ACCESS
Substitution ciphers
Notes for Tuesday June 12

1) We will play with the Caeser shift code at Professor Carlson’s Web page, go to
http://www.math.utah.edu/~carlson, then follow the links to Undergraduate Colloquia/Cryptography.
We could break his message by hand too, since there are only 26 possible decodings of a Ceaser-shifted
message.

Enter the decoded message here:

We will discuss the truth of decoded message! It’s proof is related to the scientific method! Honest!

2) Following the conventions in "The Code Book", we will use upper case letters for encrypted
messages and lower case letters for decoded versions. So, assuming a message of upper case letters is
encrypted via a substitution cipher, how many ways could we have encrypted it? (It sure seems like a
lot.)

3) Happily, despite this large number of possibilities, substitution ciphers relatively easy to break. That
is one reason that solving cryptograms is a popular past-time. On a less happy note, Mary Queen of
Scots also found this out, even though her coding method was a little more sophisticated than a
substitution cipher. The attack method for such codes is to use language clues: mainly frequency
analysis of individual letters and of repeating multi-letter patterns (possible words). We could do this
work by hand, but the computer was made to help us avoid such things. In fact, some people say that the
idea of modern computers originated with one of history’s most famous cryptographers, Alan Turing,
see page 167 of "The Code Book." He was one of the primary people involved in the successful
struggle to break the unbelievably complicated German "Enigma" code during World War II.

Here are some Maple procedures which will help us carry out frequency analysis. We will also use
the letter frequency table on page 19 of "The Code Book". Let’s use the following message to test our
procedures on. Notice Maple requires you to enclose letters in quotes when they are representing text as
opposed to mathematical variables.

> MSG1:="WHAT DO YOU MEAN THIS MESSAGE IS ENCODED. IT SURE DOESN’T
LOOK TOO HARD TO READ TO ME. IT WOULD HAVE BEEN A LOT HARDER IF
YOU HAD ACTUALLY CHANGED SOME OF THE LETTERS."

> MSG1 := "WHAT DO YOU MEAN THIS MESSAGE IS ENCODED. IT SURE DOESN’T
LOOK TOO HARD TO READ TO ME. IT WOULD HAVE BEEN A LOT HARDER IF
YOU HAD ACTUALLY CHANGED SOME OF THE LETTERS."

4) Single letter frequency, upper case letters, by percent of total letters:


count1 := proc(Z)       # Z is a string of characters
    local C,            # Character being processed
    counts,       # Table of character counts
    L,            # Length of Z
    i;            # index
    for C from "A" to "Z" do counts[C] := 0 od; # initialize counts array
    L:=length(Z);      # L is the number of characters in Z
    # Get character count:
    for i from 1 to L   do
        C:=Z[i];
        if C >= "A" and C <= "Z" then
            counts[C] := counts[C] + 1
        fi
    od;
    # Display a table of the counts.
    for C from "A" to "Z" do
        print(C,evalf(100*counts[C]/L));
    od;
end:

5) 2 letter word counts - actual words

count2 := proc(Z)       # written assuming message has kept punctuation, to count two letter words
    local C1,C4,      # non-letter characters
    C2,C3,      # letter characters
    i,          # index in list
    counts,     # Table of word counts
    L;          # List length
    # Clear the table of character counts.
    for C2 from "A" to "Z" do
        for C3 from "A" to "Z" do
            counts[C2,C3] := 0 od; od;
    L:=length(Z);
    for i  from 1 to L-4 do
        C1:=Z[i];
        C2:=Z[i+1];
        C3:=Z[i+2];
C4:=Z[i+3];  
   if    (C1 < "A" or C1 > "Z") #C1 not a letter
    and (C2="A" and C2="Z") #C2 a letter
    and (C3="A" and C3="Z") #C3 a letter
    and (C4<"A" or C4">"Z") #C4 not a letter
    then
    counts[C2,C3]:=counts[C2,C3]+1;
    fi;
   od;
   # Display a table of the counts, if they have at least
   # one occurrence:
   for C2 from "A" to "Z" do
    for C3 from "A" to "Z" do
     if counts[C2,C3]>=1
     then print(cat(C2,C3),counts[C2,C3]);
     #cat is the concatenate command
     fi;
     od;od;
end:
>
Test count2 on our message

6) 3 letter words

[ > count3 := proc(Z)         #three letter words
    local   C1, C5,        #non-letters
   C2,C3,C4,      #letters
   i,            #index in list
   counts,       #Table of character counts
   L;            #List length
    # Clear the table of character counts.
    for C2 from "A" to "Z" do
    for C3 from "A" to "Z" do
    for C4 from "A" to "Z" do
    counts[C2,C3,C4] := 0 od;od;od;
L:=length(Z);

    for i  from 1 to L-4 do
    C1:=Z[i];
    C2:=Z[i+1];
    C3:=Z[i+2];
    C4:=Z[i+3];
    C5:=Z[i+4];
    if    (C1<"A" or C1">"Z")
    and (C2 >= "A" and C2 <= "Z")
    and (C3="A" and C3="Z")
    then


and (C4>="A" and C4<="Z")
and (C5<"A" or C5>"Z")
then
counts[C2,C3,C4]:=counts[C2,C3,C4]+1;
fi;
od;
# Display a words with non-zero counts
for C2 from "A" to "Z" do
for C3 from "A" to "Z" do
for C4 from "A" to "Z" do
if counts[C2,C3,C4]>=1
then print(cat(C2,C3,C4),counts[C2,C3,C4]);
  #cat is the concatenate command
  fi;
  od;
  od;
od;
end:

Test it!

7) YOUR GROUP ASSIGNMENT FOR TODAY: Decode the following message. Create a Microsoft Word document in which you include both the coded and decoded message, and also include the decoding key. Email your document to me, korevaar@math.utah.edu, as an attachment. Make sure you list your group members at the top of your document. Hint: You may want to see Appendix B, page 391 of "The Code Book."

> MSG2:="SDDUNN HN S NDPGCSMNPHJ QPSQ GOOUMN SJJMGYHESQUCW QXUFQW XGEUF TFHKTU UYJUMHUFDUN SFR GJGMOQTFHQHUN HF NDHUFDU SFR QUDPFGCGLW. SF SDDUNN NDPGCSMNPHJ MUDHJHUFQ NJUFRN QPU NTEEUM GO PUM OMUNPESF WUSM XHQP NGEU GO QPU IMHLPOQIJQ XGEUF UFQUMHFL QPU TFHUMNHFQW GO TQSP. HF S NJUDHSCCW RUNHLFUR DGTMNQ QSTLPQ IW NGEU GO QPU TFHZUMNHQW`N EGNQ UFQPTNHSQHD SFR DUCUIMSQUR JMGOUNINGMN, NPU HN HFQHDRUQ RG PSFRN-GF UYJUMHUFDU XHQP MUSC-XGMCR JMGICUEN QPMGTLQ HFNQMTDQHGF, CSIGMWSQGMW XGMB SFR QUSE XGMB GF SNNHFLFUR JMGAUDQN. QPU DCSNN HN NJUDHSCCW RUNHLFUR RG LHZU JSMQHDHJSFQN FGQ GFCW ZSCTSICU CSIGMWSQGMW UYJUMHUFDU, ITQ SCNG S QSNQU GO USDP GO QPU ESAGM NDHUFDUN.";