіб безпечного виконання декількох компонент в межах одного процесу
- Опрацювання помилок за допомогою винятків
- Атрибути типів і методів – корисні метадані
- Збірка – логічна одиниця компільованого коду:
  - бібліотека або аплікація (має точку входу);
  - приватна або загального використання (строге ім’я);
  - містить маніфест і метадані, доступні для рефлексії.

Засоби .NET компіляції та виконання

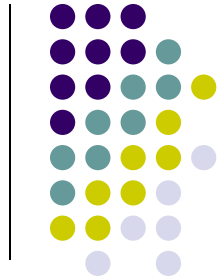


Створення аплікацій .NET



- Класи .NET Framework
  - Підтримка системи типів
  - Підтримка графічного інтерфейсу Windows
  - Підтримка Web Forms (ASP.NET), взаємодія з мережею та доступ до Web
  - Доступ до даних (ADO.NET)
  - Доступ до каталогів, файлової системи, реєстру
  - Атрибути, підтримка рефлексії
  - Доступ до ОС, взаємодія з COM
- Простори імен
- Категорії аплікацій
  - Настільні аплікації Windows Forms та Windows Presentation Foundation
  - Мережеві аплікації Web Forms, ASP.NET
  - Web-служби, служби Windows, транспортування WCF

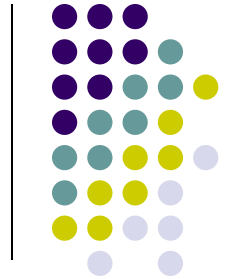
“Обов’язкова” програма



```
//using System;

namespace LNU.CSharp.Learn
{
    class Program
    {
        static void Main()
        {
            System.Console.WriteLine("Hello, WORLD!");
            System.Console.ReadLine();
            return;
        }
    }
}
```

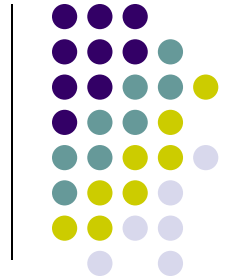
Типи даних C#



Цілі 5; -17; 0x1AB; 255U; 5L; 10UL

	Тип CTS	біт	Діапазон
sbyte	System.SByte	6	-128:127
short	System.Int16	16	-32768:32767
int	System.Int32	32	-2147483648: 2147483647
long	System.Int64	64	-9223372036854775808:9223372036854775807
byte	System.Byte	6	0:255
ushort	System.Uint16	16	0:65535
uint	System.Uint32	32	0:4294967295
ulong	System.Uint64	64	0:18446744073709551615

Типи даних C#



Дійсні 1.5; -2.09; 1e-6; .5; 12.3F

	Тип CTS	біт	зн	Діапазон
float	System.Single	32	7	1.5×10^{-45} : 3.4×10^{38}
double	System.Double	64	15	5.0×10^{-324} : 1.7×10^{308}

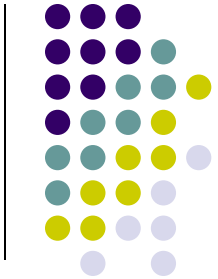
Десятковий (фінансовий) 12.50M

	Тип CTS	біт	зн	Діапазон
decimal	System.Decimal	128	28	1.0×10^{-28} : 7.9×10^{28}

Логічний

	Тип CTS	Значення
bool	System.Boolean	true | false

Типи даних C#



Символьний 'A'; (char)65; '\u0041'; '\x0041'

	Тип CTS	біт	Значення
char	System.Char	16	Unicode-символ

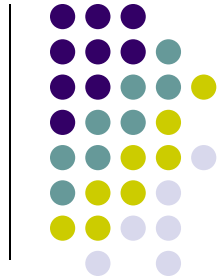
спеціальні літери (взяти в апострофи)

\'	апостроф	\a	звук	\r	переведення каретки
\"	лапки	\b	повернення	\t	горизонтальна табуляція
\\	обернена коса	\f	форма	\v	вертикальна табуляція
\0	(char)0	\n	кінець рядка		

Вбудовані типи посилання

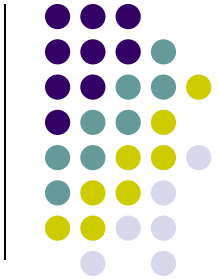
	Тип CTS	Значення
object	System.Object	Єдиний кореневий тип CTS GetType(), ToString(), …
string	System.String	Рядок символів Unicode "Hello!" @"C:\Learn\first.txt"

Окремі типи користувача



- Перелік (цілий тип користувача)
  - enum – нащадок System.Enum
  - public enum TimeOfDay
    { Morning=0, Afternoon=1,Evening=2 }
    TimeOfDay time = TimeOfDay.Afternoon;
    Console.WriteLine(time.ToString());
    TimeOfDay time2 = (TimeOfDay) Enum.Parse(
    typeof(TimeOfDay), "afternoon", true);
- Масив (посилання на структуру)
  - тип [ ] посилання
  - int [ ] arr = new int [32];
  - arr[0] = 5; arr[i] = arr[i-1]+3; … arr[31]…

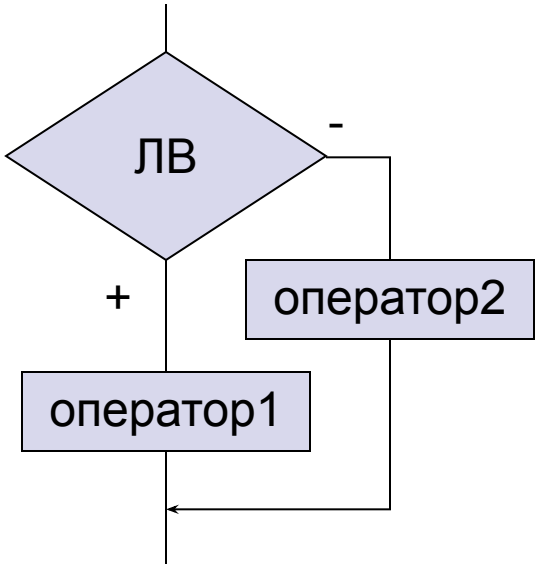
Оголошення змінних, констант



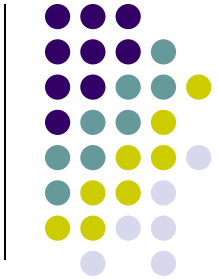
- Оголошення змінних
  - тип ім’я_змінної = значення; // ініціалізація !
  - double a = 1.57, s = 0.0;
  - { int k; … k = 1; int m = k + 1; …}
  - MyClass object = new MyClass();
  - var x = 5; // виведення типу
  - заборона перекриття локальних імен
- Константи
  - const тип символьне_ім’я = константне_значення;
  - const int size = 256; // неявно статичне значення
  - нема константних методів

Галуження потоку керування

- Вкорочений умовний оператор
  - if (логічний вираз) оператор | {блок операторів}
- Умовний оператор
  - if (логічний вираз)
    оператор | {блок операторів} // 1
    else
    оператор | {блок операторів} // 2
- Продовжений умовний оператор
  - if (логічний вираз)
    оператор | {блок операторів}
    else if (логічний вираз)
    оператор | {блок операторів}
    else …



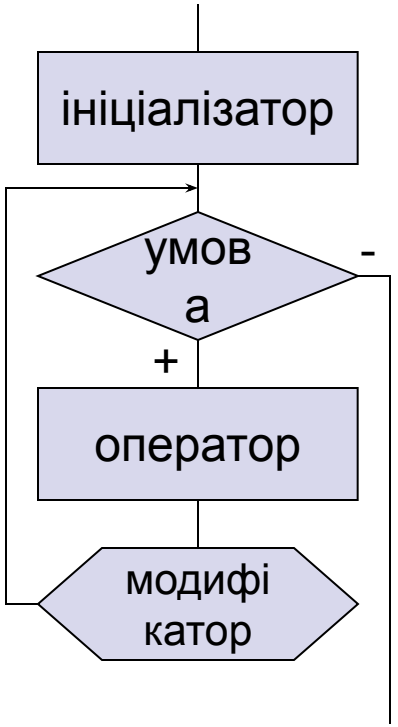
Галуження потоку керування



- Оператор вибору
  - switch (expression)
    {
      case constantValue1: operator(s)
        break;
     case constantValue2: operator(s)
        break;   …
     default: operator(s)
        break;
    }
  - switch (country)
    {
     case "au": case "uk": case "us":
       language = "English"; break;
    case "at": case "de":
       language = "German"; break;
    }

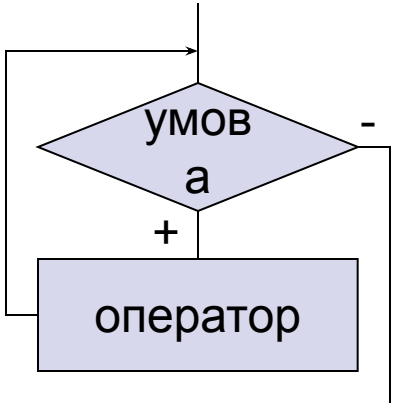
Повторення

- for (ініціалізатор; умова; модифікатор)
  оператор | блок
  - int f = 1; for (int i = 1; i<=n; ++i) f *= i;
  - for (int k = 1, j = 9; k != j; ++k, --j)
    System.Console.WriteLine(
    k.ToString() + " " + j.ToString());
  - for (int i = 0; i < 100; i += 10)
    {
    for (int j = i; j < i+10; ++j)
    {
    Console.WriteLine(" " + j);
    }
    }
  - for (; ;) DoSomethingUntilBreak();

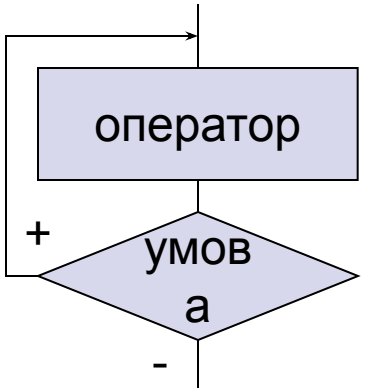


Повторення

- while (умова) оператор | блок
  - ```
    bool condition = true;
    while (condition)
    {
       DoSomeWork();
       condition = CheckCond();
    }
    ```

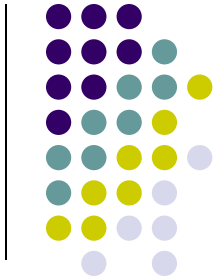


- do оператор | блок while (умова);
  - ```
    do
    {
       MustBeExecutedAtLeastOnce();
       condition = CheckCond();
    }
    while (condition);
    ```



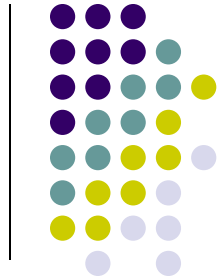
- foreach (var X in AnEnumerable) оператор | блок

Переходи



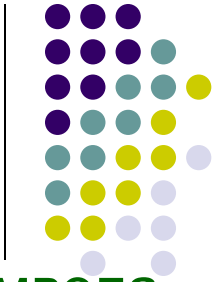
- goto Label;
  - вихід з вкладених циклів
  - перехід між альтернативами switch
  - заборонено входити в блок циклу чи галуження, виходити з класу чи блока finally
- break;
  - перериває виконання структурованого оператора
- continue;
  - перериває одну ітерацію циклу
- return; або return Value;
  - завершення виконання методу

Простори імен



- Структурування імен, логічне поєднання класів
- “Простір до файла” – “багато до багатьох”
  - namespace LNU {
    namespace CSharp {
    class Lesson01 {
    public string GetNamespace()
    { return this.GetType().Namespace; }
    }
    }
    }
  - namespace LNU.CSharp {
    class Lesson01 { … }
    }
  - LNU.CSharp. Lesson01 inst; … inst.GetNamespace();
  - using LNU.CSharp; Lesson01 inst = new Lesson01();
  - using S1 = LNU.CSharp; S1::Lesson01 inst …;

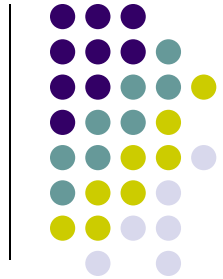
Консольне введення-виведення



- int System.Console.Read(); //код одного прочитаного символа
- string System.Console.ReadLine(); // прочитаний рядок
- int K = int.Parse(Console.ReadLine()); // несимвольні величини

- void System.Console.Write(aVal); // виведення відомого типу
- void System.Console.WriteLine(aVal); // те саме + кінець рядка
- Console.Write(K); Console.WriteLine(K.ToString());
  Console.WriteLine(" K = " + K);
- Console.WriteLine("наступне після {0} рівне{1,10:D4}",K,K+1);
  - C – грошовий формат; G – загальний (E або F);
  - D – десятковий (0015); N – числовий з розділювачами;
  - E – експотенційний; P – відсотковий;
  - F – з фіксованою комою; X – шістнадцятковий.

Для самостійного опрацювання



- Передавання аргументів у Main()
- Додаткові параметри компілятора
- Директиви препроцесора C#
- Домовленості щодо іменування
- Документація XML