Department of Information Systems and Technologies

CTIS 152 – Data Structures and Algorithms

Fall 2024 - 2025

Lab Guide #13 – Week 9-2

OBJECTIVE : Recursion

Instructor : Burcu LIMAN
Assistant : Berk ÖNDER, Engin Zafer KIRAÇBEDEL

Q1. **a)** Write a C program that counts the number of occurrences of the given character in a sentence **recursively**.

Example Run:

```
Enter a string: Round Robin rode a really rusty roller coaster.
Enter a character: r

Round Robin rode a really rusty roller coaster.
The number of 'r' is 8
```

Project_name: LG13_Q1a
File_name: Q1a.cpp

b) Write a C program that counts the number of occurrences of the given string in a sentence **recursively**.

Example Run:

```
Enter a sentence: How much wood would a woodchuck chuck if a woodchuck could chuck wood?

Enter a word to search in the sentence: wood
The word -wood- occurred 4 times in the sentence
```

Project_name: LG13_Q1b
File_name: Q1b.cpp

Q2. **a)** Write a **recursive** function;

- **`printDigits:`** prints the digits of a number.

Write a C program which gets a number input from the user and calls the **`printDigits`** function.

Example Run:

```
Enter a number: 35154321
Digits of the number 35154321 are:
    1    2    3    4    5    1    5    3
```

Project_name: LG13_Q2a
File_name: Q2a.cpp

b) Modify your solution from **Q2a.cpp** so that the recursive function prints the digits in the correct order.

Example Run:

```
Enter a number: 35154321
Digits of the number 35154321 are:
    3    5    1    5    4    3    2    1
```

Project_name: LG13_Q2b
File_name: Q2b.cpp

Q3. Write a C Program that reads the task list of a printer (task id, task order no and file name) from the file **"taskList.txt"** into a structure array with the maximum **SIZE 30**, sorts the array according to the filenames in ASCENDING order, displays the list on the screen, searches for a specified filename by the help of binary search algorithm.

Write the following functions; **readFromFile, display, recBubbleSort, recBinarySearch**

(Write the **bubble sort** and **binary search** functions recursively.)

Example run:

```
Task Id    Order No    File Name
********  **********  **********
228         1474        attendance.csv
111         2954        exam.pdf
435         2423        labguide1.doc
291         3453        labguide2.docx
213         2324        labquiz1.pdf
276         3456        labquiz2.doc

Enter a filename (END for exit): labguide1.doc
Task Id    Order No    File Name
********  **********  **********
435         2423        labguide1.doc

Enter a filename (END for exit): labguide2.doc
NOT FOUND

Enter a filename (END for exit): labguide2.docx
Task Id    Order No    File Name
********  **********  **********
291         3453        labguide2.docx

Enter a filename (END for exit): END
```

taskList.txt

```
111 2954 exam.pdf
435 2423 labguide1.doc
291 3453 labguide2.docx
228 1474 attendance.csv
213 2324 labquiz1.pdf
276 3456 labquiz2.doc
```

Project_name: LG13_Q3
File_name: Q3.cpp

Additional Questions

AQ.

Some algorithms require nested recursion where the result of one function call is a parameter to another function call. For Example, Ackermann's function is defined as:

$$A(m,n) = \begin{cases} n+1 & \text{if } m = 0 \\ A(m-1,1) & \text{if } m > 0 \text{ and } n = 0 \\ A(m-1, A(m,n-1)) & \text{if } m > 0 \text{ and } n > 0. \end{cases}$$

Write a **recursive** function;

- **`isAckermann`** takes two integer numbers as parameters, finds and returns the result according to the given rules above.

Write a C program that gets two integer numbers from the user and computes the result of **isAckermann(m, n)**. The value of **m** and **n** both have got to be non-negative values {m, n} >= 0 with m being less than 4 (m < 4) if n is bigger than 0; seriously, don't put 4 into m if n > 0.

Example Run #1:

```
Enter the value of m: 3
Enter the value of n: 7

The result is 1021
```

Example Run #2:

```
Enter the value of m:6
Enter the value of n: 5

The value of the m had to be less than 4. Exiting.
```

Project_name: LG13_AQ
File_name: AQ.cpp