Student Name: Andrea Chang

IT Foundation using C# EXAM 1 Instructor: Vallejo

Note: The test is worth **100 points**. **Show all your work** for each problem. No partial credit will be given if no work is shown for each answer. Read the entire description to each question before answering the question. **Good Luck!**

True / False (2 points each)

Circle One

1.	A block (**{ }**)can contain more than one statement.	[✓ TRUE] / FALSE
2.	Every program must have a function called Main.	[✓ TRUE] / FALSE
3.	The type **int** is signed.	[✓ TRUE] / FALSE
4.	Multi-line comments are started by **//**.	TRUE / [✓ FALSE]
5.	Variables are used only for storing constants.	TRUE / [✓ FALSE]
6.	All statements are terminated by a comma.	TRUE / [✓ FALSE]
7.	A variable name may begin with an underscore (_).	[✓ TRUE] / FALSE
8.	**\n** is used by **WriteLine** to go to the next new line.	[✓ TRUE] / FALSE
9.	Upper- and lower-case letters are significant for names.	[✓ TRUE] / FALSE
10.	The type **char** is Unicode (2 bytes).	[✓ TRUE] / FALSE

Multiple Choice (3 points each)

11. Which feature will execute a block of code at least once:
    - A. **while**
    - B. **for**
    - C. **do-while**
    - [✓ D. **foreach**]

12. An **int** variable occupies:
    - A. One byte
    - B. 9 bits (1 for parity)
    - [✓ C. Four bytes]
    - D. 7 bits (for unsigned)

13. What function is used to read in a string:
    - A. Console.WriteLine
    - B. Console.Read
    - C. Console.Write
    - [✓ D. Console.ReadLine]

14. Which is an INVALID statement:

A. `x = x / -1;`
B. `y = y + 2`
C. `z = z + z;`
[✓ D. `t += t;`]

15. What does X == Y mean?

A. X is assigned to Y
B. Y is assigned to X
[✓ C. X is compared to Y]
D. None of the above

16. Which of the following is NOT a logical operator:

[✓ A. ;]
B. ||
C. &&
D. !

17. Which is an invalid type of Field:

A. readonly
[✓ B. get]
C. const
D. None of the above

18. **break** is used to:

A. Exit a program
B. Exit stage right
C. Exit a function
[✓ D. Exit a loop]

19. C# ignores:

[✓ A. Whitespace]
B. Braces
C. Commas
D. Semicolons

20. What is the significance of **while (true)**

A. It is an invalid expression
B. It is an infinite trip
C. It is an infinite statement
[✓ D. It will never stop]

21. What is the result of the following statement for x = 4? (5 points) 8

```
result = X + X++;
result = --X + X;
```

```
    class Program
    {
        static void Main()
        {

            int  x= 4;
            int result = x + x++; // the result is 4 + 4 which is 8, then x is incremented up to 5
            result = --x + x; // x is decreased back down by one to 4. 4 + 4 = 8.

            {

                Console.WriteLine("{0}", result);
                Console.ReadLine();


            }
        }
    }
}
```

```
The output (the result) was 8.
```

22. What are the basic arithmetic operations? Show the operational signs. (5 points)

+	Add numbers
-	Subtract numbers
*	Multiply numbers
/	Divide numbers
%	Divide numbers and return remainder

23. What are the basic conditional operations (less than, equality, etc.)?

==	is equal to/comparing
!=	is not equal to
>	is greater than
>=	is greater than or equal to
<	is less than
<=	is less than or equal to

What are the basic logical operations? (5 points)

||	means "or"
&&	means "and"
!	means "not"

24. Check if the following *if* expressions below result in TRUE or FALSE? (5 points)

A.

```
usCnt = 10; usSum = 10;
```

B.

```
usCnt = 10; usSum = 10;
```

```
if (usSum++ == usCnt)          if (usSum == ++usCnt)
   {                           {
   etc...                      etc...
   }                           }
```

[✓ TRUE] FALSE TRUE [✓ FALSE]

A.

```
 class Program
    {
        static void Main()
        {

            int usCnt = 10;
            int usSum = 10;

            if (usSum++ == usCnt)
            {

                Console.WriteLine("This works.");

                Console.ReadLine();


            }
        }
    }
}
This outputs “This works.”
```

B.

```
class Program
    {
        static void Main()
        {

            int usCnt = 10;
            int usSum = 10;

            if (usSum== ++ usCnt)
            {

                Console.WriteLine("This works.");

                Console.ReadLine();


            }
        }
    }
}
This does not output anything.
```

25. What is the value of usSum after the following code segment? (5 points)

```
usSum = 10; usCnt = 2;
switch (usCnt)
   {
   case 3:
      {
      usSum = usSum + 2;
      break;
      }
   default:
      {
      break;
      }
   case 2:
      {
      usSum = usSum * 3;
      goto case 3;
      }
   }
```

usSum = __32___

```
class Program
    {
        static void Main()
        {

            int usSum = 10;
            int usCnt = 2;

            switch (usCnt)
            {
                case 3:
                    {
                        usSum = usSum + 2;

                        break;
                    }
                default:
                    {
                        break;
                    }
                case 2:
                    {
                        usSum = usSum * 3;
                        goto case 3;
                    }


            }

        }
    }
      }
```

Pinning “usSum” while hitting F10 to this code, 32 is the final value of it.

26. ***Circle*** the COMPILER/SYNTAX errors in this program (5 points)

```
static void Main( }   // should be )
{
int   iSum , ;   // no comma needed

int   iCnt = Sum, iValue; iTotal;
                // The names "Sum" and "iValue" do not exist in this current context
                and a comma is used incorrectly after "Sum." The variable "iValue" is
                declared but never used.
char  chChar = "a";
                //Cannot implicitly convert type "string" to "char."
iSum = chChar
;
while (iSum = 100);
                //Cannot implicitly convert type "int" to "bool."
   {
   iSum = iSum + 1;
   }
)  // this should }
```

27. Given the following program what will the last value of usCnt be when the program completes execution? Is there anything unusual about this program? What does it show and what do you conclude from this? (10 points)

```
using System;
class Test
{
   static void Main( )
   {
   uint  usCnt;
   uint  usSum = 0;

   for (usCnt = 10; usCnt >= 0; usCnt--)
      {
      Console.WriteLine("{0}",usCnt);
      usSum = usSum + usCnt;
      }
   }
}
```

After "usCnt" hit zero, when we decreased it by --, the value turned into 4294967295. The program doesn't ever finish executing because the value of usCnt will always be greater than or equal to zero. This is because it is of type uint, which is unsigned and goes from a value of 0 to 4.2 billion. That is why after 0, it doesn't turn into a negative number and will cycle back to 4.2 billion.

28. What will the following program display? (10 points)

```
using System;
class Test
{
   static void Main( )
   {
   int   iX;
   int   iY;

   iX = 15321;
   while (iX != 0)
      {
      iY = iX % 10;
      Console.Write(iY);
      iX /= 10;
      }
   Console.WriteLine();
   }
}
```

Display will be __it will not display anything because there is not Console.ReadLine(); at the end of it. With that added, it should display 12351.________

29. We have a stack object (10 points):

- What is while ( ) statement do?
  Keep running the code while the stack has values in it and is not empty.

- What is IsEmpty?
  Is a Boolean that returns false when sp is not 0 (there are still values in the stack) and returns true when it is 0 (no more values).

- What is Pop()?
  Moves the sp to the point on top and returns that value.

- What is {0}?
  The top value of the stack.

- What is the code below going to do?

```
while (!stack.IsEmpty)

    {

       Console.WriteLine("Popping {0}", stack.Pop());

    }
```

While the stack is not empty and still has values in it, it will write each value in the stack starting from the one that was last pushed.

30. What is the difference between a “Class” and a “Struct” in C#? (10 points)

The difference between a struct and a class is that you don’t need to call “new” unless there is a constructor. You also cannot inherit from a struct or can a struct inherit from another type.