**Assignment 1 C#**

Enze Zhang

**Understanding Data Types**

*Knowledge*

1.

A person’s telephone number - ulong

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 - decimal

A book’s ISBN - ulong

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.

Value Types and Reference Types:

1. Value type will directly hold the value, while reference type will hold the memory address or reference for its value
2. Value types will be stored in stack memory while reference type will be stored in heap memory
3. Value type will not be collected by garbage collector, while reference type will be collected by garbage collector
4. Value type can be created by Struct or Enum, while reference type can be created by classes, interfaces, delegates, array
5. Value type cannot accept null values, but reference types can accept null values

boxing: convert a value type into a reference type

unboxing: convert a reference type back to value type

3.

Managed resource is under the control of the garbage collector, but unmanaged resource is not under the control of the garbage collector.

4.

The garbage collector manages the allocation and release of memory for an application. For developers working with managed code, this means that you don't have to write code to perform memory management tasks. Automatic memory management can eliminate common problems, such as forgetting to free an object and causing a memory leak or attempting to access memory for an object that's already been freed.

*Playing with Console App*

using System;

namespace Antra\_Assignment\_1

{

class Program

{

static void Main(string[] args)

{

string clr;

string month;

string num;

Console.WriteLine("Hello, this is a username generator. Please enter your favorite color:");

clr = Console.ReadLine();

Console.WriteLine("Great! Please enter your favorite month:");

month = Console.ReadLine();

Console.WriteLine("Finally, please enter your favorite number:");

num = Console.ReadLine();

Console.WriteLine("Your username is: " + clr + month + num );

}

}

}

*Practice number sizes and ranges*

1.

using System;

namespace \_02UnderstandingTypes

{

class Program

{

static void Main(string[] args)

{

string [] Types = { "sbyte", "byte", "short", "ushort", "int", "uint", "long", "ulong", "float", "double", "decimal"};

int[] MemUse = { 1, 1, 2, 2, 4, 4, 8, 8, 4, 8, 16};

string [] MinVal = { "-128", "0", "-32,768", "0", "-2,147,483,648", "0", "-9,223,372,036,854,775,808", "0", "-3.4e38", "-1.7e308", "-7.9e28" };

string[] MaxVal = { "127", "255", "32,767", "65,535", "2,147,483,647", "4,294,967,295", "9,223,372,036,854,775,807", "18,446,744,073,709,551,615", "3.4e38", "1.7e308", "7.9e28" };

Console.WriteLine("{0,-10} {1,30} {2,30} {3,30} \n", "Types", "Memory Usage by Byte", "Max Values", "Min Values");

for(int i = 0; i< Types.Length; i++)

{

Console.WriteLine("{0,-10} {1,30} {2,30} {3,30} \n", Types[i], MemUse[i], MaxVal[i], MinVal[i]);

}

}

}

}

2.

using System;

namespace CenturiesConversion

{

class Program

{

static void Main(string[] args)

{

Console.WriteLine("Please input the number of centuries:");

long n = Convert.ToInt64(Console.ReadLine());

long d = (int)Math.Round(n \* 100 \* 365.242199);

long s = d \* 24 \* 60 \* 60;

Console.Write("{0} centuries = {1} years = {2} days = {3} hours = {4} minutes = {5} seconds", n, n \* 100, d, d \* 24, d \* 24 \* 60, s);

Console.Write(" = {0}000 milliseconds = {1}000000 microseconds = {2}000000000 nanoseconds",s,s,s);

}

}

}

**Controlling Flow and Converting Types**

*Knowledge*

1. There will be a System.DivideByZeroException.
2. The answer will be ∞.
3. The overflowed part will start from the minimum value of int and go up. The same process will repeat if it overflows again.
4. x = y++ will set the value of x as y first, then increase y by 1. However, x = ++y will increase y by 1 first, then set the value of x as the new y.
5. Break will go out of the loop directly. Continue will begin the next iteration directly. Return will return without going to the remaining lines.
6. The three parts of a for statement is initializer, condition, and iterator. The condition is required.
7. = set the value of a valuable. == compare the values of two valuables to see if they are equal.
8. Yes. The loop will run forever.
9. The underscore character replaces the default keyword to signify that it should match anything if reached.
10. IEnumerator interface. It provides iteration over a collection-type object in a class. The IEnumerable interface permits enumeration by using a foreach loop.

*Practice number sizes and ranges*

1.

using System;

namespace FizzBuzz

{

class FizzBuzz

{

static void Main(string[] args)

{

for (int i = 1; i<=100; i++)

{

if(i % 3 == 0)

{

if(i % 5 == 0)

{

Console.WriteLine("fizzbuzz");

}

else

{

Console.WriteLine("fizz");

}

}

else if(i % 5 == 0)

{

Console.WriteLine("buzz");

}

else

{

Console.WriteLine(i);

}

}

}

}

}

The code will run forever because i has a type of byte, which is too small and can cause overflow.

The code does run forever.

We can add a comment saying that the byte type can overflow, and the code can run forever if the max is 500.

using System;

namespace GuessNum1to3

{

class GuessNum1to3

{

static void Main(string[] args)

{

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

Console.WriteLine("Please guess an integer from 1 to 3:");

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

while (true)

{

if (guessedNumber > 3 || guessedNumber < 1)

{

Console.WriteLine("Out of range. Try again:");

}

else if (guessedNumber < correctNumber)

{

Console.WriteLine("Too low. Please guess again:");

}

else if (guessedNumber == correctNumber)

{

Console.WriteLine("Bingo!");

break;

}

else if (guessedNumber > correctNumber)

{

Console.WriteLine("Too high. Please guess again:");

}

else

{

Console.WriteLine("Invalid input. Try again:");

}

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

}

}

}

}

2.

using System;

namespace PrintAPyramid

{

class PrintAPyramid

{

static void Main(string[] args)

{

Console.WriteLine("Please enter the height of the pyramid:");

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

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

{

for(int j = h - i; j>0; j--)

{

Console.Write(" ");

}

for(int k = 1; k < (2\*i); k++)

{

Console.Write("\*");

}

Console.WriteLine("");

}

}

}

}

3.

using System;

namespace GuessNum1to3

{

class GuessNum1to3

{

static void Main(string[] args)

{

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

Console.WriteLine("Please guess an integer from 1 to 3:");

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

while (true)

{

if (guessedNumber > 3 || guessedNumber < 1)

{

Console.WriteLine("Out of range. Try again:");

}

else if (guessedNumber < correctNumber)

{

Console.WriteLine("Too low. Please guess again:");

}

else if (guessedNumber == correctNumber)

{

Console.WriteLine("Bingo!");

break;

}

else if (guessedNumber > correctNumber)

{

Console.WriteLine("Too high. Please guess again:");

}

else

{

Console.WriteLine("Invalid input. Try again:");

}

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

}

}

}

}

4.

using System;

namespace BrithdateCalc

{

class BrithdateCalc

{

static void Main(string[] args)

{

Console.WriteLine("Please enter your birth year:");

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

Console.WriteLine("Please enter your birth month:");

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

Console.WriteLine("Please enter your birth date of the month:");

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

var Bday = new DateTime(y, m, d);

var Today = DateTime.Now;

var DayAge = Today - Bday;

Console.WriteLine("You are {0} days old.",DayAge.Days);

int daysToNextAnniversary = 10000 - (DayAge.Days % 10000);

var NewDay = Today.AddDays(daysToNextAnniversary);

Console.WriteLine("The next 10000 day anniversary will be: {0}",NewDay.ToString("MM/dd/yyyy"));

}

}

}

5.

using System;

namespace Greeting

{

class Greeting

{

static void Main(string[] args)

{

var CurTime = DateTime.Now.TimeOfDay;

switch (CurTime.Hours)

{

case < 4:

Console.WriteLine("Good Night");

break;

case < 12:

Console.WriteLine("Good Morning");

break;

case < 18:

Console.WriteLine("Good Afternoon");

break;

case < 22:

Console.WriteLine("Good Evening");

break;

case < 24:

Console.WriteLine("Good Night");

break;

}

}

}

}

6.

using System;

namespace Counting24

{

class Program

{

static void Main(string[] args)

{

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

{

for(int j = 0; j <= 23; j += i)

{

Console.Write("{0}, ",j);

}

Console.WriteLine("24");

}

}

}

}