|  |  |
| --- | --- |
| Загальний вигляд | Приклад |
| опис змінної без початкової ініцініалізації  <тип змінної> <ідентифікатор змінної>; | int a; |
| опис змінної з початковою ініціалізацією  <тип змінної> <ідентифікатор змінної> [ = <значення>] ; | int b=40;  double s=3.67; |
| **Опис констант**  const <тип константи> <ідентифікатор констати> = <значення>; | const int a = 10;  const float b = 2.5; |
| Виведення значення змінної  Console.WriteLine(“ назва змінної ={0}”, змінна ); | Console.WriteLine(“i={0}”,i);  Console.WriteLine(“d={0}”,d); |
| Введення змінної рядкового типу (string)  змінна = Console.ReadLine(); | string s;  s=Console.ReadLine(); |
| Введення змінної цілого типу (int)  Console.Write(" змінна ="); //Виводимо повідомлення  змінна =int.**Parse**(Console.ReadLine()); //Вводимо значення | int c;  Console.Write("c=");  c=int.**Parse**(Console.ReadLine()); |
| Введення змінної дійсного типу (double)  Console.Write(" змінна ="); //Виводимо повідомлення  змінна =double.**Parse**(Console.ReadLine()); //Вводимо значення | double d;  Console.Write("d=");  d=double.**Parse**(Console.ReadLine()); |
| Умовний оператор з двома альтернативами (щось треба робити і коли умова виконуєтсья, і коли умова не виконується)  if (<умова>)  {  <оператор1.1>;  . . . . . . . . . . .  <оператор1.N>;  }  else  {  <оператор2.1>;  . . . . . . . . . . .  <оператор2.M>;  } | if(x>y)  {  max=x;  min=y;  }  else  {  max=y;  min=x;  } |
| Умовний оператор з однією альтернативою (щось треба робити лише коли умова виконуєтсья; якщо умова не виконується, то нічого робити не потрібно)  if (<умова>)  {  <оператор 1>;  . . . . . . . . . . .  <оператор N>;  **}** | if(x>0)  {  z=1/x;  l=y/x;  } |
| **switch (<селектор вибору>)**  **{**  **case <константа 1> : <оператор 1>;**  **break;**  **case <константа 2> : <оператор 2>;**  **break;**  **………………………………………….**  **case <константа N> : <оператор N>;**  **break;**  **default : <оператор N+1>;**  **break;**  **}** | using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  namespace ConsoleApplication3  {  class Program  {  static void Main(string[] args)  {  int mark;  Console.Write("Mark = ");  mark = int.Parse(Console.ReadLine());  **switch (mark)**  **{**  **case 2: Console.WriteLine("Незадовільно");**  **break;**  **case 3: Console.WriteLine("Задовільно.");**  **break;**  **case 4: Console.WriteLine("Добре");**  **break;**  **case 5: Console.WriteLine("Відмінно");**  **break;**  **default: Console.WriteLine("Неправильна**  **оцінка.");**  **break;**  **}**  Console.ReadKey();  }  }  } |
| while (<умова>)  <оператор>; | Знайти суму перших n чисел.  int sum=0;  int i=1;  while(i<=n)  sum=sum+i++; |
| while (<умова>)  {  <оператор 1>;  . . . . . . . . . . .  <оператор N>;  **}** | **Приклад.** Знайти суму і добуток перших n чисел.  int sum=0;  int mult=1;  int i=1;  while(i<=n)  **{**  sum=sum+(i);  mult=mult\*(i++);  **}** |
| do  {  <оператор>;  }  while (<умова>); | **Приклад.** Знайти суму перших n чисел.  int sum=0;  int i=1;  do  {  sum=sum+i++;  }  while(i<=n); |
| for (<вираз ініціалізації>;<логічна умова>;<ітераційна частина>)  <оператор>; | **Приклад.** Знайти суму перших n чисел.  int n,sum=0;  Console.WriteLine("n=");  n=int.Parse(Console.ReadLine());  **for (int i = 1; i <= n; i++)**  **sum = sum + i;** |
| Опис і виділення пам’яті для одновимірного масиву  **<тип елем.>** [ ] <ім’я масиву> **= new <тип елем.> [<кількість елементів>];** | //--------Коли знаємо наперед кількість елментів  byte [ ] **b = new byte [20];**  double [ ] **d = new double[50];**  **//---------- коли кількість задає користувач**  int n; // n - кількість елементів  Console.Write("n=");  n = int.Parse(Console.ReadLine());  int [ ] a = new int[n]; |
| Введення елементів масиву  for(<параметр>= 0; <параметр> < <кільк.елем.>; <параметр>++)  {  Console.Write (“<ім’я мас.>[ {0} ]= ”,<параметр>);  <ім’я мас.>[ <параметр> ] =<тип елементів>.Parse(Console.ReadLine());  } | **for (int i = 0; i < a.Length; i++)**  **{**  **Console.Write("a[{0}]=",i);**  **a[i]= double.Parse(Console.ReadLine());**  **}** |
| Виведеня елементів масиву  for(<параметр>= 0; <параметр> < <кільк.елем.>; <параметр>++)  {  Console.WriteLine(“<ім’я мас.>[ {0} ]={1} ”,<параметр>,<ім’я мас.>[ <параметр>]);  } | **for (int i = 0; i < n; i++)**  **{**  **Console.WriteLine("a[{0}]={1}",i,a[i]);**  **}** |
| Можлива загальна форма схеми обробки масивів  for ( вказуємо як змінюються номери елементів масиву )  {  тут пишемо, що потрібно зробити з кожним із елементів масиву з вказаним номером  } | double Sum = 0;  for (int i = 0; i < n; i++) {  Sum+= a[i];  } |
| *Опис двовимірного масиву*  <тип елементів> [ , ] <ім’я масиву>; | int [ , ] A;  double [ , ] d; |
| *Виділення пам’яті для двовимірного масиву*  **<тип елем.> [ , ] <ім’я масиву>=new <тип елементів>[кільк. рядків, кількі. стовпців];** | int [ , ] A= new int[ 30,20];  double [ , ] d = new double [5 , 3]; |
| *Звертання до елементів масиву*  <ім’я масиву> [ <номер рядка > , <номер стовпця > ] | А[2,7]=21;  d[1,5]=34; |
| **ВВЕДЕННЯ ДВОВИМІРНОГО МАСИВУ**  //1.Вводимо кількість рядків  Console.Write("Row count=");  int rowCount = int.Parse(Console.ReadLine());  //2.Вводимо кількість стовпців  Console.Write("Column count=");  int columnCount = int.Parse(Console.ReadLine());  //3. Виділяємо пам"ять  double[,] a = new double[rowCount,columnCount];  //4.Вводимо елементи масиву  for (int i = 0; i < a.GetLength(0); i++) //Вказуємо як змінюються номери рядків «і»  {  for (int j = 0; j < a.GetLength(1); j++) //Вказуємо як змінюються номери стовпців «j»  {  //Вказуємо що потрібно зробити з кожним із елементів a[i,j]  //які знаходяться у рядку з номером "і" та стовпці з номером "j"  Console.Write("a[{0},{1}]=",i,j);  a[i, j] = double.Parse(Console.ReadLine());  }  } | |
| **ВИВЕДЕННЯ ДВОВИМІРНОГО МАСИВУ**  for (int i = 0; i < a.GetLength(0); i++) //Вказуємо як змінюються номери рядків  {  for (int j = 0; j < a.GetLength(1); j++) //Вказуємо як змінюються номери стовпців  {  //Вказуємо що потрібно зробити з кожним із елементів a[i,j]  //які знаходяться у рядку з номером "і" та стовпці з номером "j"  Console.Write("{0,7}",a[i,j]);  }  Console.WriteLine();  } | |
| **Приклад використання операцій над двовимірними масивами**  //--- Знайти суму усіх елементів  double sum = 0;  for (int i = 0; i < a.GetLength(0); i++) //Вказуємо як змінюються номери рядків  {  for (int j = 0; j < a.GetLength(1); j++) //Вказуємо як змінюються номери стовпців  {  //Вказуємо що потрібно зробити з кожним із елементів a[i,j]  //які знаходяться у рядку з номером "і" та стовпці з номером "j"  sum += a[i, j];  }  } | |
| Загальний вигляд функції  **static <тип функції> <ім’я функції>(<список форм. парам.>)**  {  //тіло методу  }  (поки що **<список форм. парам.>**– це те, що треба знати, щоб функція могла виконуватися ) | Знайти максимальне з трьох цілих чисел  **static int getMax(int c1,int c2,int c3)**  **{//Знаходимо результат**  **int m = c1;**  **if (c2 > m)**  **m = c2;**  **if (c3 > m)**  **m = c3;**  **//Повертаємо результат**  **return m;**  **}**  //Приклад виклику  static void Main(string[] args)  {  int a,b,c;  Console.Write("a=");  a = int.Parse(Console.ReadLine());  Console.Write("b=");  b = int.Parse(Console.ReadLine());  Console.Write("c=");  c = int.Parse(Console.ReadLine());  int max = **Max(a,b,c)**;  Console.Write("max={0}",max);  } |
| Якщо хочемо, щоб змінити значення фактичних параметрів у функції, то потрібно описувати з модифікатором **ref** .  static <тип функції> <ім’я функції>( **ref** тип ім’я парам. …)  {  //тіло методу  }  При цьому:   * тип формального параметра повинен точно співпадати з типом фактичного параметра; * фактичного параметр має бути ініціалізований перед викликом функції; | **Приклад.** Створити функцію, яка більше значення зменшує на 2, а менше збільшує на 3.  class Program  {  **static void Fun1(ref int a, ref int b)**  **{**  **if (a > b)**  **{**  **a -= 2;**  **b += 3;**  **}**  **else**  **{**  **b -= 2;**  **a += 3;**  **}**  **}**  static void Main(string[] args)  { int n,m;  Console.WriteLine("n=");  n=int.Parse(Console.ReadLine());  Console.WriteLine("m=");  m = int.Parse(Console.ReadLine());  **Fun1(ref n,ref m);**  Console.WriteLine("n= {0} m={1}",n,m);  Console.ReadKey();  } |
| Якщо хочемо з функції повернути декілька результатів, то їх треба можна повернути за допомогою формальних параметрів посилань, описаних з використанням модифікатора **out**  При цьому:   * тип формального параметра повинен співпадати з типом фактичного параметра; * в середині функції обов’язково потрібно формальному параметру надати значення . | **Приклад.** Створити функцію, яка повертає максимальне та мінімальне значення з двох дійсних чисел.  class Program  {  static void MinMax(int c1, int c2, out int max, out int min)  {  if (c1 > c2)  {  max = c1;  min = c2;  }  else  {  max = c2;  min = c1;  }  }  static void Main(string[] args)  { int n,m;  Console.WriteLine("n=");  n=int.Parse(Console.ReadLine());  Console.WriteLine("m=");  m = int.Parse(Console.ReadLine());  int max, min;  **MinMax(n, m, out max, out min);**  Console.WriteLine("max= {0} min={1}",max,min);  Console.ReadKey();  }  } |
|  |  |
|  |  |
|  |  |