Lists and Arrays

List

- Ordered or sequenced collection of objects
- A way to move around many values together
- Elements can be accessed in the list via an index
  - Index is an integer that starts at 0
- Lists can contain duplicate entries
- A wrapper around an Array

List Example

```
List<string> instructors = new List<string>();
instructors.Add("Mike");
instructors.Add("Nevin");
instructors.Add("Brad");

foreach (string instructor in instructors)
{
    Console.WriteLine(instructor);
}
```

Output:

```
Mike
Nevin
Brad
```

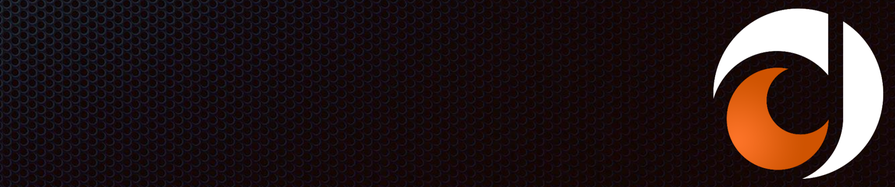
List Example

```
List<string> instructors = new List<string>();
instructors.Add("Mike");
instructors.Add("Nevin");
instructors.Add("Brad");
instructors.Add("Ian");
instructors.Remove("Nevin");
instructors.RemoveAt(1);

foreach (string instructor in instructors)
{
    Console.WriteLine(instructor);
}
```

Output:

```
Mike
Ian
```



List

- A way to create the list size and add values at the same time:

```
List<int> numbers = new List<int>() { 1, 2, 3 };

foreach (int number in numbers)
{
    Console.WriteLine(number);
}
```

List

- Lists don’t have to be of just data type string, int, boolean, etc.
- In Object-Oriented Programming, it may be more useful to have a list of a user-defined class or object

List Example

- For example, there is a class Dog:

```
Dog spot = new Dog("Spot");
Dog lambeau = new Dog("Lambeau");
List<Dog> dogs = new List<Dog>();

dogs.Add(spot);
dogs.Add(lambeau);

foreach(Dog dog in dogs)
{
    Console.WriteLine($"Name: {dog.name}");
}
```

```
1 reference
class Dog
{
    public string name;

    0 references
    public Dog(string name)
    {
        this.name = name;
    }
}
```

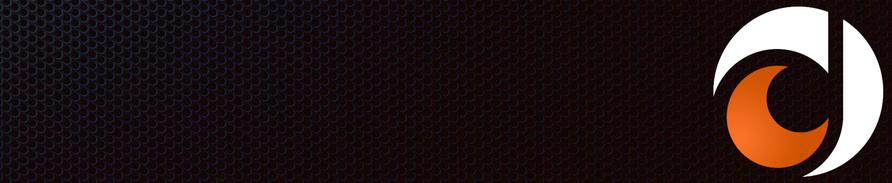
Output:

```
Name: Spot
Name: Lambeau
```

List Example

- List can be passed as an argument to another method

```
public void PrintDogsInGroup(List<string> dogNames)
{
    foreach (string name in dogNames)
    {
        Console.WriteLine(name);
    }
}
```



List

- Now we can call the method in the Program class and pass the List as an argument

```
class AnimalShelter
{
    0 references
    public void PrintDogsInGroup(List<string> dogNames)
    {
        foreach(string name in dogNames)
        {
            Console.WriteLine(name);
        }
    }
}
```

```
class Program
{
    0 references
    static void Main(string[] args)
    {
        Dog spot = new Dog("Spot");
        Dog lambeau = new Dog("Lambeau");
        AnimalShelter shelter = new AnimalShelter();
        List<string> dogNames = new List<string>();

        dogNames.Add(spot.name);
        dogNames.Add(lambeau.name);
        shelter.PrintDogsInGroup(dogNames);
    }
}
```

Array

- Arrays store a fixed-size collection of elements of the same data type
- In order to use an array, you must declare a variable to reference the array:

```
string[] packersArray;
```

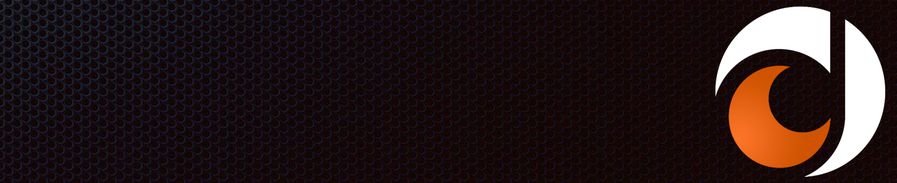
- In the above example, packersArray will reference the string array

Array

- One of the main differences between an array and a list is you must declare the size of the array on creation.
- To initialize an array, you must use the new keyword to create an instance of the array in memory:

```
string[] packersArray = new string[5];
```

- In the above example, packersArray holds five values, with indices from 0 to 4.



Array

- To add values to packersArray:

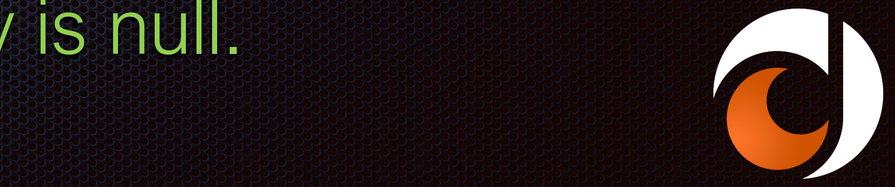
```
public void AddPackerChant()
{
    packersArray[0] = "Go";
    packersArray[1] = "Pack";
    packersArray[2] = "Go";
    packersArray[3] = "!";

    foreach (string item in packersArray)
    {
        Console.WriteLine(item);
    }
}
```

Output:

```
Go
Pack
Go
!
```

* Notice the blank space after the '!'. This shows that the fifth index, or [4], in the array is null.



Array

- A way to create the array size and add values at the same time:

```
string[] instructorArray = { "Hello", "I'm", "an", "instructor" };
```

- instructorArray has an array size of 4, with string values “Hello” at index 0, “I’m” at index 1, “an” at index 2, and “instructor” at index 3;

Review Questions

- What is a List? Give an example of why you should use one?
- What is an array? Give an example of why you should use one?
- What is the difference between a List and an Array?
- When should you use an Array over a List?