Fundamental Computer Programming- C++ Lab(I)

LAB 5
Word Processing

Week 5, Fall 2022

International Bachelor Program in Informatics
College of Informatics
Yuan Ze University

Purposes

- Get familiar with the basic structure of a C++ program
- Get familiar with some control statements such as:
  - if (…) {… } else {…}
  - while (..) { ….}
  - for(…) {…},
  - switch(…) {…}, etc.
- Get familiar with the use of logical operators &&, ||, !, etc.
- Get familiar with an array
- Develop problem solving skills

switch (…) {…}

```
switch (anIE)
{
      case aCIE1: cout << "1";
      case aCIE2: cout << "2";
     //execution starts at this case label
      case aCIE3: cout << "3";
      case aCIE4: cout << “4";
                 break;
    //execution of subsequent statements is stopped
      case aCIE5: cout << “5";
      default:
                 cout << “The rest”;
                break;
   }
```

anIE: **an expression evaluated into an integer**.

For example,

Int x=1;

Int y=3;

char z;

x+y is evaluated into an integer.

x+z is evaluated into an integer.

x+y+’x’ is evaluated into an integer.

x, y, and z themselves each are evaluated into an integer. However, all the above expressions are not a constant integral expression.

aCIE1: **an expression evaluated into a constant integer value.** That is, it is a constant integral expression. The same is for **aICE2**, **aCIE3**, and **aCIE**4. For example,

const int i=2;

const char c=‘a’;

i+c is a constant integral expression.

Both i and c each are a constant integral expression.

break & continue

- **break** is used to exit from a loop or from a switch statement
- **continue** is used to skip the statements following it till the end of the loop and continue executing the next iteration.

After "continue" is executed, k++ statement will be executed first and then k < 5 is checked. That is, the statements are executed in order of ①, ②, ③ after "continue" is executed.

```
for (int j = 0; j < 2; j++) {
                          ②    ①
    for (int k = 0; k < 5; k++) {
    //only this loop is affected by break
     ③ if (k == 2) break;
        if (k == 3) continue;
        cout << j << k << " ";
     }
          //cout << j+k << endl;
}
```

```
int i = 3;    // 34 is printed out.
switch (i) {
   case 1: cout << "1";
   case 2: cout << "2";
            break;
  //execution starts at this case label
   case 3: cout << "3";
   case 4: cout << "4";
           break;
   //execution of subsequent statements is
   //terminated
   case 5: cout << "5";
}
```

What if i = 4?

If there is a match, the statements after the matched case and that in the following cases before a break is met will be executed.

https://en.cppreference.com/w/cpp/language/break

LAB 5: Word Processing

- **Problem description**
  - **You are given a number of words from keyboard. A word is formed by a sequence of symbols. Words are separated by space, tab, or newline. A vowel is one of a, e, i, o, u, A, E, I, O, and U.**
  - **You are asked to calculate**
    - ✓ **total number of words, each of which has at least a letter, total number of words whose first letter is a vowel, and total number of words, each of which contains some letter repeated at least twice. The upper case letter and lower case letter are treated as the same letter. Note that the first letter of a word may not be its first symbol.**
    - ✓ **Calculate the distribution of words by word length where the length of a word counts only letters. A word whose length is greater than 15 is treated as if its length is equal to 15.**
    - ✓ **Calculate the distribution of words by the first letter. Here, the lower case letter and upper-case letter of the same alphabet are treated as the same letter. For example A and a are treated as the same letter.**

Input

- **Input**
  - Your program should accept an unknown number of words. You should use “ctrl z” (in Windows) or “ctrl d” (in Unix, Mac OS) to terminate reading input from keyboard.

- An example for input

cout << "Total number of words: " << initVowelNum + nonVowelNum + firstCharNum[26] + firstCharNum[27] << endl;

This is a java program to implement monoalphabetic cypher. In cryptography, a substitution cipher is a method of encoding by which units of plaintext are replaced with ciphertext, according to a regular system; the “units” may be single letters (the most common), pairs of letters, triplets of letters, mixtures of the above, and so forth. The receiver deciphers the text by performing an inverse substitution. 134Abc, 7sdjk, wekjf0, 23i5, Ab AB ab aB cD Ef gh IJ KL Mn op Qr St UV wx YZ Y Za ZA za zA Ab bc cd dE ef Fg Gh hi IJ JK kl LM mN No Op pq Qr Rs St TU uV vw Wx Xy YZ Za For example, eve, ada, Noun, rear, Sears, pop, kick, Sos, etc. are words whose first letter and last letter are the same.

Input example for Code::Block

```
cout << "Total number of words: " << initVowelNum + nonVowelNum + firstCharNum[26] + firstCharNum[27]
<< endl;
This is a java program to implement monoalphabetic cypher. In cryptography, a substitution cipher is a
 method of encoding by which units of plaintext are replaced with ciphertext, according to a regular s
ystem; the "units" may be single letters (the most common), pairs of letters, triplets of letters, mix
tures of the above, and so forth. The receiver deciphers the text by performing an inverse substitutio
n. 134Abc, 7sdjk, wekjf0, 23i5, … . Ab AB ab aB cD Ef gh IJ KL Mn op Qr St UV wx YZ Y Za ZA za zA
Ab bc cd dE ef Fg Gh hi IJ JK kl LM mN No Op pq Qr Rs St TU uV vw Wx Xy YZ Za For example, eve, ada, N
oun, rear, Sears, pop, kick, Sos, etc. are words whose first letter and last letter are the same.
^Z
```

- **“according”** contains at least one letter. Its first letter is a vowel. It contains a letter repeated twice. Its word length is 9.
- **+** is a word that does not contain a letter. Its word length is 0.
- **Letters** contains at least one letter. Its first letter is not a vowel. It contains two letters each repeated twice. Its word length is 7.

Pay attention to the use of ^Z to stop reading input, where ^Z corresponds to “ctrl z” on the keyboard. ctrl means control. ^Z should be followed by an Enter key.

Output

- You should print out something like what is shown below

 # Total number of words, each of which has at least a letter: 189
 # Total number of words whose first letter is a vowel: 49
 # Total number of words, each of which contains some letter repeated at least twice: 67
 # Distribution of words by word length:
 # L0 = 0
 # L1 = 9
 # L2 = 66
…
…
 # L14 = 1
 # L15 = 2

L1 = 9 means that there are 9 words with length equal to 1.

L15 = 2 means that there are 2 words with length greater than or equal to 15.

 # Distribution of words by the first letter:
 # A/a: 18
 # B/b: 4
 # C/c: 8
…
…
 # Y/y: 3
 # Z/z: 5

A/a: 18 means that the number of words whose first letter is either **A** or **a** is 18.

Input Example

cout << "Total number of words: " << initVowelNum + nonVowelNum + firstCharNum[26] + firstCharNum[27] << endl;
This is a java program to implement monoalphabetic cypher. In cryptography, a substitution cipher is a method of encoding by which units of plaintext are replaced with ciphertext, according to a regular system; the “units” may be single letters (the most common), pairs of letters, triplets of letters, mixtures of the above, and so forth. The receiver deciphers the text by performing an inverse substitution. 134Abc, 7sdjk, wekjf0, 23i5, Ab AB ab aB cD Ef gh IJ KL Mn op Qr St UV wx YZ Y Za ZA za zA Ab bc cd dE ef Fg Gh hi IJ JK kl LM mN No Op pq Qr Rs St TU uV vw Wx Xy YZ Za For example, eve, ada, Noun, rear, Sears, pop, kick, Sos, etc. are words whose first letter and last letter are the same.

Output for the Example

```
# Total number of words, each of which has at least a letter: 148
# Total number of words whose first letter is a vowel: 47
# Total number of words, each of which contains some letter repeated at least twice: 41
# Distribution of words by word length:
# L0 =  7
# L1 =  6
# L2 =  62
# L3 =  19
# L4 =  13
# L5 =  13
# L6 =  9
# L7 =  7
# L8 =  5
# L9 =  4
# L10 =  2
# L11 =  1
# L12 =  6
# L13 =  0
# L14 =  1
# L15 =  0
```

Output for the Example cont.

```
# Distribution of words by the first letter:
# a/A: 19
# b/B: 4
# c/C: 8
# d/D: 2
# e/E: 7
# f/F: 6
# g/G: 2
# h/H: 1
# i/I: 9
# j/J: 2
# k/K: 3
# l/L: 7
# m/M: 7
# n/N: 4
# o/O: 8
# p/P: 6
# q/Q: 2
# r/R: 5
# s/S: 11
# t/T: 13
# u/U: 4
# v/V: 1
# w/W: 8
# x/X: 1
# y/Y: 3
# z/Z: 5
```

Another Input Example

Each string is a character array. For example, if aStr is a string variable that stores a string “He loves you.”, aStr[0] will contain ‘H’, sStr[1] contains ‘e’, aStr[2] contains ‘ ‘, aStr[3] contains ‘l’, aStr[4] contains ‘o’, ... We can use a function aStr.length() to get the length of the string stored in aStr. It will return 13.
Each character is stored in computers using ASCII. It is an integer. So we can have the following code.

```
int aDigit, anInt;
aDigit = ‘9’ – ‘0’;   // aDigit will have a value of 9.
anInt = ‘z’ – ‘a’;    // anInt will have a value of 25.
```

You may need use static_cast<char>(90) to convert an integer to a character.

```
Each string is a character array. For example, if aStr is a string variable that stores a string "He loves you.", aStr[0
] will contain  ‘H’ , sStr[1] contains  ‘e’ , aStr[2] contains  ‘ ‘, aStr[3] contains  ‘l’ , aStr[4] contains  ‘o’ , …
. We can use a function aStr.length() to get the length of the string stored in aStr. It will return 13.
Each character is stored in computers using ASCII. It is an integer. So we can have the following code.
int aDigit, anInt;
aDigit =  ‘9’  -  ‘0’ ;   // aDigit will have a value of 9.
anInt =  ‘z’  -  ‘a’ ;    // anInt will have a value of 25.

You may need use static_cast<char>(90)  to convert an integer to a character.
^Z
```

Output for Another Example

```
# Total number of words, each of which has at least a letter: 105
# Total number of words whose first letter is a vowel: 48
# Total number of words, each of which contains some letter repeated at least twice: 33
# Distribution of words by word length:
# L0 =  14
# L1 =  13
# L2 =  21
# L3 =  13
# L4 =  19
# L5 =  9
# L6 =  12
# L7 =  5
# L8 =  6
# L9 =  5
# L10 =  1
# L11 =  0
# L12 =  0
# L13 =  0
# L14 =  1
# L15 =  0
```

Output for Another Example cont.

```
# Distribution of words by the first letter:
# a/A: 25
# b/B: 0
# c/C: 13
# d/D: 0
# e/E: 4
# f/F: 3
# g/G: 1
# h/H: 5
# i/I: 12
# j/J: 0
# k/K: 0
# l/L: 3
# m/M: 1
# n/N: 1
# o/O: 4
# p/P: 0
# q/Q: 0
# r/R: 1
# s/S: 10
# t/T: 7
# u/U: 3
# v/V: 3
# w/W: 6
# x/X: 0
# y/Y: 2
# z/Z: 1
```

Hint: Basic structure of code

- **Declaration**
  - An string variable to hold the input read from keyboard one at a time.
  - Some int arrays to register the distributions of words, to store the number of times that a letter appears, etc. Note that some arrays may need to be reinitialized for each string read from the keyboard.
- **A while loop to read the words from keyboard**
- **Print out the statistics and distributions**

C++ string and characters

- Each string is a character array. For example, if aStr is a string variable that stores a string “He loves you.”, aStr[0] will contain ‘H’, sStr[1] contains ‘e’, aStr[2] contains ‘ ‘, aStr[3] contains ‘l’, aStr[4] contains ‘o’, …. We can use a function aStr.length() to get the length of the string stored in aStr. It will return 13.
- Each character is stored in computers using ASCII. It is an integer. So we can have the following code.

```
int aDigit, anInt;
aDigit = ‘9’ – ‘0’;   // aDigit will have a value of 9.
anInt = ‘z’ – ‘a’;    // anInt will have a value of 25.
```

You may need use static_cast<char>(90) to convert an integer to a character.