

COMP 205 SYSTEMS PROGRAMMING
GROUP PROJECT REPORT

C Shell

Elif Pulukçu - 041701017

Ahsen Amil - 041701009

MEF University, Istanbul, Turkey

Fall 2019

INTRODUCTION

This section was written by both Elif Pulukçu and Ahsen Amil

The main purpose of the project is to create a custom C shell that takes user commands and performs related tasks by producing necessary system calls or C functions. In this project, 4 different functions were produced to be used in the C shell with moderate and difficult difficulty levels. Two of them (PalindromeChecker and StudentList) were written in C language, two of them (Fword and SortArray) was written in a shell script. When the C shell is started, the corresponding function is executed at the user's request. The terminal does not close unless the user wants to exit the C shell.

The user can do in the C shell:

- PalindromeChecker : find the reverse of string and check if a string has palindrome.
- StudentList : create management for add and search students in the text file.
- Fword : look up the word frequency in a file.
- SortArray : creates an array with the elements that the user wants. Sort the elements in the arrays and give the position of the desired element.

METHODS

2.1 PalindromeChecker (Intermediate C Program)

This section was written by Ahsen Amil

Algorithm Explanation

This C function is written with the purpose of finding the reverse order of a string that the user enters and check if it is Palindrome or not. To verify that, a Stack

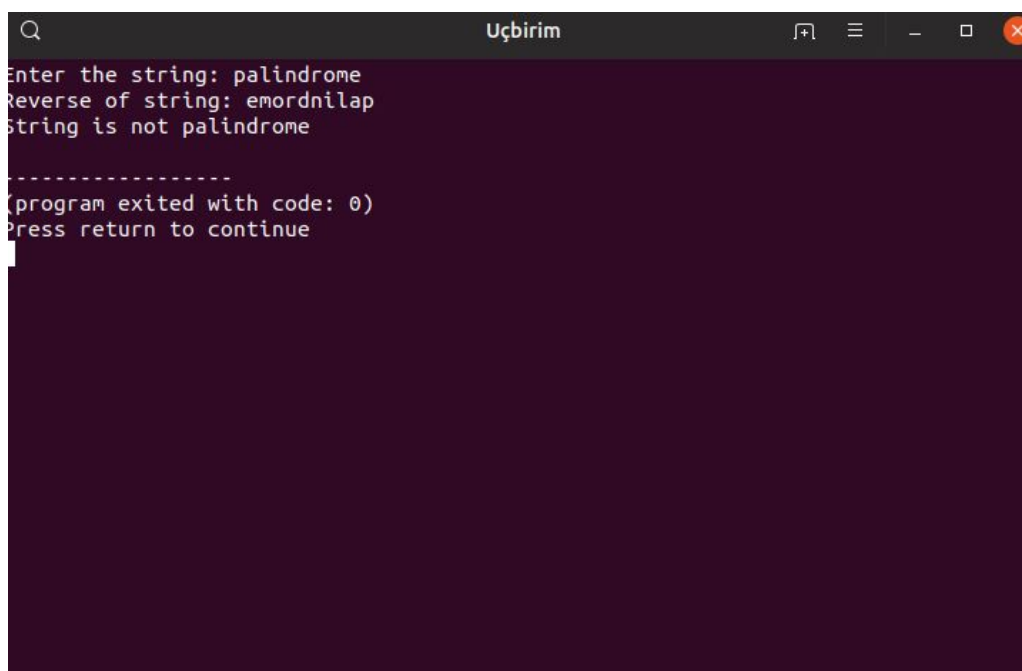
was used as a structure. The library function malloc is used to allocate a block of memory on the heap. Moreover, the size of Stack initializes as zero at the beginning.

Some functions such as isFull and isEmpty were written to check the solidity ratio of the Stack by checking the top of the Stack. Also, push an element above the top of the Stack and increase top by one or pop an element from the top of the Stack and decrease top by one.

A Stack based function Reverse was written to reverse a string. Stack's first in last out principle (FILO) is the reason why it is chosen as the structure in this function. Thence, a Stack with a capacity equal to the size of the String given by the user as an input created. If all the characters of the String are pushed to a Stack, and then all the characters popped to the Stack gives us reverse order of the characters. Because the first character of the String lastly popped out the Stack.

checkPalindrome function used to check input String is Palindrome or not. A palindrome is a word, number, phrase, or another sequence of characters which reads the same backward as forward. If the output of the reverse method equals to original String, it means that it is Palindrome.

Sample Output of The Program



```
Uçbirim
Enter the string: palindrome
Reverse of string: emordnilap
String is not palindrome
-----
(program exited with code: 0)
Press return to continue
```

Output1.Example of a String that is not Palindrome

2.2 StudentList (Complex C Program)

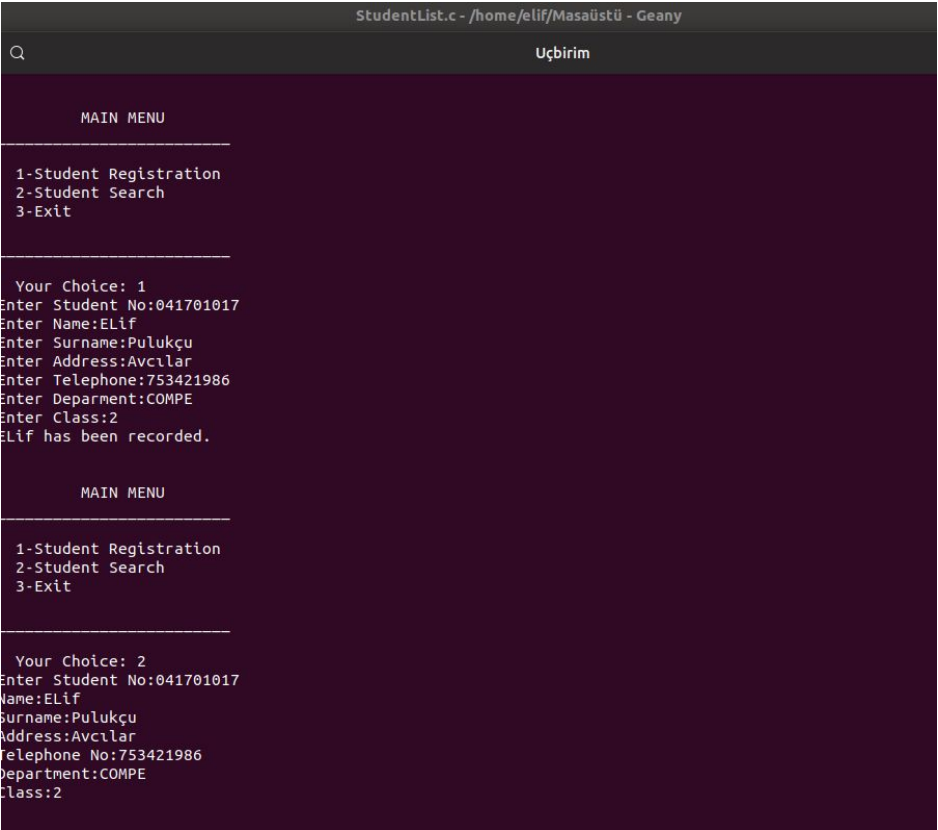
This section was written by Elif Pulukçu

Algorithm Explanation

The aim of this program is to add the students whose information such as student number, name, surname and faculty are given as input to an opened .txt file and to search by student number at any time. There is a while loop which displays a main menu to the user. In the menu 1 represents student registration, 2 represents student search and 3 represents exit. When the user presses 3 program ends.

There is a struct named Student used in this code. Student includes name, surname, address, mobileNumber, faculty, class and number data fields. Also, this program includes a hashing method that named hashing_method. This method takes students number and it creates a new code using the number and organizes the students according to these new numbers to prevent collusion.

Sample Output of The Program



```
StudentList.c - /home/elif/Masaüstü - Geany
Q Uçbirim

MAIN MENU
-----
1-Student Registration
2-Student Search
3-Exit

Your Choice: 1
Enter Student No:041701017
Enter Name:Elif
Enter Surname:Pulukçu
Enter Address:Avcılar
Enter Telephone:753421986
Enter Department:COMPE
Enter Class:2
Elif has been recorded.

MAIN MENU
-----
1-Student Registration
2-Student Search
3-Exit

Your Choice: 2
Enter Student No:041701017
Name:Elif
Surname:Pulukçu
Address:Avcılar
Telephone No:753421986
Department:COMPE
Class:2
```

Output4.Example of creation and searching a student

2.3 Freward

This section was written by Elif Pulukçu

Algorithm Explanation

The purpose of this shell script is to find out how many times the words in a text file are used. After finding the frequency of the words, sorts them from the most used word to the least used word. Firstly, checked whether the user entered the argument correctly. If the argument has not been entered, it receives an error message and is expected to enter it with the argument. When the command is entered correctly, the existence of the file to be analyzed is checked. Otherwise, the user will see an error message. When everything goes well, the 'cat' command is used to look inside the file and the 'xargs' command determines how to change the character-by-cut, byte position, delimiter-based segment, and output delimiter. In order to count not different words in the file starting with uppercase or lowercase letters, all uppercase letters are converted to lowercase with the 'tr' command. The 'sed' command lists words according to the 'dot', 'comma', and 'space' parameters between words in the file. Since more than one 'sed' command is used, the '-e' statement is added to combine the sed commands. Finally, words separated by the 'uniq' command are sorted by the 'sort' command.

```
ahsen@ahsen-VirtualBox:~/Masaüstü$ ./Freward.sh
Please enter command with filename
ahsen@ahsen-VirtualBox:~/Masaüstü$ ./Freward.sh script.sh
File "script.sh" does not exist.
ahsen@ahsen-VirtualBox:~/Masaüstü$ ./Freward.sh input.txt
The words and frequencies contained in the file
    7 hava
    7 bugün
    3 yağmurlu
    3 sıcak
    1 güzel
    1 çok
ahsen@ahsen-VirtualBox:~/Masaüstü$
```

Output5. Example of output

2.4 SortArray

This section was written by Ahsen Amil

Algorithm Explanation

Bubble Sort is a simple algorithm used to sort a particular set of n elements into an array containing n elements. Bubble Sort compares all items individually and sorts them by value. If the array has a total of n elements, the bubble sort needs to repeat this process $n-1$ times. Sequencing is accomplished by stepping all elements one by one and comparing it with the adjacent element and replacing it as necessary. If a given array needs to be sorted in ascending order, the bubble order starts by comparing the first element of the array to the second element, if the first element is larger than the second element, it replaces both items, and then proceed to compare the second and third elements, and so on.

The user is notified of the position of the desired element in the range. When searching for elements, the array is divided by 2. The desired element continues to be searched from which side of the array is divided into two. When the element is found, the relevant index is printed on the screen.

```
ahsen@ahsen-VirtualBox:~/Masaüstü$ ./SortArray.sh
Enter the limit of array:
4
Enter numbers.
1
8
45
78
Sorted array is:
1
8
45
78
Enter the element to be searched :
45
Element found at position 3
ahsen@ahsen-VirtualBox:~/Masaüstü$
```

Output6. Example of sort array

DISCUSSION

This section was written by Ahsen Amil and Elif Pulukçu

At the end of the teamwork, every two members made a considerable effort. Group members helped each other in unresolved issues so that the group members' powers of group work improved. The use of the C language with data structures, array, and pointer features has been in the developer direction for C programming. Throughout the project, we conducted research, improved our knowledge and had the chance to practice. It was a useful project to consolidate the lesson. We had a hard time finding ideas about the project. Hence it is not easy to find something that is focused on the system and represents the course content. We also had some problems with building our Shell. For example, normally, our Shell should display a command prompt to the user and wait until the user types a command, and it will continue until the user intends to exit from the Shell. However, our terminal shuts down after one command.