You can find this on my github: <https://github.com/princesscorn/assignments-antra.git>

# 01 Introduction to C# and Data Types

## Understanding Data Types

### Test your Knowledge

#### 1.What type would you choose for the following “numbers”?

A person’s telephone number

=>string

A person’s height

=>float

A person’s age

=>byte

A person’s gender (Male, Female, Prefer Not To Answer)

=>enum

A person’s salary

=>Decimal

A book’s ISBN

=>string

A book’s price

=>Decimal

A book’s shipping weight

=>float

A country’s population

=>uint

The number of stars in the universe

=>string

The number of employees in each of the small or medium businesses in the

United Kingdom (up to about 50,000 employees per business)

=>ushort

#### 2. What are the difference between value type and reference type variables? What is boxing and unboxing?

=>1) value type variable directly holds the value,

reference type variable holds the reference to the data.

2) Each value type has its own copy of data.

Two reference variables can point to the same object.

3) Operation on one cannot affect another one, reference will.

4) Value type cannot accept null values, reference can

5) value type variables not collected by garbage collector, reference can.

6) value type variables stored in stack memory, reference stored in heap.

Boxing: conversion of value to reference

Unboxing: conversion of reference to value

#### 3.What is meant by the terms managed resource and unmanaged resource in .NET

#### 4.Whats the purpose of Garbage Collector in .NET?

### Playing with Console App

Modify your console application to display a different message. Go ahead and

intentionally add some mistakes to your program, so you can see what kinds of error

messages you get from the compiler. The more familiar you are with these messages, and what causes them, the better you'll be at diagnosing problems in your programs that you didn't intend to add!

Using just the ReadLine and WriteLine methods and your current knowledge of variables, you can have the user pass in quite a few bits of information. Using this approach, create a console application that asks the user a few questions and then generates some custom output for them. For instance, your program could generate their "hacker name" by asking them their favorite color, their astrology sign, and their street address number. The result might be something like "Your hacker name is RedGemini480."

### Practice number sizes and ranges

1.Create a console application project named /02UnderstandingTypes/ that outputs the number of bytes in memory that each of the following number types uses, and the minimum and maximum values they can have: sbyte, byte, short, ushort, int, uint, long, ulong, float, double, and decimal.

Composite Formatting to learn how to align text in a console application.

2. Write program to enter an integer number of centuries and convert it to

years, days, hours, minutes, seconds, milliseconds, microseconds, nanoseconds. Use an appropriate data type for every data conversion. Beware of overflows!

Input: 1

Output: 1 centuries = 100 years = 36524 days = 876576 hours = 52594560 minutes = 3155673600 seconds = 3155673600000 milliseconds = 3155673600000000 microseconds = 3155673600000000000 nanoseconds

Input: 5

Output: 5 centuries = 500 years = 182621 days = 4382904 hours = 262974240 minutes = 15778454400 seconds = 15778454400000 milliseconds = 15778454400000000 microseconds = 15778454400000000000 nanoseconds

### Explore following topics

C# Keywords: [C# Keywords | Microsoft Docs](https://docs.microsoft.com/en-us/dotnet/csharp/language-reference/keywords/)

Main() and command-line arguments: [Main() and command-line arguments | Microsoft Docs](https://docs.microsoft.com/en-us/dotnet/csharp/fundamentals/program-structure/main-command-line)

Types (C# Programming Guide): [The C# type system | Microsoft Docs](https://docs.microsoft.com/en-us/dotnet/csharp/fundamentals/types/)

Statements, Expressions, and Operators: [Statements, Expressions, and Operators - C# Programming Guide | Microsoft Docs](https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/statements-expressions-operators/)

Strings (C# Programming Guide): [Strings - C# Programming Guide | Microsoft Docs](https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/strings/)

Nullable Types (C# Programming Guide): [Nullable value types - C# reference | Microsoft Docs](https://docs.microsoft.com/en-us/dotnet/csharp/language-reference/builtin-types/nullable-value-types)

Nullable reference types: [Nullable reference types | Microsoft Docs](https://docs.microsoft.com/en-us/dotnet/csharp/nullable-references)

## Controlling Flow and Converting Types

### Test your Knowledge

#### 1.What happens when you divide an int variable by 0?

=> Compile error: error CS0020: Division by constant zero

#### 2.What happens when you divide a double variable by 0?

=>Correct, no warning, no error.

#### 3.What happens when you overflow an int variable, that is, set it to a value beyond its range?

=>Compile error, like Constant value '93456' cannot be converted to a 'short'

#### 4.What is the difference between x = y++; and x = ++y;?

=>x = y++; means x = y; y = y + 1;

X = ++y; means y = y + 1; x = y;

#### 5.What is the difference between break, continue, and return when used inside a loop statement?

=>break: jump out of the whole loop

Continue: jump out of the current loop, continue to next loop

Return: function returned.

#### 6.What are the three parts of a for statement and which of them are required?

=> initialization

condition

increment/decrement

condition and increment/decrement are required.

#### 7.What is the difference between the = and == operators?

=>”=” means assign

“==” comapre

#### 8.Does the following statement compile? for ( ; true; ) ;

=>Yes

#### 9.What does the underscore \_ represent in a switch expression?

=> The underscore (\_) character replaces the default keyword to signify that it should match anything if reached

#### 10.What interface must an object implement to be enumerated over by using the foreach statement?

=> implemented the IEnumerable

### Practice loops and operators

1.FizzBuzz is a group words game for children to teach them about division. Players take turns to count incrementally, replacing any number divisible by three with the word /fizz/, any number divisible by five with the word /buzz/, and any number divisible by both with /fizzbuzz/.

Create a console application in Chapter03 named Exercise03 that outputs a simulated FizzBuzz game counting up to 100. The output should look something like the following screenshot:

What will happen if this code executes?

int max = 500;

for(byte i = 0; i < max; i++)

{

WriteLine(i);

}

Create a console application and enter the preceding code. Run the console application and view the output. What happens?

What code could you add (don’t change any of the preceding code) to warn us about the problem?

=>Infinite loop.

Byte -> int

2. Print-a-Pyramid. Like the star pattern examples that we saw earlier, create a program that will print the following pattern: If you find yourself getting stuck, try recreating the two examples that we just talked about in this chapter first. They’re simpler, and you can compare your results with the code included above.

This can actually be a pretty challenging problem, so here is a hint to get you going. I used three total loops. One big one contains two smaller loops. The bigger loop goes from line to line. The first of the two inner loops prints the correct number of spaces, while the second inner loop prints out the correct number of stars.

\*

\*\*\*

\*\*\*\*\*

\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*

=>

public static void Main()

{

int i, j, n;

Console.Write("\n\n");

Console.Write("Display the pattern like pyramid containing odd number of asterisks:\n");

Console.Write("----------------------------------------------------------------------");

Console.Write("\n\n");

Console.Write("Input number of rows for this pattern :");

n = Convert.ToInt32(Console.ReadLine());

for (i = 0; i <= n; i++)

{

for (j = 1; j <= n - i; j++)

Console.Write(" ");

for (j = 1; j <= 2 \* i - 1; j++)

Console.Write("\*");

Console.Write("\n");

}

}

3. Your program can create a random number between 1 and 3 with the following code:

int correctNumber = new Random().Next(3)+1;

Write a program that generates a random number between 1 and 3 and asks the user to guess what the number is. Tell the user if they guess low, high, or get the correct answer. Also, tell the user if their answer is outside of the range of numbers that are valid guesses (less than 1 or more than 3). You can convert the user's typed answer from a string to an int using this code:

int guessedNumber = int.Parse(Console.ReadLine());

Note that the above code will crash the program if the user doesn't type an integer value.

For this exercise, assume the user will only enter valid guesses.

=>

public static void Main()

{

int correctNumber = new Random().Next(3) + 1;

Console.WriteLine("Please guess a number: ");

int guessedNumber = int.Parse(Console.ReadLine());

if ((guessedNumber > 3) || (guessedNumber < 1))

{

Console.WriteLine("The range is 1~3.");

}

else if(guessedNumber < correctNumber)

{

Console.WriteLine("Low");

}

else if (guessedNumber > correctNumber)

{

Console.WriteLine("High");

}

else if (guessedNumber == correctNumber)

{

Console.WriteLine("Correct");

}

}

4.Write a simple program that defines a variable representing a birth date and calculates how many days old the person with that birth date is currently.

For extra credit, output the date of their next 10,000 day (about 27 years) anniversary.

Note: once you figure out their age in days, you can calculate the days until the next anniversary using

int daysToNextAnniversary = 10000 - (days % 10000);

=>

public static void Main()

{

DateTime bday = new DateTime(1999, 09, 09);

DateTime now = DateTime.Today;

int ageDays = (now - bday).Days;

Console.WriteLine("Age by Days:" + ageDays);

int daysToNextAnniversary = 10000 - (ageDays % 10000);

Console.WriteLine("Days to Next Anniversay is: " + daysToNextAnniversary);

}

5.Write a program that greets the user using the appropriate greeting for the time of day. Use only if, not else or switch, statements to do so. Be sure to include the following greetings:

"Good Morning"

"Good Afternoon"

"Good Evening"

"Good Night"

It's up to you which times should serve as the starting and ending ranges for each of the greetings. If you need a refresher on how to get the current time, see DateTime Formatting. When testing your program, you'll probably want to use a DateTime variable you define, rather than the current time. Once you're confident the program works correctly, you can substitute DateTime.Now for your variable (or keep your variable and just assign DateTime.Now as its value, which is often a better approach).

=>

public static void Main()

{

DateTime currentDateTime = DateTime.Now;

int currentHour = currentDateTime.Hour;

int startMorningHour = 6;

int startAfternoonHour = 12;

int startEveningHour = 17;

int startNightHour = 22;

if (startMorningHour <= currentHour && currentHour < startAfternoonHour)

{

Console.WriteLine("Good morning!");

}

if (startAfternoonHour <= currentHour && currentHour < startEveningHour)

{

Console.WriteLine("Good Afternoon!");

}

if (startEveningHour <= currentHour && currentHour < startNightHour)

{

Console.WriteLine("Good Evening!");

}

if (startNightHour <= currentHour || currentHour < startMorningHour)

{

Console.WriteLine("Good Night!");

}

Console.WriteLine("Right now it is {0}:{1} o'clock.", currentDateTime.Hour, currentDateTime.Minute);

}

}

6.Write a program that prints the result of counting up to 24 using four different increments.

First, count by 1s, then by 2s, by 3s, and finally by 4s.

Use nested for loops with your outer loop counting from 1 to 4. Your inner loop should count from 0 to 24, but increase the value of its /loop control variable/ by the value of the /loop control variable/ from the outer loop. This means the incrementing in the /afterthought/ expression will be based on a variable.

Your output should look something like this:

0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24

0,2,4,6,8,10,12,14,16,18,20,22,24

0,3,6,9,12,15,18,21,24

0,4,8,12,16,20,24

=>

public static void Main()

{

for (int i = 1; i <= 4; ++i)

{

int j = 0;

for (; j < 24; )

{

int tmp = j;

Console.Write(tmp + ", ");

j = j + i;

}

Console.WriteLine(j);

}

}

### Explore following topics

C# operators: [C# operators and expressions - C# reference | Microsoft Docs](https://docs.microsoft.com/en-us/dotnet/csharp/language-reference/operators/)

Bitwise and shift operators: [Bitwise and shift operators - C# reference | Microsoft Docs](https://docs.microsoft.com/en-us/dotnet/csharp/language-reference/operators/bitwise-and-shift-operators)

Statement keywords: [Statement keywords - C# Reference | Microsoft Docs](https://docs.microsoft.com/en-us/dotnet/csharp/language-reference/keywords/statement-keywords)

Casting and type conversions: [Casting and type conversions - C# Programming Guide | Microsoft Docs](https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/types/casting-and-type-conversions)

Fundamentals of garbage collection: [Fundamentals of garbage collection | Microsoft Docs](https://docs.microsoft.com/en-us/dotnet/standard/garbage-collection/fundamentals)

$ - string interpolation: [$ - string interpolation - C# reference | Microsoft Docs](https://docs.microsoft.com/en-us/dotnet/csharp/language-reference/tokens/interpolated)

Formatting types in .NET: [Overview: How to format numbers, dates, enums, and other types in .NET | Microsoft Docs](https://docs.microsoft.com/en-us/dotnet/standard/base-types/formatting-types)

Iteration statements: [Iteration statements - C# reference | Microsoft Docs](https://docs.microsoft.com/en-us/dotnet/csharp/language-reference/statements/iteration-statements)

Selection statements: [Selection statements - C# reference | Microsoft Docs](https://docs.microsoft.com/en-us/dotnet/csharp/language-reference/statements/selection-statements)

# 02 Arrays and Strings

## Test your Knowledge

### 1.When to use String vs. StringBuilder in C# ?

=>

If a string is going to remain constant throughout the program, then use String class object because a String object is immutable.

If a string can change (example: lots of logic and operations in the construction of the string) then using a StringBuilder is the best option.

### 2.What is the base class for all arrays in C#?

=> The Array class

### 3.How do you sort an array in C#?

=>Array.sort();

### 4.What property of an array object can be used to get the total number of elements in an array?

=> .Count

.Length

### 5.Can you store multiple data types in System.Array?

=> No, we cannot. We can do this with ArrayList.

### 6.What’s the difference between the System.Array.CopyTo() and System.Array.Clone()?

=>

The Clone() method returns a new array (a shallow copy) object containing all the elements in the original array.

The CopyTo() method copies the elements into another existing array. Both perform a shallow copy.

A shallow copy means the contents (each array element) contains references to the same object as the elements in the original array. A deep copy (which neither of these methods performs) would create a new instance of each element's object, resulting in a different, yet identical object.

## Practice Arrays

1.Copying an Array

Write code to create a copy of an array. First, start by creating an initial array. (You can use whatever type of data you want.) Let’s start with 10 items. Declare an array variable and assign it a new array with 10 items in it. Use the things we’ve discussed to put some values in the array.

Now create a second array variable. Give it a new array with the same length as the first. Instead of using a number for this length, use the Length property to get the size of the original array.

Use a loop to read values from the original array and place them in the new array. Also print out the contents of both arrays, to be sure everything copied correctly.

=>

public static void Main()

{

int[] abc = { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 };

int number = abc.Length;

int[] def = new int[number];

System.Array.Copy(abc, def, number);

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

{

Console.Write(abc[i] + " ");

def[i] = abc[i];

}

Console.WriteLine();

number = def.Length;

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

{

Console.Write(def[i] + " ");

}

Console.WriteLine();

}

2.Write a simple program that lets the user manage a list of elements. It can be a grocery list, "to do" list, etc. Refer to Looping Based on a Logical Expression ([Branches and loops - Introduction to C# tutorial | Microsoft Docs](https://docs.microsoft.com/en-us/dotnet/csharp/tour-of-csharp/tutorials/branches-and-loops-local)) if necessary to see how to implement an infinite loop. Each time through the loop, ask the user to perform an operation, and then show the current contents of their list. The operations available should be Add, Remove, and Clear. The syntax should be as follows:

+ some item

- some item

--

Your program should read in the user's input and determine if it begins with a “+” or “-“ or if it is simply “—“ . In the first two cases, your program should add or remove the string given ("some item" in the example). If the user enters just “—“ then the program should clear the current list. Your program can start each iteration through its loop with the following instruction:

Console.WriteLine("Enter command (+ item, - item, or -- to clear)):");

3.Write a method that calculates all prime numbers in given range and returns them as array of integers

Static int[]FindPrimesInRange(startNum, endNum)

{

}

=>

int[] prime = new int[endNum - startNum + 1];

int totalNum = 0;

bool isPrime = true;

if ((startNum == 2) || (endNum == 2))

{

prime[totalNum++] = 2;

}

for (int i = startNum; i <= endNum; ++i )

{

isPrime = true;

for (int j = 2; j \* j <= i; ++j)

{

if (i % j == 0)

{

isPrime = false;

break;

}

}

if (isPrime)

{

prime[totalNum++] = i;

}

}

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

{

Console.Write(prime[i] + " ");

}

4.Write a program to read an array of n integers (space separated on a single line) and an integer k, rotate the array right k times and sum the obtained arrays after each rotation as shown below.

After r rotations the element at position I goes to position (I + r) % n.

The sum[] array can be calculated by two nested loops: for r = 1 ... k;

for I = 0 ... n-1.

Input Output Comments

3 2 4 -1 3 2 5 6 rotated1[] = -1 3 2 4

2 rotated2[] = 4 -1 3 2

sum[] = 3 2 5 6

1 2 3 4 5 12 10 8 6 9 rotated1[] = 5 1 2 3 4

3 rotated2[] = 4 5 1 2 3

rotated3[] = 3 4 5 1 2

sum[] = 12 10 8 6 9

5.Write a program that finds the longest sequence of equal elements

in an array of integers. If several longest sequences exist, print the leftmost one.

Input Output

2 1 1 2 3 3 2 2 2 1 2 2 2

1 1 1 2 3 1 3 3 1 1 1

4 4 4 4 4 4 4 4

0 1 1 5 2 2 6 3 3 1 1

7.Write a program that finds the most frequent number in a given sequence of numbers. In case of multiple numbers with the same maximal frequency, print the left most of them.

Text

Description automatically generated

## Practice Strings

1.Write a program that reads a string from the console, reverses its letters and prints the result back at the console.

Write in two ways:

Convert the string to char array, reverse it, then convert it to string again

Print the letters of the string in back direction (from the last to the first) in a

for-loop

Background pattern

Description automatically generated with low confidence

2.Write a program that reverses the words in a given sentence without changing the punctuation and spaces

Use the following separators between the words: . , : ; = ( ) & [ ] " ' \ / ! ? (space).

All other characters are considered part of words, e.g. C++, a+b, and a77 are considered valid words.

The sentences always start by word and end by separator.

Graphical user interface, text, application

Description automatically generated

3.Write a program that extracts from a given text all palindromes, e.g. “ABBA”, “lamal”, “exe” and prints them on the console on a single line, separated by comma and space.Print all unique palindromes (no duplicates), sorted

Graphical user interface, text, application

Description automatically generated

4.Write a program that parses an URL given in the following format:

[protocol]://[server]/[resource]

The parsing extracts its parts: protocol, server and resource.

The [server] part is mandatory.

The [protocol] and [resource] parts are optional.

https://www.apple.com/iphone

[protocol] = "https"

[server] = "www.apple.com"

[resource] = "iphone"

ftp://www.example.com/employee

[protocol] = "ftp"

[server] = "www.example.com"

[resource] = "employee"

https://google.com

[protocol] = "https"

[server] = "google.com"

[resource] = ""

www.apple.com

[protocol] = ""

[server] = "www.apple.com"

[resource] = ""

## Explore the following Topics

Strings: [Strings - C# Programming Guide | Microsoft Docs](https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/strings/)

Arrays: [Arrays - C# Programming Guide | Microsoft Docs](https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/arrays/)

Using the StringBuilder: [Using the StringBuilder Class in .NET | Microsoft Docs](https://docs.microsoft.com/en-us/dotnet/standard/base-types/stringbuilder)