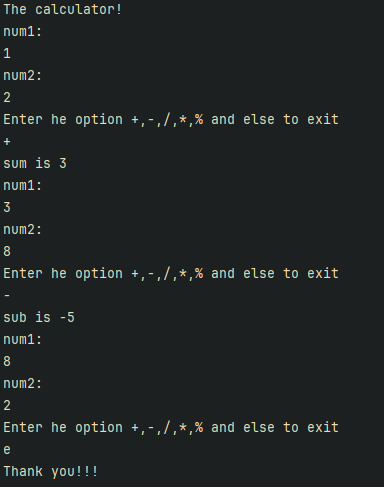
Day 3

1. Calculator:

Code:

namespace Calculator  
{  
 internal class Program  
 {  
 static void Main(string[] args)  
 {  
 Console.WriteLine("The calculator!");  
 while(true)  
 {  
 Console.WriteLine("num1: ");  
 double a = Convert.ToDouble(Console.ReadLine()??"0");  
 Console.WriteLine("num2: ");  
 double b = Convert.ToDouble(Console.ReadLine()??"0");  
 Console.WriteLine("Enter he option +,-,/,\*,% and else to exit");  
 char opt = Convert.ToChar(Console.ReadLine()??"e");  
  
 switch (opt)  
 {  
 case '+':  
 Console.WriteLine($"sum is {add(a, b)}");  
 break;  
 case '-':  
 Console.WriteLine($"sub is {sub(a, b)}");  
 break;  
 case '\*':  
 Console.WriteLine($"multiplication is {multiply(a, b)}");  
 break;  
 case '/':  
 Console.WriteLine($"division is {divide(a, b)}");  
 break;  
 case '%':  
 Console.WriteLine($"remainder is {remainder(a, b)}");  
 break;  
 default:  
 Console.WriteLine("Thank you!!!");  
 return;  
 }  
  
  
 }  
 }  
  
 static double add(double a, double b)  
 {  
 return a + b;  
 }  
  
 static double sub(double a, double b)  
 {  
 return a - b;  
 }  
   
 static double multiply(double a, double b)  
 {  
 return a \* b;  
 }  
 static double divide(double a, double b)  
 {  
 return a / b;  
 }  
  
 static double remainder(double a, double b)  
 {  
 return a % b;  
 }  
 }  
}

Output:

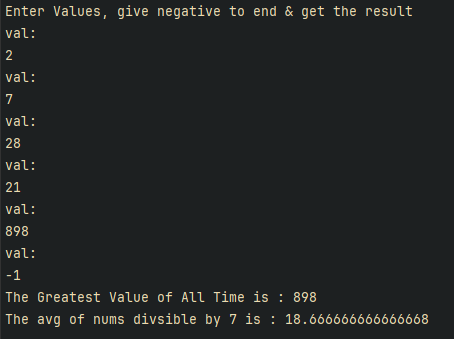


1. Greatest Num & avg of nums divisible by 7

Code:

namespace GreatestNum  
{  
 internal class Program  
 {  
 static void Main(string[] args)  
 {  
 Console.WriteLine("Enter Values, give negative to end & get the result");  
 double max = 0;  
 double avgDivBySeven = 0;  
 int avgDivBySevenCount = 0;  
 while (true)  
 {  
 Console.WriteLine("val:");  
 double input = double.Parse(Console.ReadLine()??"0");  
 if(input <= 0)  
 {  
 break;  
 }  
 if(input%7 == 0)  
 {  
 avgDivBySeven += input;  
 avgDivBySevenCount++;  
 }  
 max = input > max ? input : max;  
 }  
 Console.WriteLine($"The Greatest Value of All Time is : {max}");  
 Console.WriteLine($"The avg of nums divsible by 7 is : {avgDivBySeven / avgDivBySevenCount}");  
 }  
 }  
}

Output:



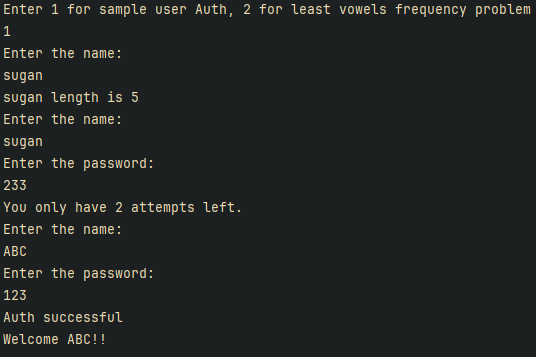
1. Basic auth visualization

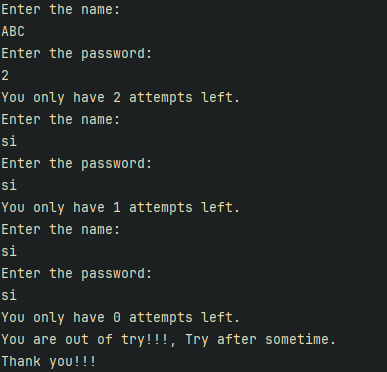
Code:

Console.WriteLine("Enter the name:");  
string name = Console.ReadLine() ?? "";  
Console.WriteLine($"{name} length is {GetSize(name)}");  
  
int maxTry = 3;  
while (maxTry > 0)  
{  
 Console.WriteLine("Enter the name:");  
 name = Console.ReadLine() ?? "";  
 Console.WriteLine("Enter the password:");  
 string password = Console.ReadLine() ?? "";  
 if (Auth(name, password))  
 {  
 Console.WriteLine("Auth successful");  
 Console.WriteLine($"Welcome {name}!!");  
 return;  
 }  
  
 maxTry--;  
 Console.WriteLine($"You only have {maxTry} attempts left.");  
  
}  
Console.WriteLine("You are out of try!!!, Try after sometime.");

static bool Auth(string username, string password)  
{  
 return username == "ABC" && password == "123";  
}  
static int GetSize(string name)  
{  
 return name.Length;  
}

Output:



If we did the max attempts  


1. Ranking words by least repeating vowels

Code:

var vowels = new Vowels();  
 Console.WriteLine("Enter the comma seperated string for words with least vowels");  
 var str = Console.ReadLine();  
 Console.WriteLine(str);  
 var res = vowels.GetLeastWords(str ?? "");  
 Console.WriteLine("hereee");  
 foreach (var item in res.Keys)  
 {  
 Console.WriteLine(item);  
 Console.WriteLine($"count {res[item]}");  
 }  
}  
Console.WriteLine("Thank you!!!");

namespace UserValidation  
{  
 internal class Vowels  
 {  
 private readonly HashSet<char> \_vowels = new HashSet<char> { 'a', 'e', 'i', 'o', 'u'};  
  
 */\**  
 *\* for getting Frequency map*  
 *\*/*  
private Dictionary<char, int> GetFrequency(string str)  
 {  
 var map = new Dictionary<char, int>();  
 foreach (var ch in str.ToCharArray())  
 {  
 if (\_vowels.Contains(ch))  
 {  
 map[ch] = map.GetValueOrDefault(ch, 0) + 1;  
 }  
 }  
 return map;  
 }  
  
 private int GetVowelsCount(string str)  
 {  
 var count = 0;  
 foreach(var ch in str.ToCharArray())  
 {  
 if (\_vowels.Contains(ch))  
 count++;  
 }  
 return count;  
 }  
  
 public Dictionary<string, int> GetLeastWords(string input)  
 {  
 var inputList = input.Split(",").ToList();  
 var leastCount = GetFreqNum(inputList[0]);  
 var res = new Dictionary<string, int>();  
  
 foreach (var word in inputList) {  
 var currCount = GetFreqNum(word);  
 if( leastCount > currCount)  
 {  
 leastCount = currCount;  
 res.Clear();  
 res[word] = GetVowelsCount(word);  
 }  
 }  
 return res;  
  
 }  
 */\**  
 *\* Returns a specialised comparitor num,*  
 *\* for the vowels aaeiio it will be generated as 1122*  
 *\* as frequencies in sorted order*  
 *\*/*  
private int GetFreqNum(string word)  
 {  
 var wordFreq = GetFrequency(word);  
 var freqStore = new List<int>();  
 foreach (var item in wordFreq.Values)  
 {  
 freqStore.Add(item);  
 }  
 freqStore.Sort();  
  
 var compNo = "";  
  
 foreach (var item in freqStore)  
 {  
 compNo += item;  
 }  
 return int.Parse(compNo);  
 }  
  
 }  
}

Output:

