Arrays & Objects

code
academy

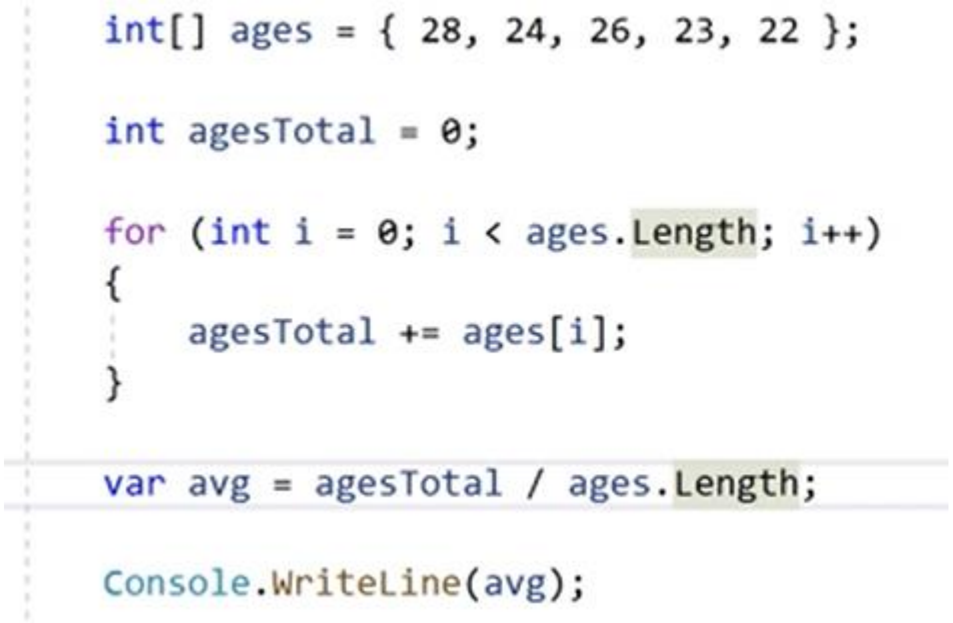
Recap

- **Git**
    - Why use it ?

Arrays (Massiv)

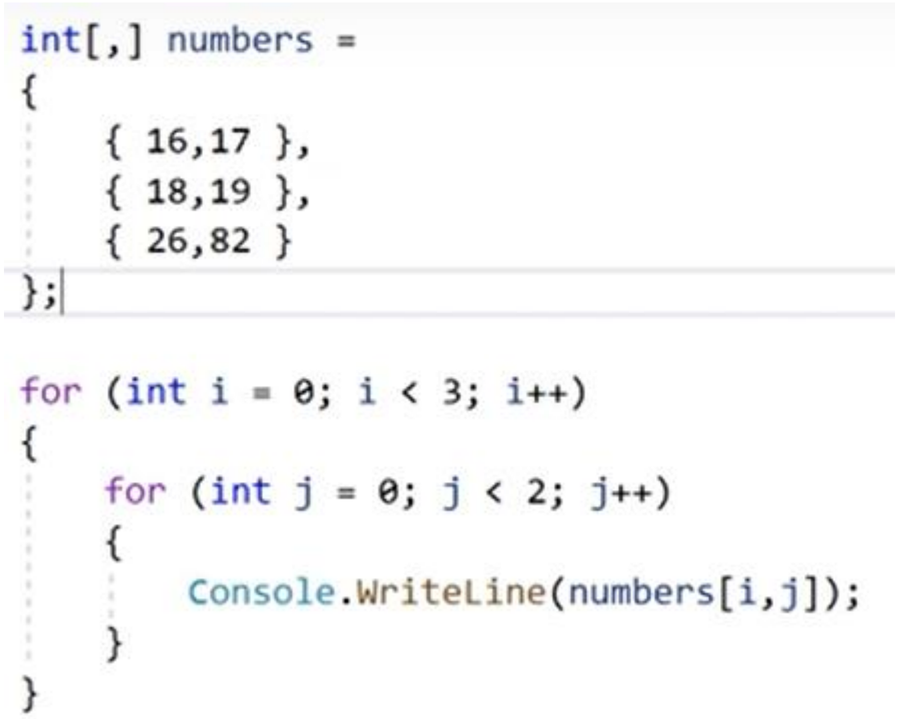
- “an impressive display or range of a particular type of thing”
- Dimensions
  - 1,2,3, … n dimensions
  - Matrices

Use of Arrays



```
int[] ages = { 28, 24, 26, 23, 22 };

int agesTotal = 0;

for (int i = 0; i < ages.Length; i++)
{
    agesTotal += ages[i];
}

var avg = agesTotal / ages.Length;

Console.WriteLine(avg);
```

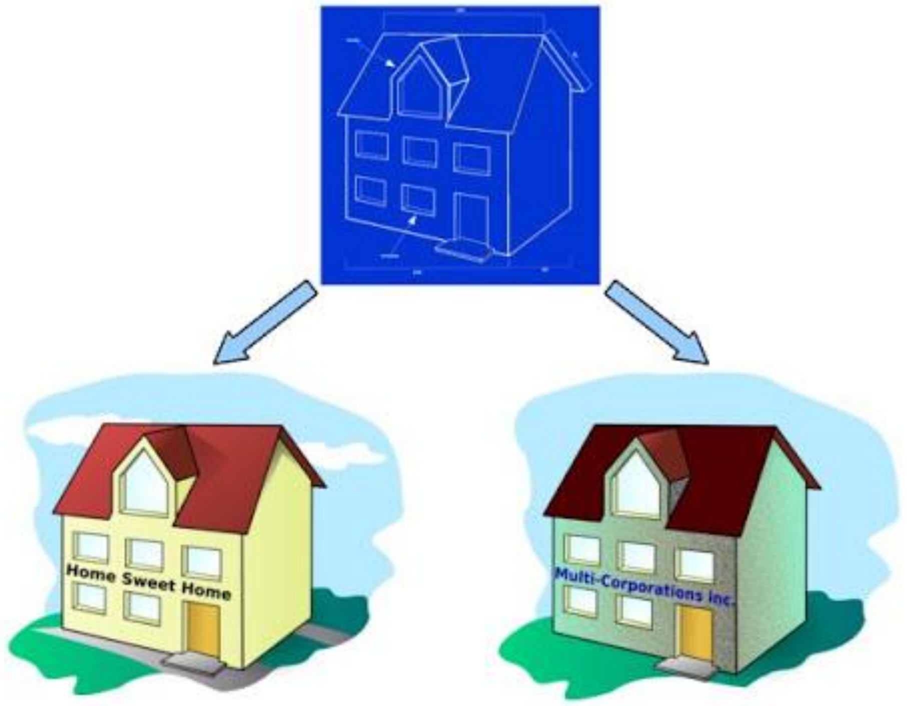
Multidimension & Nested Loops



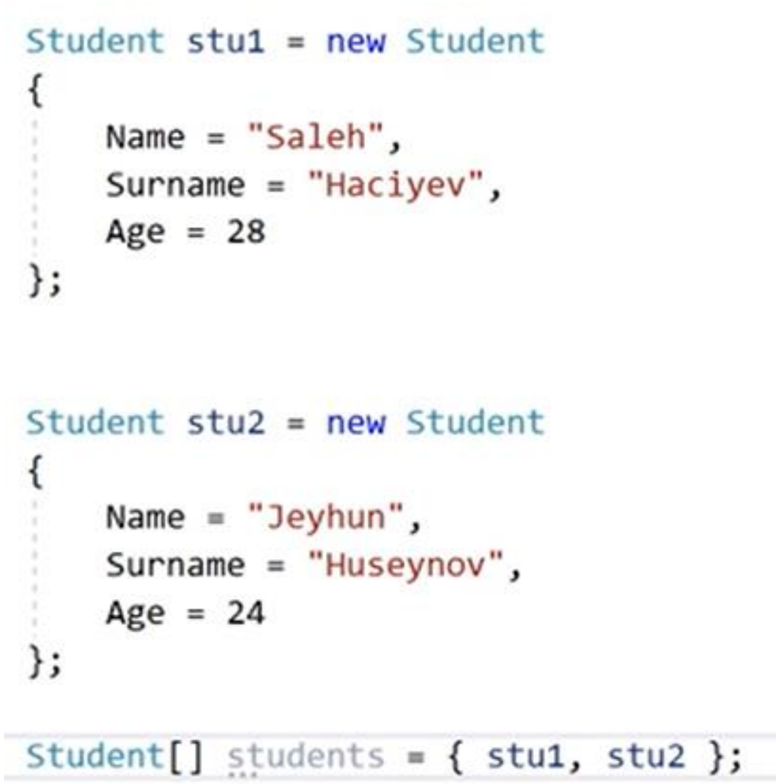
```
int[,] numbers =
{
    { 16,17 },
    { 18,19 },
    { 26,82 }
};

for (int i = 0; i < 3; i++)
{
    for (int j = 0; j < 2; j++)
    {
        Console.WriteLine(numbers[i,j]);
    }
}
```

Object & Class

- array.Length ?
- Class
  - Properties
  - Obj = new ...
    - Initialization
  - Setting
  - Getting



Objects and Arrays



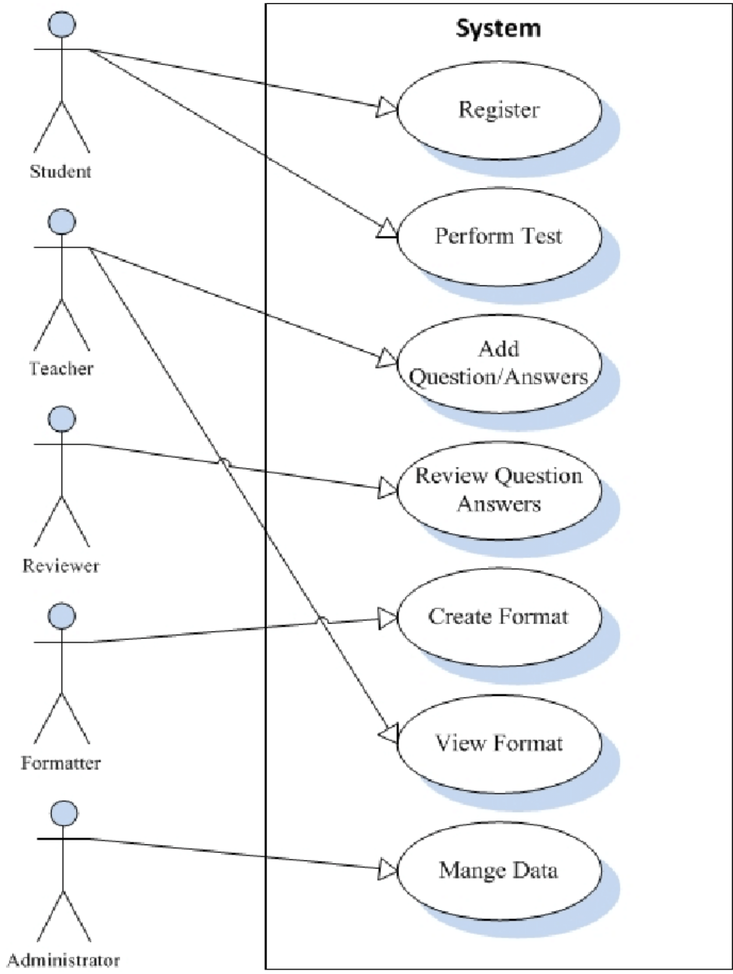
```
Student stu1 = new Student
{
    Name = "Saleh",
    Surname = "Haciyev",
    Age = 28
};


Student stu2 = new Student
{
    Name = "Jeyhun",
    Surname = "Huseynov",
    Age = 24
};

Student[] students = { stu1, stu2 };
```

More

- Build-in Classes
  - String, Math, etc.
- **Object Oriented Programming - OOP**
- Functions, Properties
  - IsLetter(), ToUpper(), etc.

Use Case Diagram

- What is is and how to build it?
- **Path from Project Requirements → Classes and Functions**
- Actors
- Nouns - Cases
- Verbs - Actions
- Simulation



Methods and Encapsulation

- Private vs Public
- Methods

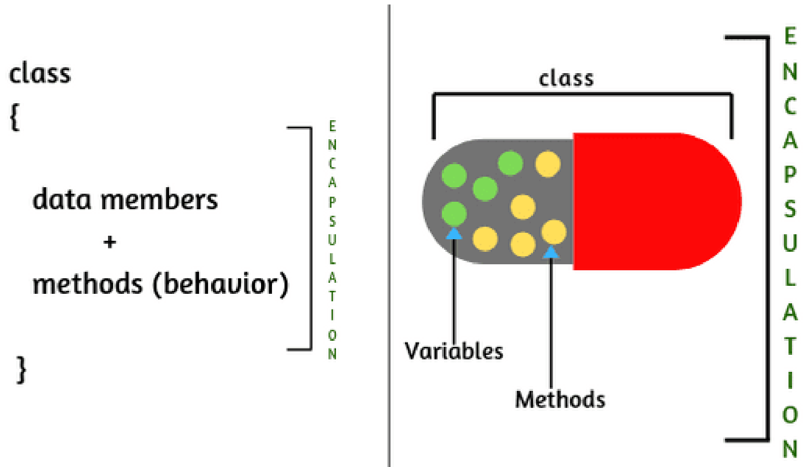
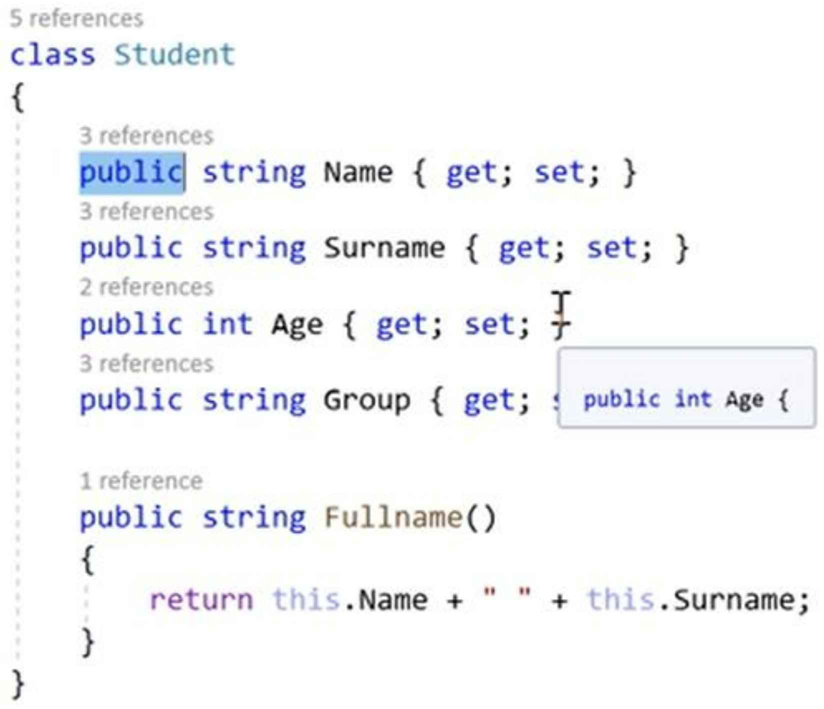


Fig: Encapsulation



```
5 references
class Student
{
    3 references
    public string Name { get; set; }
    3 references
    public string Surname { get; set; }
    2 references
    public int Age { get; set; }
    3 references
    public string Group { get; ;  public int Age {

    1 reference
    public string Fullname()
    {
        return this.Name + " " + this.Surname;
    }
}
```