ASSIGNMENT 3

Question 4

Pig Latin is an English-language word game, usually played by children, in which words are mutated systematically. A word that begins with a vowel (a, e, i, o, or u) has the syllable “way” added to the end; for instance, art becomes art-way and eagle becomes eagle-way. A word that begins with one or more consonants has the initial consonants stripped, moved to the end of the word, then the syllable “ay” is added; for instance, start becomes art-stay and door becomes oor-day. A hyphen is added as shown above as an aid to prevent ambiguity; otherwise, a word like aspray could be the translation of spray (ay-spray) or prays (ays-pray). Even so, some words remain ambiguous; art-way could be translated as either art or wart.

Your task is to write functions that translate English to Pig Latin and Pig Latin to English.

PROGRAM:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace assignment3in13

{

class Program

{

static void Main(string[] args)

{

string sentence;

string firstLetter;

string afterFirst;

string aftervow;

string firstphrase;

string pigLatinOut = "";

int x;

string vowel = "AEIOUaeiou";

Console.WriteLine("Enter a sentence to convert into PigLatin");

sentence = Console.ReadLine();

Console.WriteLine(" ---------------------------------- ");

//each word is split inside the array

string[] pieces = sentence.Split();

int a=0,z=0;

foreach (string piece in pieces)

a++;

int wordlen = a;

string[] pigpiece =new string[a];

int f = 0;

a = 0;

List<int> myInts = new List<int>();

int[] u = new int[5];

foreach (string piece in pieces)

{

//checking first letter to be vowel or not and seperating

afterFirst = piece.Substring(1);

firstLetter = piece.Substring(0, 1);

x = vowel.IndexOf(firstLetter);

z=piece.Length;

//keep index in array to eleminate the -1

u[0]=piece.IndexOf('a');

u[1]=piece.IndexOf('e');

u[2]=piece.IndexOf('i');

u[3]=piece.IndexOf('o');

u[4]=piece.IndexOf('u');

foreach (int v in u)

if (v >= 0)

myInts.Add(v);

f=myInts.Min();

//seperating the first phrase before vowel

firstphrase = piece.Substring(0,f);

aftervow = piece.Substring(f);

if (x == -1)

pigLatinOut = aftervow +"-"+firstphrase+ "ay ";

else

pigLatinOut = (firstLetter + afterFirst + "-way ");

//storing in array to use it for reconversion

pigpiece[a++] = pigLatinOut;

Console.Write(pigLatinOut);

myInts.Clear();

}

Console.WriteLine(" ---------------------------------- ");

int pospace = 0;

int g = 0;

string j,n;

foreach (string pig in pigpiece)

{

//gets index of hyphen for separation use

x = pig.IndexOf('-');

//gets index of space for end point

pospace= pig.IndexOf(' ');

if (x == 1)

{

j = pig.Substring(0, 1);

Console.Write(j + " ");

}

else

{

//split array forconversion from piglatin to english

g = (pospace-3) - x;

j = pig.Substring(x+1,g);

n = pig.Substring(0, x);

Console.Write(j +n+ " ");

}

}

Console.ReadKey();

}

}

}

NO

YES

NO

YES

START

STOP

Print it in Pig Latin

Take hyphen and space and position and split the required string and print it in English

Firstleter+afterfirst

+”-way”

Aftervow+”-“+firstphrase+”ay”

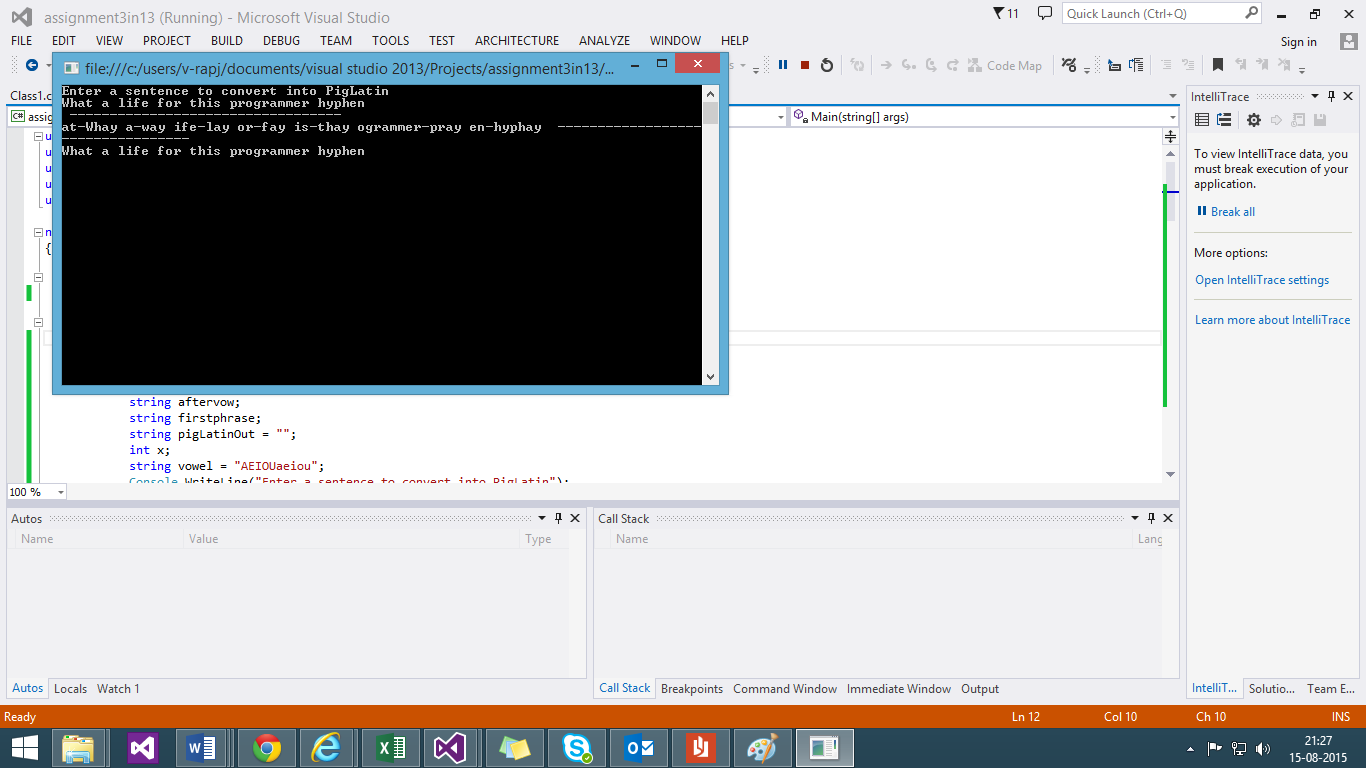
If vowel pos>0

Get the position of vowels and form front and back part attach the two parts by checking the position of vowel

Use foreach and split sentence into array of string

ALGORITH:

1. Use Foreach loop to split the sentence into array of words.
2. Check the first letter for vowel and add “-way ” if so else
3. Get the vowel index and cut the first phrase before vowel and from vowel as two parts and concat such that from vowel + “-”+first phrase+ “ay ”
4. Print the pig Latin converted words
5. Save the pig Latin words in another string array.
6. Use the index of hyphen and space to get the old English words back.



Question 15

Write a function that takes a string and determines if the delimiters in the string are balanced. The pairs of delimiters are (), [], {}, and <>, and delimiters may be nested. In addition, determine that string delimiters ‘ and ” are properly matched; other delimiters lose their magical delimiter-ness property within quoted strings. Any delimiter is escaped if it follows a backslash.

Your task is to write the function to determine if a string has balanced delimiters.

PROGRAM:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace assignment3in13

{

class Class1

{

static void Main(string[] args)

{

char[] stack = new char[100];

int i, top = -1;

bool flag = false;

//get the equation

string inp = Console.ReadLine();

char[] input = inp.ToCharArray();

//clear the stack

for (i = 0; i < 100; i++)

stack[i] = '\0';

//loop till the last element of the equation

for (i = 0; i < inp.Length; i++)

{

//check for single or double Apostrophe and top is not input and push Apostrophe

if ((input[i] == '\'' || input[i] == '\"') && stack[top] != input[i])

{

flag = true;

stack[++top] = input[i];

}

//check for single or double Apostrophe and top is input clear flag and pop Apostrophe

else if ((input[i] == '\'' || input[i] == '\"') && stack[top] == input[i])

{

flag = false;

stack[top--] = '\0';

}

//open braces are accounted and if flag is false the braces are pushed into stack

else if ((input[i] == '(' || input[i] == '[' || input[i] == '{' || input[i] == '<') && !flag)

stack[++top] = input[i];

//close braces are accounted the braces are pop out from stack

else if ((input[i] == ']' || input[i] == '}' || input[i] == '>') && input[i] == stack[top] + 2)

stack[top--] = '\0';

//close the last braces and accoount fot second in the stack and pop

else if (input[i] == ')' && input[i] == stack[top] + 1)

stack[top--] = '\0';

Console.WriteLine("Iteration {0} : {1}", i, stack[i]);

}

//if stack is null balenced else unbalenced

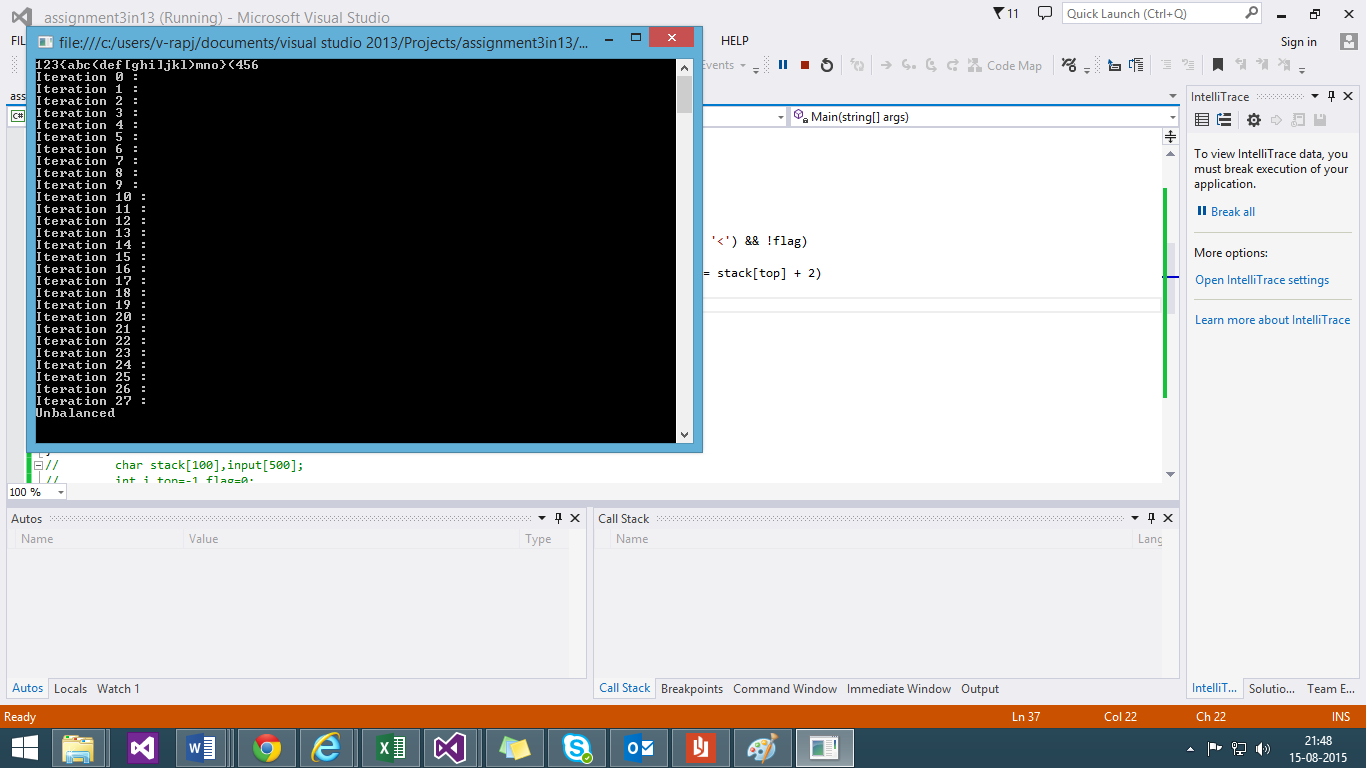
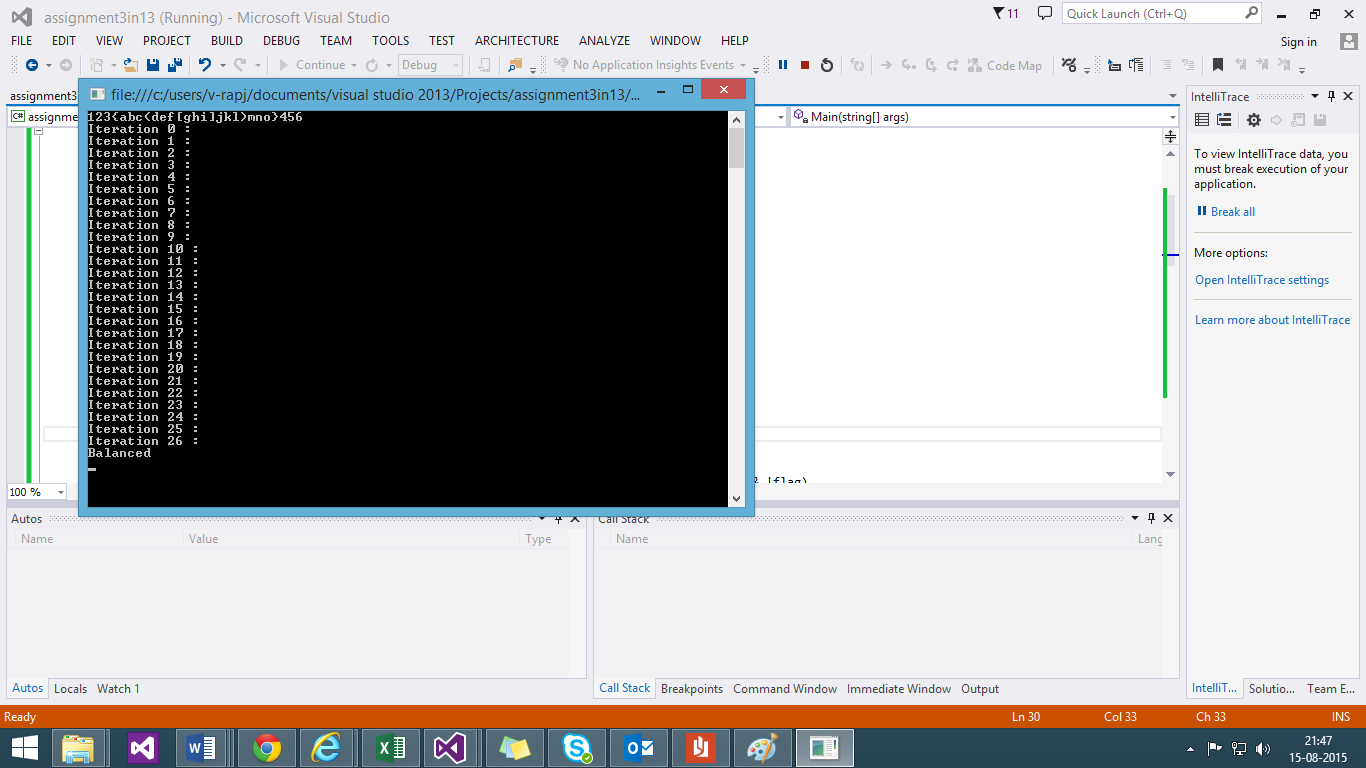
Console.WriteLine("{0}",stack[0] == '\0' ? "Balanced" : "Unbalanced");

Console.ReadKey();

}

}

}



ALGORITHM:

1. get the equation
2. clear the stack and loop till the last element of the equation
3. In the loop check for single or double Apostrophe and top is not input and push Apostrophe
4. open braces are accounted and if flag is false the braces are pushed into stack
5. close braces are accounted the braces are pop out from stack
6. After loop ends if stack is null balanced else unbalanced.

no

yes

STOP

Print result of balanced or not

no

yes

Pop from stack and negate the flag for Apostrophe

If

Input[i] =

{‘\’’,’\”’,’)’,’}’,’]’,’>’}

Push into stack the braces and account Apostrophe for flag

no

yes

If

Input[i] =

{‘\’’,’\” ‘,’{‘,’[‘,’(‘,’<’}

Get the equation and clear the stack loop till last element in the equation

START