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): string  
A person’s salary: int  
A book’s ISBN: string  
A book’s price: float  
A book’s shipping weight: float  
A country’s population: long  
The number of stars in the universe: long  
The number of employees in each of the small or medium businesses in the  
United Kingdom (up to about 50,000 employees per business): int

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

Value type is the actual value that is stored inside of the variable and reference type points to the memory allocation of the value.

Boxing is when you can use the different types as objects and unboxing when you use them as their actual type

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

A managed resource only uses the .NET runtime and resources while unmanaged can use other tracefiles and databases and not have the same runtime

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

It gets rid of unused memory allocation to free up space

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

Text

Description automatically generated  
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!  
Timeline

Description automatically generated with medium confidence

Explore following topics  
C# Keywords  
Main() and command-line arguments  
Types (C# Programming Guide)  
Statements, Expressions, and Operators  
Strings (C# Programming Guide)  
Nullable Types (C# Programming Guide)  
Nullable reference types

Controlling Flow and Converting Types

Test your Knowledge

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

It throws a DivideByZero Exception

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

It returns infinity

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

It returns the low-order bits of the mathematical sum in some two-component’s format

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

Y++ executes the statement and then increments the value. ++y increments the value and then executes the statement.

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

Break terminates the loop and stop further iterations.

Continue will skip the current loops’ iteration

Return will exit the loop and the function, returning the specified value

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

Statement 1: executed one time before execution of code block

Statement 2: defines the condition for executing the code block

Statement 3: executed every time after the code block has been executed

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

=: makes a the variable on the left equal to the variable on the left

==: returns true if the two operands of the same enum type are equal

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

Yes

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

It replaces the default keyword, signifying that it should match snything if reached. Bodies are now expressions instead of statements.

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

IEnumerator

Practice loops and operators  
1. FizzBuzzis a group word 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);  
}

It will not execute because you have to say Console.WriteLine(i); for the program to work. If this is the case, then the program will print the numbers in the range 0 to 500.

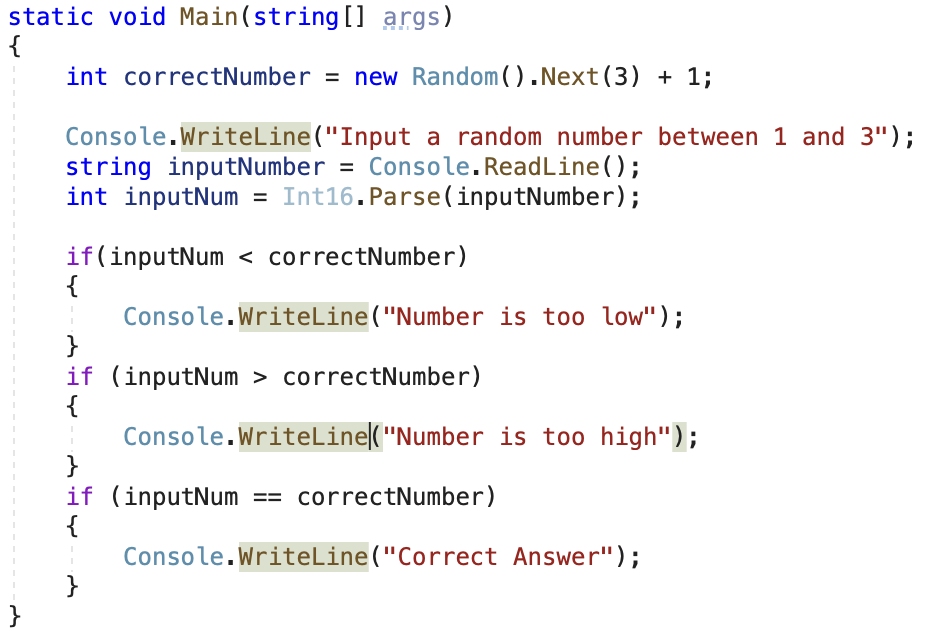
- Create a console application and enter the preceding code. Run the console application  
and view the output. What happens?

There is an error.

- What code could you add (don’t change any of the preceding code) to warn us about the  
problem? Console.WriteLine(i);

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.



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.  
\*  
\*\*\*  
\*\*\*\*\*  
\*\*\*\*\*\*\*  
\*\*\*\*\*\*\*\*\*

static void Main(string[] args)

{

Console.WriteLine(" \* ");

Console.WriteLine(" \*\*\* ");

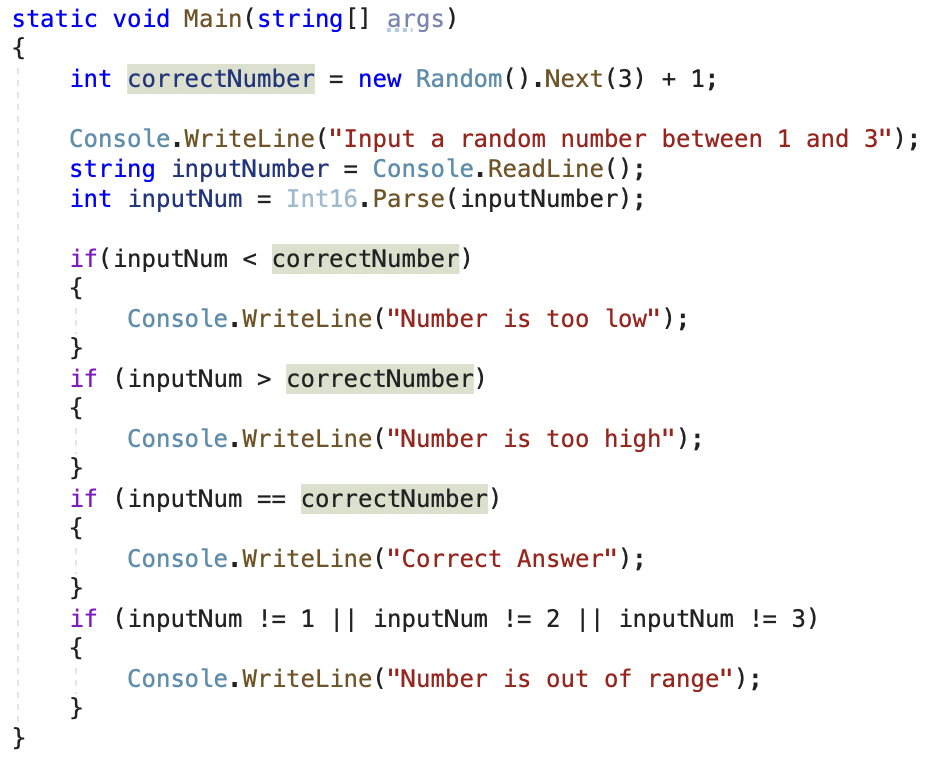
Console.WriteLine(" \*\*\*\*\* ");

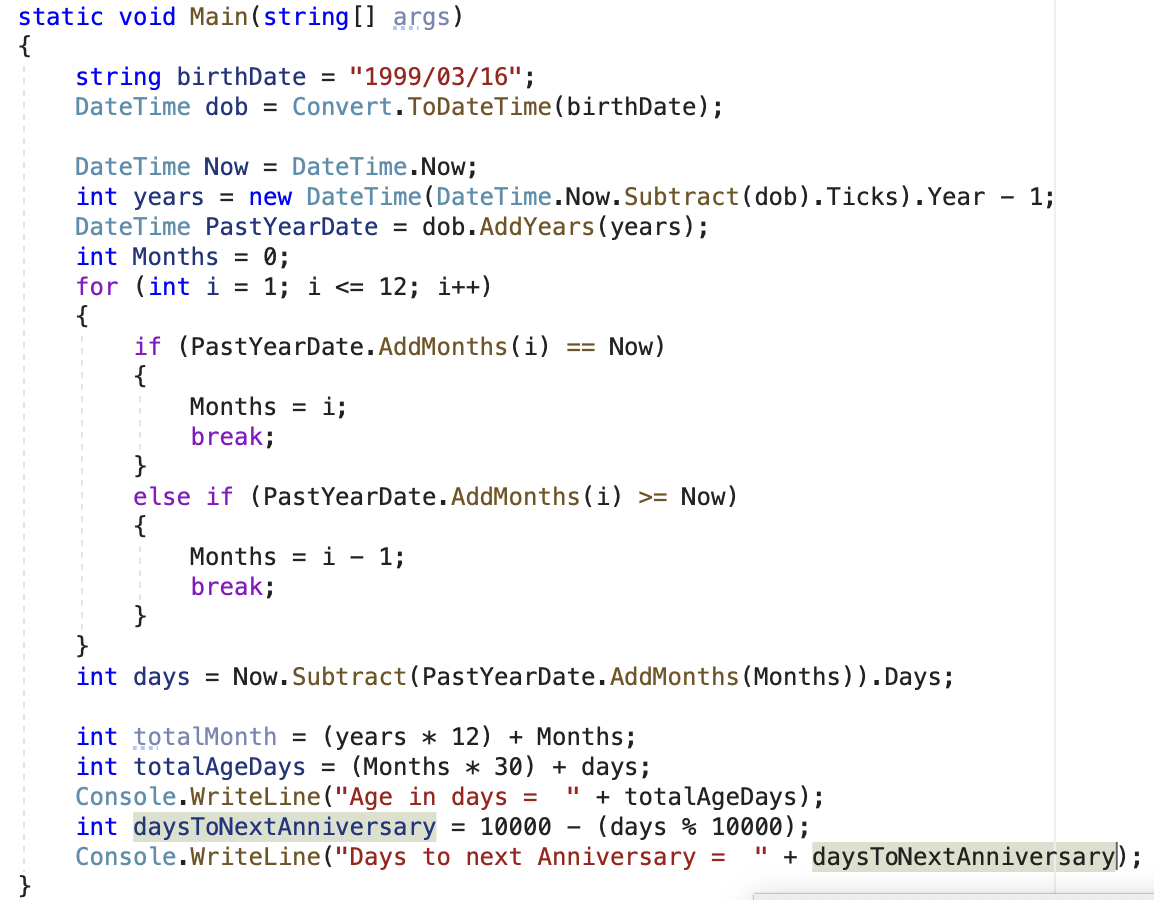
Console.WriteLine(" \*\*\*\*\*\*\* ");

Console.WriteLine("\*\*\*\*\*\*\*\*\*");

}

3. 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.



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); .

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).Graphical user interface, text, application, email

Description automatically generated

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. You 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

Graphical user interface, text, application, email

Description automatically generated