Lab 9 (10 points)

1. Write a program that uses an array to count and report the frequency of occurrence of the letters in the following text (you can code this into your program as a String):

FOXRTY YEAWXASCOIKWYATY AITMYR YAFP NXYTSCOI AT POIYTSWB AIH FOVKWYSYWB AT KOTTCQWY BOXR AITMYRT MCWW PYWK VY CVKROEY VB SYAFPCIG AIH SPY FOXRTY FXRRCFXWXV BOXR RYTKOITYT ARY AIOIBVOXT AIH C MCWW IOS TYY SPYV XISCW AUSYR UCIAW GRAHYT PAEY QYYI GCEYIPOM MOXWH BOX RASY BOXR WYEYW OU LIOMWYHGY AIH YDKYRCYIFY OU SPY TXQJYFS VASSYR QYUORY SALCIG SPCT FOXRTY CY SOSAW QYGCIYR TOVYMPAS YDKYRCYIFYH EYRB YDKYRCYIFYH YDKYRPOM MOXWH BOX RASY BOXR WYEYW OU LIOMWYHGY AIH YDKYRCYIFY OU SPY TXQJYFS VASSYR AUSYR SALCIG SPCT FOXRTYMPAS MAT SPY VOTS CVKORSAIS SPCIG BOX WYARIYH CI SPCT FOXRTYMPAS HCH BOX WCLY QYTS AQQXSP SPY FOXRTYMPAS HCH BOX WCLY WYATS AQQXSP SPY FOXRTYMPAS AFSCSECSCYT CY WYFSXRYT WATQ KROGRAVT SYTST YSF MYRY PYWKUXW SO BOXR WYARICGMPCFP AFSCSECSCYT MYRY IOS PYWKUXW SO BOXPOM FAI SPY FOXRTY QY CVKROEYHAAHHCSCOIAW FOVYIIST

The text above was originally written in English; line breaks and punctuation have been removed, and the text was then encrypted with a simple substitution cipher. Using the letter frequency document provided, try decrypting the text. Keep in mind that some letters might be substitutes for themselves. Turn in a text file with your decryption.

Extra credit options for part 1: up to 5 points each

   Extra credit option a: Write a program to decrypt the text

   Extra credit option b: Write a program that encrypts a string of text using a cipher of your own design.

2. Write a program that creates an array of 1000 random numbers in the range 1 – 5000. Then ask the user for a value between and 5000, and search the array until you either find the value or determine that the value is not in the array. Print a message indicating whether or not the value was found, and how many numbers you had to look at before you found the value (or determined that it was not there).

Extra credit option for part 2: 5 points

Using either of the sorting algorithms presented in class, sort your array, then implement the following search algorithm to look for a value requested by the user:

   • Look at the middle value of the array; if the value is found, you’re done. Otherwise, the value is either greater or less than this value
   • If the search didn’t end in the step above, search the half of the array in which the value must be found (if the value is less than the middle value, search the half from the beginning up to midpoint-1; if the value is greater than the middle value, search from midpoint+1 up to length – 1) using the same method – that is, examine the middle value.
   • Continue the process of dividing the search area in half and examining the midpoint until either the value is found, or you are down to a search area of length 1 which does not contain the value.

As with the original search method, your program should report whether or not the value was found, and how many values you had to examine to determine the outcome.