Data types

- enum

enum

```
enum Sizes { None, Small, Medium, Large, Jumbo }
static void Main(string[] args)
{
}
```

enum

```
public enum Days
   { Sunday, Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, }
   public enum Months
   { January = 1, February, March, April, May, June, July, August, September,
October, November, December }
   static void Main(string[] args)
   {
      Console.WriteLine("enum example");
      Days day = Days.Sunday;
      Console.WriteLine(day + " = " + (int)day);
      Months month = Months.September;
      Console.WriteLine($"{month} = {(int)month}");
      Console.WriteLine();

      // Type Safety
      Fn(month, day); // works as it should
      Fn(day, month); // compiler error because of order of arguments
      Fn(Months.April, Days.Tuesday); // works as it should
      Fn(4, 2);   // compiler error becuase arguments are integers
      Fn((Months)5, (Days)6); // works as it should - typecasting
}

static void Fn(Months m, Days d)
{
   Console.WriteLine($"{m} {d}");
}
```

enum

```
public enum Days
   { Sunday, Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, }
   public enum Months
   { January = 1, February, March, April, May, June, July, August, September,
October, November, December }
   static void Main(string[] args)
   {
      Console.WriteLine("enum example");
      Days day = Days.Sunday;
      Console.WriteLine(day + " = " + (int)day);
      Months month = Months.September;
      Console.WriteLine($"{month} = {(int)month}");
      Console.WriteLine();

      // Type Safety
      Fn(month, day); // works as it should
      Fn(day, month); // compiler error because of order of arguments
      Fn(Months.April, Days.Tuesday); // works as it should
      Fn(4, 2);   // compiler error becuase arguments are integers
      Fn((Months)5, (Days)6); // works as it should - typecasting
}

static void Fn(Months m, Days d)
{
   Console.WriteLine($"{m} {d}");
}
```

enum

```
public enum Days
    { Sunday, Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, }
    public enum Months
    { January = 1, February, March, April, May, June, July, August, September,
October, November, December }
    static void Main(string[] args)
    {
}
```

enum

```
static void Main(string[] args)
{
    Console.WriteLine("enum example");
    Days day = Days.Sunday;
    Console.WriteLine(day + " = " + (int)day);
    Months month = Months.September;
    Console.WriteLine($"{month} = {(int)month}");
    Console.WriteLine();
}
```

enum

```
public enum Days
   { Sunday, Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, }
   public enum Months
   { January = 1, February, March, April, May, June, July, August, September,
October, November, December }
   static void Main(string[] args)
   {
      Console.WriteLine("enum example");
      Days day = Days.Sunday;
      Console.WriteLine(day + " = " + (int)day);
      Months month = Months.September;
      Console.WriteLine($"{month} = {(int)month}");
      Console.WriteLine();

      // Type Safety
      Fn(month, day); // works as it should
      Fn(day, month); // compiler error because of order of arguments
      Fn(Months.April, Days.Tuesday); // works as it should
      Fn(4, 2);   // compiler error becuase arguments are integers
      Fn((Months)5, (Days)6); // works as it should - typecasting
}

static void Fn(Months m, Days d)
{
   Console.WriteLine($"{m} {d}");
}
```

enum

```
public enum Sizes {None, Small, Medium, Large, Jumbo}

Sizes drink = Sizes.Small;
switch(drink)
{
    case Sizes.None:
        .
        .
        .
        break;
    case Sizes.Small:
        .
        .
        .
        break;
    default:
        .
        .
        .
        break;
}
```