STRING MATCHING ASSIGNMENT

H. D. Sanduni Aaloka

20000014

2020/CS/001

University of Colombo School of Computing

Contents

Problem 2

Why Boyer Moore is used? 3

Code Explanation 4

Preprocessing function 4

boyerMoore function 5

Test Cases 7

Case 1 7

Case 2 8

Case 3 8

Problem

Write a program that performs the following tasks using a familiar programming language

(Java/C/C++/Python):

1) Prompts the user to enter a pattern P to search for, for example “Architecture”.

2) It then proceeds by reading the module catalogue stored in the file ‘modules.txt’, where each line contains the title of one module.

3) For every line in the file, the program checks whether it contains the search pattern P. If yes, it prints out this line. For example, the line corresponding to the `ARC1015` module contains the string “Architecture” and should be printed. Note: string matching should be case-insensitive, i.e., “Architecture” contains the search string “architecture”.

4) When all lines have been processed, the program prints the number of found matches.

Note: you do not have to follow exactly the same input and output format as shown in the examples. You are also encouraged to experiment by adding other useful features for manipulating the information provided in the module catalogue.

Why Boyer Moore is used?

> Results: The **Boyer-Moore-Horspool** algorithm achieves the best overall results when used with medical texts. This algorithm usually performs at least twice as fast as the other algorithms tested. Conclusion: The time performance of exact string pattern matching can be greatly improved if an efficient algorithm is used.

The difference between Boyer Moore and other algorithms is mainly because Boyer Moore algorithm search the string backwards. And if the comparison of the last character doesn't match the pattern can skip the full length. The worst-case running time of Boyer Moore is O(mn). The best-case time complexity is O(m/n). Practically different string-matching algorithms are suitable at different situations irrespective of its time complexities. If for example KMP algorithm and Boyer Moore algorithm are compared, KMP algorithm works well in situations where the alphabet is not very large and Boyer Moore works well with somewhat longer patterns. Also, Boyer Moore can be sublinear.

Code Explanation

The code is written in python language. The Boyer Moore Horspool algorithm is used by using two functions as boyerMoore() and preprocessing().

Preprocessing function

```python
def preprocessing(pat, size):
    badchar = [-1] * NO_OF_CHARS
    for i in range(size):
        badchar[ord(pat[i])] = i
    return badchar
```

The preprocessing function in Boyer Moore algorithm should result the HpBc table. In this for each character in the alphabet should have a value which is equal to pattern-1 from its rightmost character. The last symbol is not taken into account if this calculation. If only the preprocessing function is used the below array will result.

```python
NO_OF_CHARS = 256
txt = "I am Sanduni Aloka"


def preprocessing(pat, size):
    badchar = [-1] * NO_OF_CHARS
    for i in range(size):
        badchar[ord(pat[i])] = i
    return badchar

badchar = preprocessing(txt, len(txt))
print(badchar)
```

```
PROBLEMS    OUTPUT    TERMINAL    JUPYTER

TERMINAL
PS E:\uni-2nd year 1st sem\SCS2201 - DSA III\Assignment 1> python -u "e:\uni-2nd year 1st sem\SCS2201 - DSA III\Assignment 1\test.py"
[-1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, 12, -1, -1, -1, -1,
 -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, 13, -1, -1, -1, -1, -1, -1, -1, 0,
-1, -1, -1, -1, -1, -1, -1, -1, -1, 5, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, 17, -1, -1, 8, -1, -1, -1, -1, 11, -1, 16, 14, 3, 10, 15,
 -1, -1, -1, -1, -1, 9, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1,
-1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1,
-1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1,
-1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1]
PS E:\uni-2nd year 1st sem\SCS2201 - DSA III\Assignment 1> 
```

NO_OF_CHARS is a variable which is equal to the number 256. Initially the array is initialized with -1. Then for each character in the pattern is the position of it in

the pattern is inserted at its index. The index of each character is taken from its Unicode value as ord() function in python returns the Unicode from a given character. At the end of the loop each character in the alphabet will be initialized with its rightmost position in the pattern.

boyerMoore function

```
8   def boyerMoore(txt, pat, string):
9       txt = txt[8:]
10      n = len(txt)
11      m = len(pat)
12
13      badchar = preprocessing(pat, m)
14
15      s = 0
16      while(s <= n-m):
17          j = m-1
18          while(j>=0 and pat[j] == txt[s+j]):
19              j -=  1
20          if(j<0):
21              global count
22              count += 1
23              #print("pattern occur shift ", s)
24              print(string)
25              s += (m-badchar[ord(txt[s+m])] if s+m<n else 1)
26          else:
27              s += max(1, j-badchar[ord(txt[s+j])])
28
```

The text and pattern are inserted as arguments into the boyerMoore function. In the module.txt a single line will be for example APL1001 Alternative Practice Histories where “APL1001” is the code of the module and “Alternative Practice Histories” is the name of the module.

```
with open('modules.txt') as openfileobject:
    for line in openfileobject:
        boyerMoore(line.lower(), pat.lower(), line)
```

As txt lower case of each line is included and as line each line of txt as it is included. The argument string is used because we need to print the line of the module in boyerMoore function if the search is successful. Both txt(text) and pat(pattern) is included as lowercase is because the search should be case sensitive.

Ex: “Architecture” contains the search string “architecture”.

The variable s denotes the shift. Then txt is truncated to get only the name of the module. Thus, the pattern searched will only be searched through the name of the module. j variable is the indicator for the pattern and j starts from the end of the pattern. That’s because in Boyer Moore searching is done backwards. If the character at position j is matching with the corresponding character of the text and if it is not the first character of the pattern the variable j is decremented by 1. Then that character is compared and so on.

If the first character of the pattern is also compared then the index j will become -1.Thus, if j==-1 then there is a match. The count variable which stores the number of times the pattern is found is incremented by 1 and the text is printed. The argument string is used to printing as txt will be all lower case and also it is trunacated.

Then the equation,

```
    s += (m-badchar[ord(txt[s+m])] if s+m<n else 1)
```

is used to get the next shift.

As s is the shift and m is the length of the pattern s+m gives the last index of the text where the pattern is matched. Badchar[ord(txt[s+m])] gets the value of that particular character in the text from the HpBc table. The value that comes from the badchar array is the index of the final encounter of that particular character in the

pattern. Thus substarcting it by m(length of the pattern) gives us the shift where the badcharacter is matched to the pattern.

Test Cases

Case 1

```
PS E:\uni-2nd year 1st sem\SCS2201 - DSA III\Assignment 1> python -u "e:\uni-2nd year 1st sem\SCS2201 - DSA III\Assignment 1\assignment.py"
Enter a search string: Architecture
APL2005 Twentieth Century Architecture

APL3004 Chinese politics culture and urban development architecture planning and visual culture

ARC1015 Introduction to Architecture

ARC2024 About Architecture Cities Cultures and Space

ARC4020 Architecture in Practice

ARC8051 Tools for Thinking About Architecture

ARC8053 Dissertation in Architecture A

ARC8061 Architecture  Construction Process  Management

ARC8062 Dissertation in Architecture B

ARC8116 Architecture and Cities specialist studio

EEE8126 Embedded Systems Architecture and Programming

MAR1012 Naval Architecture

MAR2017 Further Naval Architecture

MAR2108 Naval Architecture II

MAR3044 Project and Report in Naval Architecture
```

.....

```
INU0523 English for Academic Purposes for Foundation Architecture 40 Credits Version

INU0524 Architectural Communication for Foundation Architecture

INU0525 Architecture Culture  History

INU1109 Architectural Communication for Intl Year One Architecture

INU1116 Architecture Technology

INU1509 Architectural Communication for Intl Year One Architecture

INU1516 Architecture Technology

INU3103 English for Academic Purposes Graduate Diploma for Architecture

INU3116 Professional Studies Graduate Diploma for Architecture

INU3119 Architecture Culture  History Graduate Diploma for Architecture

INU3119 Architecture Culture  History Graduate Diploma for Architecture

INU3503 English for Academic Purposes Graduate Diploma for Architecture January Intake

INU3516 Professional Studies Graduate Diploma for Architecture January Intake

INU3519 Architecture Culture  History Graduate Diploma for Architecture January Intake

INU3519 Architecture Culture  History Graduate Diploma for Architecture January Intake

Number of matches:  42
```

Case 2

```
Enter a search string: Microprocessor
EEE2007 Computer Systems and Microprocessors

EEE2206 Computer Systems and Microprocessors

EEE8022 Microprocessor Systems

Number of matches:  3
```

Case 3

To prove the program is case sensitive

```
Enter a search string: Arc
APL2005 Twentieth Century Architecture

APL2007 Visual and Creative Practice Research Skills

APL3004 Chinese politics culture and urban development architecture planning and visual culture

ARC1007 Architectural Design 11

ARC1015 Introduction to Architecture

ARC1016 Architectural Representation

ARC2001 Architectural Design 21

ARC2020 Dissertation Studies and Research Methods

ARC2024 About Architecture Cities Cultures and Space

ARC3001 Architectural Design 31

ARC4020 Architecture in Practice

ARC8022 Research Project  Urban Energy

ARC8051 Tools for Thinking About Architecture
```

.....

```
PHY8003 Research Skills II Project Proposal

PHY8005 Computational Research Skills in Physics

PHY8007 Research Project

PHY8033 Theoretical Research Project Inactive

FMS8360 Researching Film Skills and Methods

FMS9001 Film Studies MLitt Research Assignments

SML8009 Research Methods in Translating and Interpreting

SML9001 Modern Languages MLit Research Assignements

SML9003 Translation Studies MLitt Research Assignments

Number of matches:  264
```