**OPEN ENDED PROBLEM**

RABIN-KARP PATTERN FINDER & REPLACER

**Code:**

import java.util.\*;

import java.io.\*;

//Replacer class will Replacet the existing pattern with new pattern in the text file

class Replacer {

void modifyFile(String filePath, String oldString, String newString) {

File fileToBeModified = new File(filePath);

String oldContent = "";

BufferedReader reader = null;

FileWriter writer = null;

try {

reader = new BufferedReader(new FileReader(fileToBeModified));

// Reading all the lines of input text file into oldContent

String line = reader.readLine();

while (line != null) {

oldContent = oldContent + line + System.lineSeparator();

line = reader.readLine();

}

// Replacing oldString with newString in the oldContent

String newContent = oldContent.replaceAll(oldString, newString);

// Rewriting the input text file with newContent

writer = new FileWriter(fileToBeModified);

writer.write(newContent);

} catch (IOException e) {

e.printStackTrace();

} finally {

try {

// Closing the resources

reader.close();

writer.close();

} catch (IOException e) {

e.printStackTrace();

}

}

}

}

class RabinKarp {

// d is the number of characters in the input alphabet

final int d = 256;

void search(String pat, String txt, int q) {

int M = pat.length();

int N = txt.length();

int i, j;

int p = 0; // hash value for pattern

int t = 0; // hash value for txt

int h = 1;

// The value of h would be "pow(d, M-1)%q"

for (i = 0; i < M - 1; i++)

h = (h \* d) % q;

// Calculate the hash value of pattern and first

// window of text

for (i = 0; i < M; i++) {

p = (d \* p + pat.charAt(i)) % q;

t = (d \* t + txt.charAt(i)) % q;

}

// Slide the pattern over text one by one

for (i = 0; i <= N - M; i++) {

if (p == t) {

/\* Check for characters one by one \*/

for (j = 0; j < M; j++) {

if (txt.charAt(i + j) != pat.charAt(j))

break;

}

// if p == t and pat[0...M-1] = txt[i, i+1, ...i+M-1]

if (j == M)

System.out.println("Pattern found at index " + i);

}

if (i < N - M) {

t = (d \* (t - txt.charAt(i) \* h) + txt.charAt(i + M)) % q;

if (t < 0)

t = (t + q);

}

}

}

}

class AdaOep {

public static void main(String[] args) {

RabinKarp m = new RabinKarp();

Replacer r = new Replacer();

String pat = "";

int line = 0;

int ch;

String replacement;

int q = 101; // A prime number

Scanner sc = new Scanner(System.in);

System.out.println("\nEnter The Pattern you are looking for : ");

pat = sc.nextLine();

System.out.println("new replacement : ");

replacement = sc.nextLine();

do {

System.out.println("\nEnter a choice (1-4):");

System.out.println("1. Find a pattern");

System.out.println("2. Replace a pattern");

System.out.println("3. Do both 1 & 2.");

System.out.println("4. Exit\n");

System.out.println("\n Choice: ");

ch = sc.nextInt();

switch (ch) {

case 1:

try {

FileReader fr = new FileReader("File.txt");

BufferedReader br = new BufferedReader(fr);

while (br.read() != -1) {

line++;

System.out.println("line no: " + line);

m.search(pat, br.readLine(), q);

}

} catch (IOException e) {

System.out.println(e);

}

break;

case 2:

r.modifyFile("File.txt", pat, replacement);

System.out.println("\nDone Replacment !!\n");

break;

case 3:

try {

FileReader fr = new FileReader("File.txt");

BufferedReader br = new BufferedReader(fr);

while (br.read() != -1) {

line++;

System.out.println("line no: " + line);

m.search(pat, br.readLine(), q);

}

} catch (IOException e) {

System.out.println(e);

}

// replacing

r.modifyFile("File.txt", pat, replacement);

System.out.println("\nDone Replacment !!\n");

break;

case 4:

System.exit(0);

}

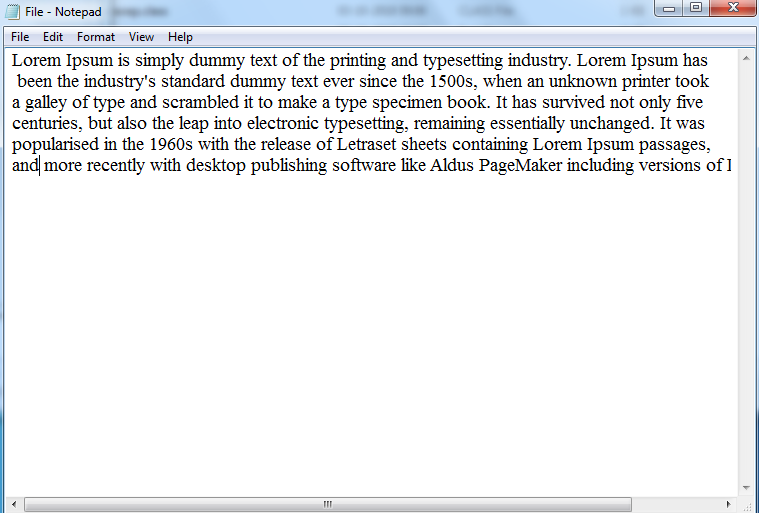
} while (ch != 4);

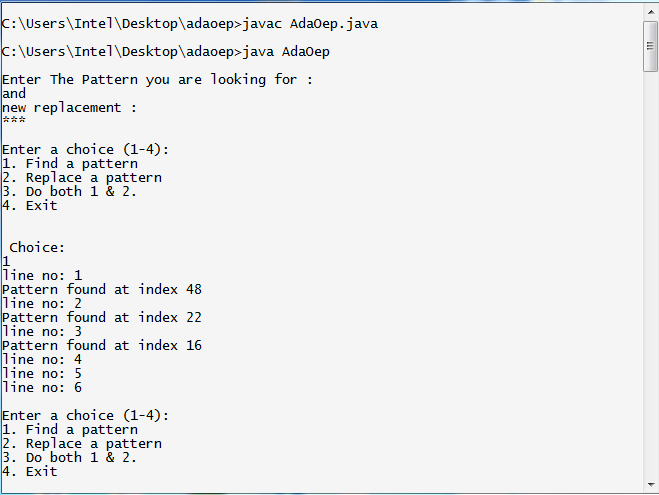
}

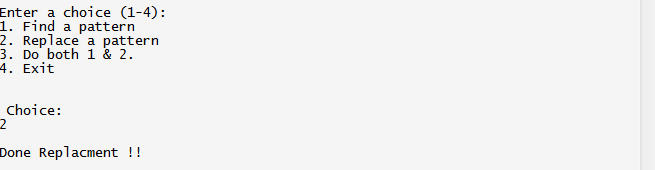
}

OUTPUT

Initial view of Text file







After Replacement

