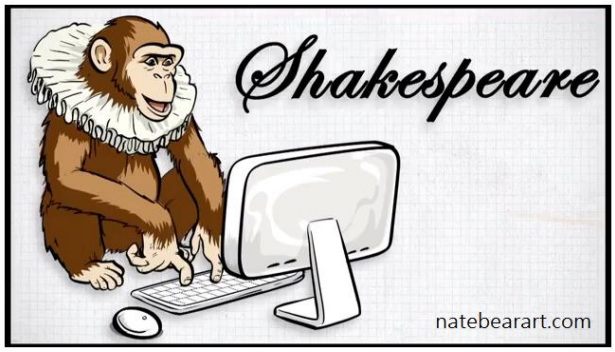

8.11 ***(Random Sentences)*** Write a program that uses random number generation to create sentences. The program should use **four arrays of pointers** to char called **article**, **noun**, **verb** and **preposition**. The program should create a sentence by selecting a word at random from each array in the following order:

Article + noun + verb + preposition + article + noun.

As each word is picked, it should be concatenated to the previous words in an array large enough to hold the entire sentence. The words should be separated by spaces. When the final sentence is output, it should start with a capital letter and end with a period. The program should generate 20 such sentences. The arrays should be filled as follows: The **article** array should contain the articles "the", "a", "one", "some" and "any"; the **noun** array should contain the nouns "boy", "girl", "dog", "town" and "car"; the **verb** array should contain the verbs "drove", "jumped", "ran", "walked" and "skipped"; the **preposition** array should contain the prepositions "to", "from", "over", "under" and "on". After the preceding program is written and working, modify it to produce a short story consisting of several of these sentences. (How about the possibility of a random term paper writer?)

ANS:

```
A dog skipped to any car.
Some town ran on the boy.
A dog jumped from the dog.
One girl jumped on one town.
One dog jumped from some boy.
One girl jumped under any dog.
One car drove on some girl.
One town walked on a girl.
Some town ran on one dog.
One car walked from any town.
A boy drove over some girl.
The dog skipped under a boy.
The car drove to a girl.
Some town skipped under any car.
A boy jumped from a town.
Any car jumped under one town.
Some dog skipped from some boy.
Any town skipped to one girl.
Some girl jumped to any dog.
The car ran under one dog.
```

8.31(a) ***(Text Analysis)*** Write a program that reads three lines of text and prints a table indicating the number of occurrences of each letter of the alphabet in the text. For example, the phrase "To be, or not to be: that is the question:" contains one "a," two "b's," no "c's," and so on.

ANS:

```
Enter three lines of text:
This program counts the occurrences of each
letter of the alphabet in the input text. Then,
it prints a summary of the occurrences.

Total letter counts in the three-line text are as follows:
a:  6
b:  1
c:  8
d:  0
e: 14
f:  3
g:  1
h:  8
i:  5
j:  0
k:  0
l:  2
m:  3
n:  7
o:  7
p:  4
q:  0
r:  9
s:  6
t: 15
u:  5
v:  0
w:  0
x:  1
y:  1
z:  0
```

8.34 ***(Writing the Word Equivalent of a Check Amount)*** Continuing the discussion of the previous example, we reiterate the importance of designing check-writing systems to prevent alteration of check amounts. One common security method requires that the check amount be both written in numbers and "spelled out" in words. Even if someone is able to alter the numerical amount of the check, it is extremely difficult to change the amount in words. Write a program that inputs a numeric check amount and writes the word equivalent of the amount. For example, the amount 52.43 should be written as "FIFTY-TWO and 43/100".

```
Enter a check amount (0.01 ~ 99.99, 0.00 to end): 72.63
The check amount in the written equivalent words: SEVENTY-TWO and 63/100

Enter a check amount (0.01 ~ 99.99, 0.00 to end): 13.22
The check amount in the written equivalent words: THIRTEEN and 22/100

Enter a check amount (0.01 ~ 99.99, 0.00 to end): 5.75
The check amount in the written equivalent words: FIVE and 75/100

Enter a check amount (0.01 ~ 99.99, 0.00 to end): 0.00
The check amount in the written equivalent words: ZERO and 0/100
```

ANS: