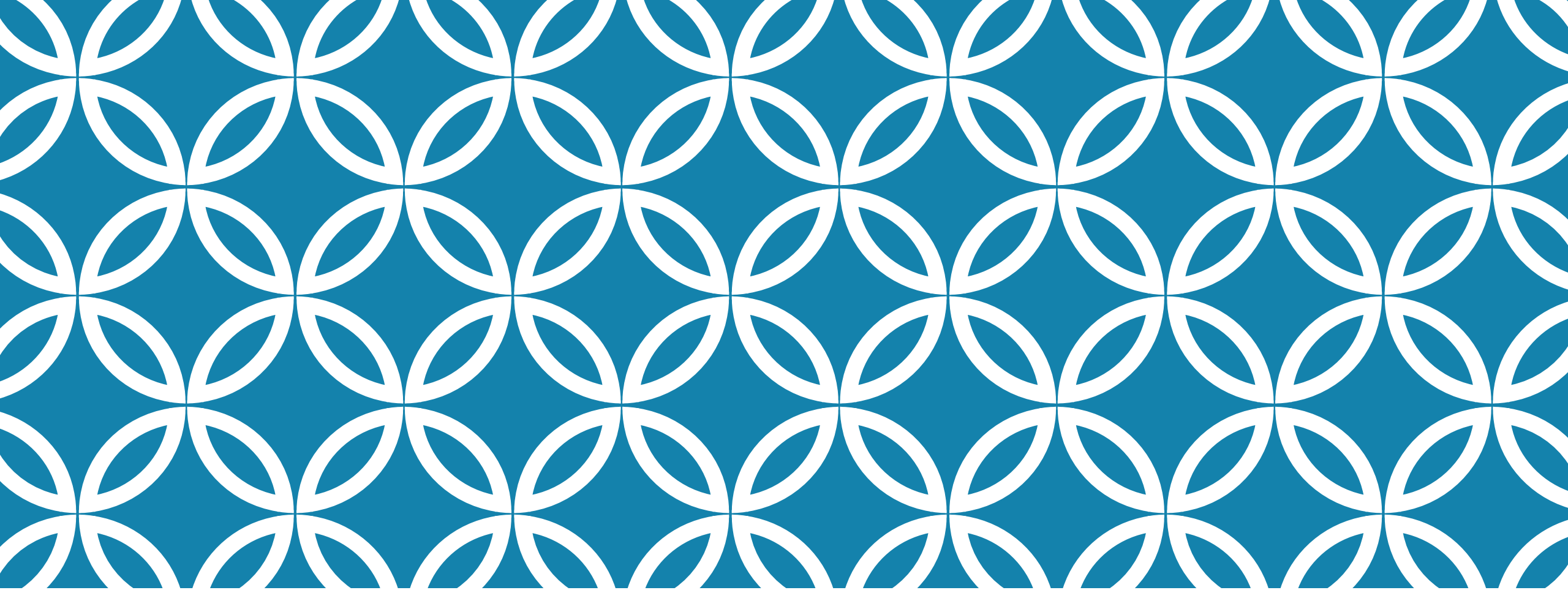

COMP 5481 PROGRAMMING AND PROBLEM SOLVING

Tutorial #3
Predict Patterns

WHAT IS N-GRAM?

An “n-gram” is a contiguous sub-sequence of *n* items in a given sequence.

For example, given the sequence “ALIGATOR”, its only 5-grams are ALIGA, LIGAT, IGATO and GATOR. There are special names for the first few n-grams: unigram for 1-gram, bigram for 2-gram, and trigram for 3-gram.

Areas of Application:

- Computer Science
- Computational Linguistics
- Applied mathematics

PROBLEM

Write a program that, given a paragraph, will find the most-frequently appearing n-gram in it. We are interested in n-grams consisting of lower-case letters only. More specifically, you need to find "the single letter that appears the most (Unigram)", "two consecutive letters that appear the most (Bigram)", and "three consecutive letters that appear the most (Trigram)". If there is a tie for most number of occurrences, (e.g., several bigrams appearing the most), print the one that appears first alphabetically (i.e., the smallest in string comparison). Note that "consecutive" letters means one letter immediately after another letter, i.e., no other characters (spaces or other separators) in between.

INPUT FORMAT

The first line of the input is an integer p (0<p<51) indicating the number of lines in the paragraph. The following p input lines provide the text for the paragraph.

Each of these input lines will contain only lowercase letters, spaces, commas and periods.

Assume that these input lines will not exceed column 70 and that each line will contain at least one letter.

The only separators are spaces, commas, periods, and end-of-line.

The last line in the input is an integer n (0<n<4). It indicates n-gram to be printed as the output (1->Unigram, 2->Bigram and 3-> Trigram).

INPUT LINES

```
// This code allows a user to read a number for input lines from System.in
     Scanner sc_in = new Scanner(System.in);
     int lines = sc_in.nextInt();
// Running loop to get all input lines
     String input="";
     for(int i=0; i<lines;i++)
     {
       input += " " + sc_in.nextLine();
     }
```

INPUT FOR N-GRAM

```
// This code is for ngram integer input, which is expected to be in the range 0 < ngram < 4.
    int ngram = sc_in.nextInt();
    if(ngram < 1 || ngram > 3)
      return;
    …

    switch(ngram)
    {
      case 1: uni … break;
      case 2: bi …  break;
      case 3: tri … break;
    }
```

STRING TOKENIZER

`public StringTokenizer(String str, String delim)`

Constructs a string tokenizer for the specified string. The characters in the delim argument are the delimiters for separating tokens. Delimiter characters themselves will not be treated as tokens.

`public boolean hasMoreTokens()`

Tests if there are more tokens available from this tokenizer's string.

Returns **true** if and only if there is at least one token in the string after the current position; **false** otherwise.

`public String nextToken()`

Returns the next token from this string tokenizer.

CASE FOR UNIGRAM

```
uni = new int[26];

while(s.hasMoreTokens())
{
    String word = s.nextToken();

    for(int i=0;i<word.length();i++)
    {
// charAt function returns the char value at the specified index.
        uni[word.charAt(i)-'a']++;
    }
}
```

CASE FOR UNIGRAM (CONTD..)

```
 // Pick the best, if there is a tie print the one that appears
first alphabetically (i.e., smallest in string comparison).

 for(int i=0;i<26;i++)
 {
    if(uni[i]>best)
    {
       besti = i;
       best = uni[i];
    }
 }

 System.out.println("Unigram "+(char)(besti+'a'));
```

CASE FOR BIGRAM

```
bi = new int[26][26];

while(…)
{
    …

    for(int i=1;i<word.length();i++)
    {
// charAt function returns the char value at the specified index.
        bi[word.charAt(i-1)-'a'][word.charAt(i)-'a']++;
    }
}
```

CASE FOR BIGRAM (CONTD..)

```
  // Pick the best, if there is a tie print the one that appears
first alphabetically (i.e., smallest in string comparison).


    if(bi[i][j]>best)
    {
      besti = i;
      bestj = j;
      best = bi[i][j];
    }
```

TEST CASES (1 OF 7)

Input:

1

cbb cc

1

Output:

Unigram c

TEST CASES (2 OF 7)

Input:

2

zop et wel nm az

zoaz te le wnazm

2

Output:

Bigram az

TEST CASES (3 OF 7)

Input:

1
abababababababababa
3

Output:

Trigram aba

TEST CASES (4 OF 7)

Input:

6
kjahfkasdhnfkj neravnwaevo qoegj ehg oiqjrvoirjtvkjreovj
elwrj qerj kljeqlkg let jglkjerklgltejhglk gt glt rgtrkllltr lk tgjjtv
i
,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,e,,,,,,,,,,,,,,,,,,,,,,,,
.....t
t.....
3

Output:

Trigram glk

TEST CASES (5 OF 7)

Input:

5
dr. orooji,
i have made test data for problem trigram.
i will bring it to local contest to give it to jason.
sincerely,
raymond
1

Output:

Unigram o

TEST CASES (6 OF 7)

Input:

6
jfxuezcxrodffrdxmebf.emxfuulifseewwbsi patmlwzunp
,bfwzmdtchvtuvsfrgreneurbrxreqqtwqfsiih k,eimefu
q rknnufnsykfinhiog iiv yvxzqzg.hgzyzklowi bjyvlgoxyiatlhq,hpbi.
swexewucpnnd.xv.ebwpeseucrq sfhjz tygchfdyhudb rxcuvy.jq.xbgqd.wdeedtg
niccfg
latjggrwffcs,nmsmkrctxujgjo.mmb,h,.bmu,mtvdravg.th cbs
2

Output:

Bigram at

TEST CASES (7 OF 7)

Input:

```
26
.zzzzzzzzzzzzzzzzzzzzzzzzzz
 yyyyyyyyyyyyyyyyyyyyyyyyyy
,xxxxxxxxxxxxxxxxxxxxxxxxx
 wwwwwwwwwwwwwwwwwwwwwwwwww
vvvvvvvvvvvvvvvvvvvvv
uuuuuuuuuuuuuuuuuuuuu
ttttttttttttttttttt
.sssssssssssssssssss
rrrrrrrrrrrrrrrrr
qqqqqqqqqqqqqqqqq
ppppppppppppppppp
ooooooooooooooo
nnnnnnnnnnnnnn
mmmmmmmmmmmmmm
lllllllllllll
kkkkkkkkkkkk
jjjjjjjjjjj
iiiiiiiiii
hhhhhhhhh
ggggggggg
fffffff
eeeeeee
dddddd
ccccc
bbbb
aaa
3
```

Output:

Trigram zzz