CSC180 Final Exam

**1.** A word is abecedarian if the letters in the word appear in alphabetical order. Complete the method IsAbecedarian below.

using System;

namespace CSC180FinalExam

{

class Program

{

static void Main(string[] args)

{

string[] arr = { "apple", "best", "biopsy", "cestuy", "floor" };

foreach (var item in arr)

{

Console.WriteLine("Is " + item + " abecedarian?\t" + IsAbecedarian(item));

}

}

static bool IsAbecedarian(string w)

{

int n = w.Length;

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

{

if (w[i] < w[i - 1])

return false;

}

return true;

}

}

}

**2. The Complex class can be found in the System.Numerics namespace. Basically it is defined as following:**

public struct Complex

{

public int real;

public int imaginary;

public Complex(int real, int imaginary)

{

this.real = real;

this.imaginary = imaginary;

}

//more code is omitted here ...

}

1) Transform the following static method into an instance method.

public static double Abs(Complex c)

{

return Math.Sqrt(c.Real \* c.Real + c.Imaginary \* c.Imaginary);

}

1. I removed static.

2. I changed the name of the method to be a verb.

3. I removed the parameter.

4. Inside an instance method you can refer to fields as if they were local variables, so I

changed c.Real to real, and likewise for imaginary.

public double FindAbs()

{

Complex c = new Complex(2,2);

return Math.Sqrt(real \* real + imaginary \* imaginary);

}

2) Transform the following instance method into a static method (fill the places marked with ???).

public bool Compare(Complex b)

{

return (Real == b.Real && Imaginary == b.Imaginary);

}

public static bool Compare(???)

{

return ???;

}

public static bool Compare2(Complex x, Complex y)

{

return (x.real == y.real && x.imaginary == y.imaginary); ;

}

**3.** Use a few sentences to describe the program.

namespace CSC180FinalExam

{

public class Program

{

public static void Main()

{

// integer array arr is already sorted

int[] arr = { 1, 2, 5, 13, 18, 22 };

System.Console.WriteLine(DoSomething(arr, 6));

System.Console.WriteLine(DoSomething(arr, 13));

}

static int DoSomething(int[] arr, int num)

{

int left = 0;

int right = arr.Length - 1;

int middle;

while (left <= right)

{

middle = (left + right) / 2;

if (num == arr[middle])

{

return ++middle;

}

else if (num < arr[middle])

{

right = middle - 1;

}

else

{

left = middle + 1;

}

}

return -1;

}

}

}

This is an example of a binary search, or bisection search. With the array already sorted, choose an index that is between low and high. That is the middle value. The high and low value are left and right in the array. When you search with int num, and you don’t find the number, you bisect the values and ask if it is lower or higher(binary). If the int num is present, program returns the index in the array. If not, program returns -1. Binary search has the advantage of being faster because program is searching both ends of an array.

**4.** Write a method that prints the three numbers in ascending order.

using System;

class GFG

{

public static void Main()

{

Sort(5, 4, 3);

}

public static void Sort(double x, double y, double z)

{

int[] a = { (int)x, (int)y, (int)z };

Array.Sort(a);

for (int i = 0; i < 3; i++)

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

}

}