Trevor Fune

CSC 242

Professor Guha

28 Aug 2024

**Assignment 5 Pseudocode**

Spell Check Pseudocode

This pseudocode describes how to check the spelling of all words in a file utilizing a dictionary file and outputs all words not in the dictionary.

Declare input-stream variable named dictionary

Open the dictionary file in dictionary

If file doesn’t open:

Output “Unable to open dictionary file”

Define a vector of strings called words

Define a string called word

For each word in the dictionary file:

Append the word to the words vector

Close the dictionary file

Declare input-stream variable named words\_to\_check

Open the file to be checked (“WordsToCheck.txt”) in words\_to\_check

If file doesn’t open:

Output “Unable to open file to check spelling.”

For each word in words\_to\_check:

If the word is not contained in the words vector

Print the word

Country Populations Pseudocode

This pseudocode describes how to read country population data from a file and use it to calculate total population.

Create void function named process\_line:

Purpose: processes line containing country name and population into separate variables

Parameter named line: the string being processed

Parameter named country: name of country in line

Parameter named population: population of country

Create istringstream named strm that contains line

Create char variable named ch

For every character in strm

If character is not white space

Append character to country

If character is a white space

Skip to the next non-white space character

If character is not a digit

Append a space to country

Append character to country

Else

Unget the character

Place the whole number into population

Break out of function

Main Function:

Declare input-stream variable named input\_file

Open the world population file (“worldpop.txt”) in input\_file

If the file doesn’t open:

Output “Unable to open world population file.”

Create a string variable named line

Create a floating-point variable named total\_population

For every line in input\_file:

Create a string variable named country

Create an integer variable named population

Make function call to process\_line (line, country, population)

If country is not the European Union

Add population to total\_population

Close the file

Output the total population excluding the European Union

Keyword Cipher Pseudocode

This pseudocode describes how to encrypt or decrypt an input file into an output file using a keyword cipher.

Create constant string variable named ALPHABET with all uppercase letters

Create string function named create\_encrypted\_alphabet

Purpose: Create encrypted alphabet using keyword and remaining letters in reverse order

Parameter named keyword: keyword used for cipher

Return: a string containing the encrypted alphabet

Create string named encrypted\_alphabet

Create Boolean array of size 26 named used to hold used letters

For every character in keyword

Convert character to uppercase

If character is a letter and is not in the used array

Append character to encrypted\_alphabet

Update the element representing the current character in used to true

Create string named remaining\_letters

For every character in ALPHABET

If the character hasn’t been used yet

Append the character to remaining\_letters

Reverse remaining\_letters

Append remaining\_letters to encrypted\_alphabet

Return encrypted\_alphabet

Create void function named encrypt\_file

Purpose: Encrypt a file using an encrypted alphabet

Parameter named input: ifstream of input file

Parameter named output: ofstream of output file

Parameter named encrypted\_alphabet: string containing encrypted alphabet

Create string named line

For every line in input

Create string named encrypted\_line

For every character in line

If character is a letter

Convert character to uppercase

Find position of character in ALPHABET

Append encrypted\_alphabet[pos] to encrypted\_line

Else

Append character to encrypted\_line

Send encrypted\_line to output

Create void function named decrypt\_file

Purpose: Decrypt a file using an encrypted alphabet

Parameter named input: ifstream of input file

Parameter named output: ofstream of output file

Parameter named encrypted\_alphabet: string containing encrypted alphabet

Create string named line

For every line in input

Create string named decrypted\_line

For every character in line

If character is a letter

Convert character to uppercase

Find position of character in encrypted\_alphabet

Append ALPHABET[pos] to decrypted\_line

Else

Append character to decrypted\_line

Send decrypted\_line to output

Main Function:

If number of command line arguments does not equal 5

Output: “Usage: keywordcipher.cpp <-e|-d> -k<keyword> <input\_file> <output\_file>

Exit out of program

Create string named mode and set equal to argv[1]

Create Boolean variable named encryption\_mode and set equal to true

If mode is “-e”

Keep encryption\_mode equal to true

Else if mode is “-d”

Set encryption\_mode to false

Else

Output “Invalid mode: use ‘-e’ or ‘d’”

Exit out of program

Create string named keyword\_arg and set equal to argv[2]

Create string named keyword and set equal to substring of argv[2] after “-k”

If keyword\_arg doesn’t have “-k” or a keyword

Output “Invalid keyword: use -k<keyword>

Exit out of program

Create ifstream named in\_file

Open argv[3] in in\_file

If in\_file fails to open

Output “Unable to open input file”

Exit out of program

Create ofstream named out\_file

Open argv[4] in out\_file

If out\_file fails to open

Output “Unable to open output file”

Exit out of program

Create string named encrypted\_alphabet using the function create\_encrypted\_alphabet

If encryption\_mode is true

Use the function encrypt\_file

If encryption\_mode is false

Use the function decrypt\_file

Playfair Cipher Pseudocode

This program describes how to encrypt or decrypt an input file into an output file using a playfair cipher.

Create constant string variable ALPHABET with all upper case letters

Create constant integer variable SIZE equal to 5

Create string function named create\_encrypted\_alphabet

Purpose: Create encrypted alphabet using keyword and remaining letters excluding ‘J’

Parameter named keyword: keyword used for playfair cipher

Return: a string containing the encrypted alphabet

Create string named encrypted\_alphabet

Create Boolean array that holds used letters

For every character in keyword

Convert character to uppercase

If character is ‘J’ convert it to ‘I’

If character is a letter and is not in the used array

Append character to encrypted\_alphabet

Update the element representing the current character in used to true

For every character in ALPHABET

If the character hasn’t been used yet and is not ‘J’

Append the character to encrypted\_alphabet

Return encrypted\_alphabet

Create vector function named create\_playfair\_matrix

Purpose: Creates 5x5 vector matrix and fills with encrypted alphabet

Parameter named encrypted\_alphabet: string containing encrypted alphabet

Return: 5x5 matrix filled with encrypted alphabet

Initialize a 5x5 vector named matrix

Create integer variable named index

For every row in matrix

For every column in row

Update matrix[i][j] to corresponding character in encrypted alphabet

Increment index

Return matrix

Create void function named find\_position

Purpose: Finds position of character within matrix

Parameter named ch: the character we are finding the position of

Parameter named matrix: the 5x5 matrix that contains encrypted alphabet

Parameter named row: the row the character is found in

Parameter named col: the column the character is found in

For every row in matrix

For every column in matrix

If we find the character we are looking for

Set row equal to i

Set col equal to j

Return

Create string function named encrypt\_pair

Purpose: encrypts/decrypts a pair of letters using playfair\_matrix

Parameter named input\_pair: Pair of letters being encrypted/decrypted

Parameter named matrix: the 5x5 matrix that contains encrypted alphabet

Return: a string that contains encrypted pair of letters

Create char variable named ch1 and set equal to first letter

Create char variable named ch2 and set equal to second letter

If ch1 or ch2 are not letters

Return original pair of characters

Create string variable named encrypted\_pair

Create integer variables that hold row and column of each character

Use find\_position function for ch1 to update row1 and col1

Use find\_position function for ch2 to update row2 and col2

If the characters are in the same row or same column

Reverse input\_pair

Set encrypted\_pair equal to input pair

Return encrypted\_pair

Else (characters form a rectangle)

Change ch1 to opposite corner in same row

Change ch2 to opposite corner in same row

Append ch1 to encrypted\_pair

Append ch2 to encrypted\_pair

Return encrypted\_pair

Create string function named process\_line

Purpose: converts line to uppercase and removes any ‘J’ characters

Parameter named line: line that is being processed

Return: a string with letters in uppercase and no ‘J’

Create string named processed\_line

For every character in line

If character is a letter

Convert character to uppercase

If character is equal to ‘J’

Set character equal to ‘I’

Append character to processed\_line

Else (character is not ‘J’)

Append character to processed\_line

Else (if character is not a letter)

Append character to processed\_line

Return processed\_line

Create string function named encrypt\_line

Purpose: encrypts/decrypts line using a keyword

Parameter named keyword: keyword used to create playfair cipher

Parameter named input\_line: the line being encrypted/decrypted

Return: a string containing the encrypted line

Create encrypted alphabet using the function create\_encrypted\_alphabet

Create matrix using the function create\_playfair\_matrix

Process the input line using the function process\_line

Create string named encrypted\_line

For every 2 characters in line:

Extract the 2 characters into variable named input\_pair

Encrypt the pair using the function encrypt\_pair

Append encrypted\_pair to encrypted\_line

Return encrypted\_line

Main Function:

Create string variable named keyword

Create string variable named input\_text

Prompt user to enter keyword

Store user input into keyword

Clear the newline character left after cin >> keyword

Prompt user to enter text that will be encrypted/decrypted

Store user input into input\_text

Encrypt/decrypt the input text using the function encrypt\_line and keyword

Output the encrypted\_text