Assignment Questions

01. Introduction to C# and Data Types

Textual Questions

1. What type would you choose for the following "numbers"?
    - A person's telephone number - `string`
    - A person's height -`double`
    - A person's age - `int`
    - A person's gender (Male, Female, Prefer Not To Answer) -`string`
    - A person's salary - `decimal`
    - A book's ISBN - `string`
    - A book's price - `decimal`
    - A book's shipping weight - `double`
    - 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) `long`
2. What is the difference between value type and reference type variables? What is boxing and unboxing?

```
Value Types are stored on Stack while Reference type variable are
stored on Heap.

Value Types store the actual values, while reference type stores the
reference to the data.

Value types, when copied, create an independ copy. Changes made to one
copy don't affect the other.
Reference type, when copied, copy the reference, which is still
pointing at the same object,
changes made affect all references.

Default value for the value type is the default for tis type,
example:0 is default for int type
Reference types are intialzied to null

ValueTYpes: Float, int,bool char [User Types:Struct, Enum)
Reference Types: Class, Arrays, Strings.

Boxing and Unboxing is the process of converting:
1: a value type into reference type(boxing it into a object), storing
it in heap
2: A reference type into a Value type (Typecasting the value type on
an object[unboxing])
```

```
Boxing is implicit, that is no manual casting required when boxing a
value into object

Unboxing is explicit, developer has to manually typecast the object
into a vaiable,
and needs to make sure correct type is cast lest we get compile error
```

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

```
In .Net, managed resource are resources that are fully managed by .NET
runtime (CLR - Common Language Runtime)
Managed Resources are automatically allocated and deallocated BY
Garbage Collector, which ensures that memory is handled
efficiently.

EXAMPLES:
Objects such as Strings, Lists, Arrays and other managed types,
Objects on Heap
Primitive Data Types such as  int float.

UnManaged Resources are resources not controlled by .NET runtime,
these are OS level resources that GC doesn't track.
Examples:
File Handles,
Database Connection
Network Sockets etc.

Manual cleanup is required, developers need to release these resouces
to avoid memory leaks.

.NET used IDisposable interface, Safe Handle classes and other ways to
handle unmanaged resources.
```

4. What is the purpose of the Garbage Collector in .NET?

   GC provies automatic memory management so that memory use is optimized.

   GC utilizes memory management techniques such as Mark and Sweep and Generations to keep track of memory usage and cleanup.

Coding Questions

Playing with Console App

1. Modify your console application to display a different message. Add mistakes intentionally to see the error messages provided by the compiler.

2. Create a console application that asks the user a few questions and generates custom output, such as a "hacker name" based on user input.

Practice Number Sizes and Ranges

1. Create a console application project (`02UnderstandingTypes`) that outputs the number of bytes each number type uses and their minimum and maximum values:
   - Types: `sbyte`, `byte`, `short`, `ushort`, `int`, `uint`, `long`, `ulong`, `float`, `double`, `decimal`.

Composite Formatting

2. Write a program to convert centuries into years, days, hours, minutes, seconds, milliseconds, microseconds, and nanoseconds.
   - Input: `1`
   - Output: `1 century = 100 years = 36524 days = 876576 hours = 52594560 minutes = ...`

---

02. Controlling Flow and Converting Types

Textual Questions

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

```
  Compilation Error: System.DivideByZeroException: Attempted to divide
by zero.
```

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

```
Compiles succesfully :
If printed, prints Infinity, if we attempt to substract any number
from it, still is infinity.
If we attempt to substract itself from itself, will compile but return
NaN(Not a Number)
```

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

```
It overflows and sets to the min range of itself, a int overflowed by
+1 will set to -2147483648
```

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

```
++x and x++ are pre-increment and post-increment operators
respectively
```

```
y= ++x -> x is incremented first and then assigned to y
y = x++ -> x is assigned to y first, and then incremented
```

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

> break will exit out of the loop abd transfer control to the statment following the loop
>
> continue will skip the remaining code in current iteration and move on to next iteration
>
> return will exit out of the loop, and enclosing method and return control to caller of method

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

> Three parts of for loop statements are:
>
> **Initialization** This is where the control variable value is set at beginning of loop
>
> **Condition** This part is a boolean expression that is evaluated at each start of iteration, if condition true, continue loop, else, terminate.
>
> **Updation** This part is executed at the end of each iteration and is used to update the control variable
>
> **All three parts are optional, but the semicolons sepearing the parts is required, without any parts and just semicolons, we will create an infinite loop**

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

> '=' is a assignment operator, it assigns the value on the right side to the variable on the left side.
>
> '==' is a equality operator, it cheks if values on both sides are equal and returns a boolean result i.e true or false.

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

> Yes, it starts a infinite loop, need to escape it from terminal with a ctrl + c.

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

> It represents the default case.

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

> IEnumerable interface must be implemented by the object.

Coding Questions

Practice Loops and Operators

1. **FizzBuzz Simulation:**
   Create a console application (`Exercise03`) that simulates a FizzBuzz game up to 100.

2. **Handling Overflow:**
   Use the code:

```
int max = 500;
for (byte i = 0; i < max; i++)
{
    WriteLine(i);
}
```

Analyze the output and add code to warn about the problem without changing the existing code.

> Without changing anything the code won't even compile as WriteLine isn't a method, it has to be used with Console dot notation.
>
> Assuming it's a typo in question, running it with Console.WriteLine() will result in an infinite loop, as the byte is an unsigned numeric data type which holds max value till 255.
>
> It resets back to 0 and keeps going, infinite loop until keyboard interrupt is done.
>
> Change to code below

```
static void Main()
{
int max = 500;

     Console.WriteLine("Warning: The variable 'i' is byte, itss max is
255, itwill overflow. Do you want to continue? Y or N");
     string choice = Console.ReadLine();
     if (choice == "Y")
     {


     for (byte i = 0; i < max; i++)
     {
         Console.WriteLine(i);
         if(i == byte.MaxValue) {
         }


     }
     }

}
```

3. Guessing Game: Write a program that generates a random number between 1 and 3 and allows the user to guess it. Provide hints if the guess is high, low, or correct.

4. Print-a-Pyramid: Create a program to print the following pattern:

```
    *
   ***
  *****
 *******
*********
```

5. **Age in Days and Milestones:**
   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); •

6. **Time-Based Greetings:**
   Write a program to greet the user with appropriate messages based on the time of day:

- "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 Date Time 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).

7. **Counting Increments:**
   Create a program using nested loops to count up to 24 with increments of 1, 2, 3, and 4.
   Example Output:

```
0, 1, 2, 3, ..., 24
0, 2, 4, 6, ..., 24
0, 3, 6, 9, ..., 24
0, 4, 8, 12, ..., 24
```

---

03. Explore Topics

Topics to Research

- 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
- C# Operators
- Bitwise and Shift Operators
- Statement Keywords
- Casting and Type Conversions
- Fundamentals of Garbage Collection
- `$` - String Interpolation
- Formatting Types in .NET
- Iteration Statements