**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**Чорноморський національний університет   
імені Петра Могили**

**Факультет комп'ютерних наук**

**Кафедра інженерії програмного забезпечення**

**ЗВІТ**

з лабораторної роботи № 2

**«Масиви. Клас Object. Структури.»**

**Варіант 5**

Дисципліна «Об'єктно-орієнтоване програмування»

Спеціальність: **Інженерія програмного забезпечення**

121-ЛР.02-208.22120802

***Студент\_\_Д. М. Дзундза***

*(підпис)*

*\_\_09.12.2021* \_\_

*(дата)*

***Викладач \_\_\_\_\_\_\_\_С. Ю. Боровльова***

*(підпис)*

*\_\_\_\_\_\_\_\_\_\_\_\_\_*

*(дата)*

Миколаїв – 2021

**Лабораторна робота №2**

|  |  |
| --- | --- |
| Тема роботи: | Масиви. Клас Object. Структури. |

|  |
| --- |
| Мета роботи: |

Ознайомитись з типами масивів в C# та принципами роботи з ними, призначенням класу Object та його основними методами, властивостями структур. **Завдання:**

Створіть застосунок в якому:

1. Оголосіть два одновимірних масиви:

1.1. З ініціалізацією елементів масиву та без ініціалізації.

1.2. Студенти з парними номерами варіантів підраховують суму елементів масивів, з непарними - добуток.

1.3. Обов‘язково розмір масиву підраховувати програмно.

2. Оголосіть двовимірний прямокутний масив:

2.1. З ініціалізацією елементів масивів та без ініціалізації.

2.2. Студентам з парними номерами варіанті підрахувати добуток елементів парних рядків масивів, з непарними - суму непарних стовпчиків масивів.

2.3. Увага! Розмір рядків та стовпчиків підраховувати програмно.

3. Оголосіть двовимірний двовимірний невирівняний масивами:

3.1. Присвойте значення елементам масиву будь-яким зручним для Вас способом.

3.2. Студентам з парними номерами варіанті підрахувати добуток елементів парних рядків масивів, з непарними - суму непарних стовпчиків масивів.

4. За допомогою оператора foreach підрахувати в кожному із трьох типів масивів кількість елементів із заданим значенням(значення вводиться з клавіатури).

5. Оголосіть клас згідно свого варіанту із наведеного нижче списку.

5.1. Продемонструйте використання модифікаторів доступу в межах однієї збірки.

5.2. Створіть два об‘єкти класу.

5.3. Використайте методи Equales та ToString для цих об‘єктів.

5.4. Перевизначте метод Equales для свого класу.

5.5. Перевизначте метод ToString для свого класу.

5.6. Використайте методи Equales та ToString для цих об‘єктів.

6. Виконай наступні дії із структурою:

6.1. Оголосіть всі можливі конструктори.

6.2. Продемонструйте роботу із всіма можливими конструкторами структури.

6.3. Покажіть в коді неможливість використання структури якості базової.

6.4. Покажіть в коді неможливість наслідування структурою іншої структури або класу.

6.5. Продемонструйте роботу методу Equals та поясніть отриманий результат.

6.6. Продемонструйте роботу стандартного методу ToString. Перевизначте цей метод.

**Лістинг коду:**

using System;

namespace Lection2

{

class WorkerBase

{

protected string \_name;

protected int \_age;

public string Name { get { return \_name; } set { \_name = value; } }

public int Age { get { return \_age; } set { \_age = value; } }

public WorkerBase() { }

public WorkerBase(string name, int age)

{

Name = name;

Age = age;

}

}

class Worker : WorkerBase

{

public Worker() { }

public Worker(string name, int age) : base(name, age) { }

public override string ToString()

{

return $"name: {Name}; age: {\_age}";

}

public override bool Equals(object obj)

{

if ((obj == null) || !this.GetType().Equals(obj.GetType()))

{

return false;

}

else

{

Worker p = (Worker)obj;

return (Name == p.Name) && (Age == p.Age);

}

}

}

public struct WorkerStruct

{

public WorkerStruct(string name, int age)

{

Name = name;

Age = age;

}

public WorkerStruct(string name)

{

Name = name;

Age = 18;

}

public WorkerStruct(int age)

{

Name = "";

Age = age;

}

public string Name { get; set; }

public int Age { get; set; }

public override string ToString() => $"Name: {Name}; age: {Age}";

}

public struct WorkerStruct1

{

public WorkerStruct1(string name, int age)

{

Name = name;

Age = age;

}

public string Name { get; set; }

public int Age { get; set; }

}

/\*struct a: WorkerStruct {

}\*/

/\*class a: WorkerStruct {

}\*/

class Program

{

static void Main(string[] args)

{

var rand = new Random();

int max\_rand\_value = 10;

//Завдання 1

int[] array1 = new int[] { 5, 8, 1, 30 }; // Инициализация

int[] array2; //Без инициализации

array2 = new int[6]; //Расположение в памяти

array2[0] = 0; //Присвоение каждому елементу значение

array2[1] = 2;

array2[2] = 35;

array2[3] = 33;

array2[4] = 46;

array2[5] = 99;

int product = 1;

for (uint i = 0; i < array1.Length; i++)

{

product \*= array1[i];

}

Console.WriteLine($"Произведение с инициализацией {product}");

Console.WriteLine($"Длинна массива с инициализациией {array1.Length}");

Console.WriteLine();

product = 1;

foreach (var value in array2)

{

product \*= value;

}

Console.WriteLine($"Произведение без инициализации {product}");

Console.WriteLine($"Длинна массива без инициализации {array2.GetLength(0)}");

Console.WriteLine();

//Завдання 2

int[,] squaredArray1 = new int[,] { {1, 2, 3, 4 },

{5, 6, 7, 8},

{9, 10, 11, 12} };

Console.WriteLine($"Высота двумерного массива с инициализацией {squaredArray1.GetLength(0)}");

Console.WriteLine($"Длинна двумерного массива с инициализацией {squaredArray1.GetLength(1)}");

int result = 0;

for (int i = 0; i < squaredArray1.GetLength(0); i++)

for (int j = 0; j < squaredArray1.GetLength(1); j++)

if (j % 2 != 0)

result += squaredArray1[i, j];

Console.WriteLine($"Сумма нечётных столбцов матрицы с инициализацией = {result}");

Console.WriteLine();

Console.Write("Введите n: ");

string input = Console.ReadLine();

Console.WriteLine();

int n, m;

int.TryParse(input, out n);

Console.Write("Введите m: ");

input = Console.ReadLine();

Console.WriteLine();

int.TryParse(input, out m);

int[,] squaredArray2 = new int[n, m];

Console.WriteLine("Rank: " + squaredArray2.Rank);

for (int i = 0; i < squaredArray2.GetLength(0); i++)

for (int j = 0; j < squaredArray2.GetLength(1); j++)

{

squaredArray2[i, j] = rand.Next(max\_rand\_value);

Console.WriteLine($"i:{i}, j:{j}, value:{squaredArray2[i, j]}");

}

Console.WriteLine($"Высота двумерного массива без инициализацией {squaredArray1.GetLength(0)}");

Console.WriteLine($"Длинна двумерного массива без инициализации {squaredArray2.GetLength(1)}");

result = 0;

for (int i = 0; i < squaredArray2.GetLength(0); i++)

for (int j = 0; j < squaredArray2.GetLength(1); j++)

if (i % 2 != 0)

result += squaredArray2[i, j];

Console.WriteLine($" Сумма нечётных строк матрицы без инициализации = {result}");

Console.WriteLine(Environment.NewLine);

int r;

while (true)

{

Console.Write("Введите количество строк: ");

input = Console.ReadLine();

Console.WriteLine();

if (int.TryParse(input, out r))

{

break;

}

Console.WriteLine("Вы ввели не правильное количество строк! Повторите попытку!");

}

// Создание и заполнение массива

int[][] jagged = new int[r][];

for (int i = 0; i < r; i++)

{

while (true)

{

Console.Write($"Введите количество столбчиков в {i}-ой строке: ");

input = Console.ReadLine();

Console.WriteLine();

if (int.TryParse(input, out m))

{

jagged[i] = new int[m];

for (int j = 0; j < m; j++)

{

jagged[i][j] = rand.Next(max\_rand\_value);

}

break;

}

Console.WriteLine("Вы ввели не правильное количество столбцов! Повторите попытку!");

}

}

result = 0;

for (int i = 0; i < jagged.GetLength(0); i++)

for (int j = 0; j < jagged[i].Length; j++)

if (j % 2 != 0)

result += jagged[i][j];

Console.WriteLine($"Сумма нечётных столбцов матрицы = {result}");

Console.WriteLine(Environment.NewLine);

int value\_to\_cmp;

while (true)

{

Console.Write($"Введите значение для сравнения: ");

input = Console.ReadLine();

Console.WriteLine();

if (int.TryParse(input, out value\_to\_cmp))

{

break;

}

Console.WriteLine("Вы ввели не правильное значение для сравнения! Повторите попытку!");

}

int repeats = 0;

foreach (var value in array1)

{

if (value == value\_to\_cmp)

{

repeats += 1;

}

}

Console.WriteLine($"Колво повторов в 1D масиве- {repeats}!");

repeats = 0;

foreach (var value in squaredArray1)

if (value == value\_to\_cmp)

repeats += 1;

Console.WriteLine($"Колво повторов в 2D масиве - {repeats}!");

repeats = 0;

foreach (var row in jagged)

foreach (var value in row)

if (value == value\_to\_cmp)

repeats += 1;

Console.WriteLine($"Колво повторов в 2D невыровняном масиве - {repeats}!");

Console.WriteLine();

Console.ReadKey();

Console.Clear();

WorkerBase wb1 = new WorkerBase("Кристина", 12);

WorkerBase wb2 = new WorkerBase("Кристинаa", 12);

Console.WriteLine($"wb1: {wb1.ToString()}");

Console.WriteLine($"wb1 == wb2: {wb1.Equals(wb2)}");

Console.WriteLine($"wb2 == wb1: {wb2.Equals(wb1)}");

Console.WriteLine($"wb1 == wb1: {wb1.Equals(wb1)}");

Console.WriteLine();

Worker w1 = new Worker();

Worker w2 = new Worker("Дмитро", 18);

w1.Name = "Дмитро";

w1.Age = 19;

Console.WriteLine($"w1: {w1.ToString()}");

Console.WriteLine($"w2: {w2.ToString()}");

Console.WriteLine($"w1 == w2: {w1.Equals(w2)}");

Console.WriteLine($"w2 == w1: {w2.Equals(w1)}");

Console.WriteLine($"w2 == w2: {w2.Equals(w2)}");

Console.WriteLine();

WorkerStruct ws1 = new WorkerStruct("Андрей", 15);

WorkerStruct ws2 = new WorkerStruct("Игорь");

WorkerStruct ws3 = new WorkerStruct(75);

Console.WriteLine($"ws1: {ws1.ToString()}");

Console.WriteLine($"ws2: {ws2.ToString()}");

Console.WriteLine($"ws3: {ws3.ToString()}");

//w1.\_name; - Буде помилка через модифікатор доступу

Console.ReadKey();

}

}

}

**Результат роботи програми:**

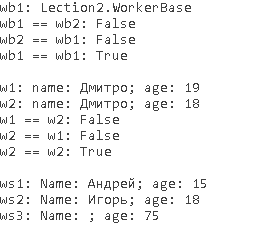


Рисунок 1 – Результат роботи програми з класами

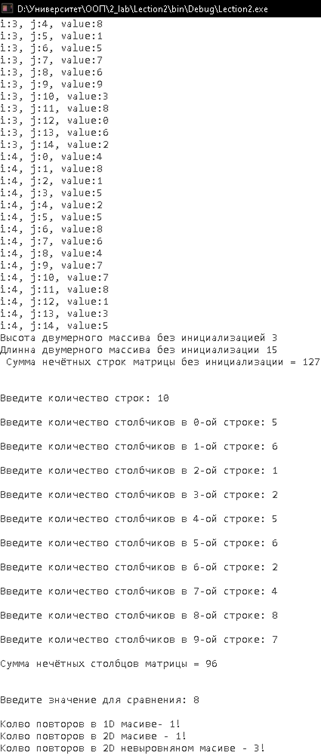
****

Рисунок 2 – Результат роботи програми з масивами

**Питання до захисту:**

1. Масиви бувають одномірними, багатомірними(прямокутними чи невирівнянними).

2. Оголошення без ініціалізації одновимірного масиву

int[] arr = new int[7];

Оголошення з ініціалізацією одновимірного масиву

int[] arr1 = { 1, 2, 3, 4, 5, 6, 7 };

Оголошення без ініціалізації двовимірного масиву

int[,] myArr = new int[5, 6];

Оголошення з ініціалізацією двовимірного масиву

int[,] myArr1 = { { 1, 2, 3, 4, 5 }, { 3, 5, 7, 8, 9 } };

3. Підрахувати кількість рядків та стовпчиків можна за допомогою метода GetLength() чи просто Length.

4. Вони будуть вказувати на один і той самий масив, копія створюватися не буде. Якщо треба щоб змінні вказували на різні ділянки в пам’яті треба використовувати оператор “new” для кожної з них.

5. Для прямокутних двовимірних масивів, foreach проходиться по всім стовпцям та строкам. У невирівняних масивів, foreach проходить тільки по рядкам.

6. public – доступно всім з будь-якої збірки; private – доступно тільки в цьому класі; protected - доступно в цьому класі та в похідних класах будь-якої збірки; internal - доступно всім тільки в поточній збірці (public); protected internal - доступно в поточній збірці всім (public), в інших збірках –тільки похідним класам (protected) з обмеженнями (якщо звернення йде через змінну типу похідного класу); private protected – доступно тільки в похідних класах поточної збірки.

7. Клас Object є базовим для всіх типів даних. Клас має наступні методи: Equals- порівнює адреси об’єктів; ReferenceEquals – також порівнює адреси об’єктів; GetHashCode – повертає значення для розрахунку; GetType – повертає тип об’єкту; ToString – повертає рядок в якому міститься інформація про тип об’єкту який зазвичай перевизначається в класі.

8. Оператор is використовується для порівняння типів змінних. Оператор as використовується для приведення до типу.

9. Структури не можна використовувати в якості базового та структури не можуть наслідувати класи( але можуть реалізувати інтерфейси). Структурам не можна присвоїти початкове значення, якщо вони не є статичними. В конструкторі не можна оголосити конструктор без параметрів. При використанні конструктора за замовчуванням, поля будуть ініціалізовані за замовчуванням

10. Структури є більш ефективнішими при малій величіні структури. Класи ефективніші, коли використовується наслідування та об’єкти є більшими.

**Висновок:**

Протягом процесу виконання лабораторної роботи було створено 1 консольний застосунок що використовує різні типи масивів в С#. Було показано різні принципи роботи з ними. Представлено призначення класу Object та його основні методи.