COMP 205
Systems Programming
Instructor: Şeniz Demir

Term Project Report

Emre ARSLAN
041701043
Ertuğrul YILMAZ
041701030

Submitted: 24.12.*2019*

MEF University

Computer Engineering Program

Introduction

While doing this project our main goal was use the knowledge that we learned during this course and practicing them. We tried to implement this learnings to programs that we can use in our daily lives. This thoughts lead us to choose these topics for our C and Bash Script functions. While coding this program, we faced with some errors and we did some researches to solve them and it improved our coding skills.

Purpose of Functions

1) Uppercase Recognition (Complex)

We developed this code for recognize uppercase letters in text and split them into lines to use them in text file like list. Some data may contains sentences or texts back to back without any space between words. This program recognizes uppercase letter and holds it as a word until next uppercase letter. We stored the uppercase indices from given file and toke the letters between those indices. After holding words in memory, it prints the data to a text file line by line like a list.

2) Prime Number Factorization (Mid Complex)

We developed this code for doing prime number factorization calculation. In some mathematical problems, we need to factorize a number into its prime number factors. This program recognizes if the given number is a prime number or not and if it is not it divides number by smallest prime number to biggest one. After division, program prints the output like a table that most of students and instructors uses to a text file. This program can help many students who are dealing with bigger prime numbers and who can not solve prime number factorization correctly.

3) Body Mass Index Calculator (Mid Complex)

Body Mass Index (BMI) is very important value for people's lives. Within the obesity rate and unhealthy nourishment rise, people should be careful about their healths. This program gets your height and weight values and put it into a formula that calculates your BMI. After calculation, it gives you a results depends on your BMI value to show you are healthy or not. Also the program gives errors if user inputs invalid arguments like weight greater than height etc. .

4) Number Guessing Game (Complex)

We developed this code for users who wants to have fun with guessing a number that computer holds with computer's redirections. The program gets a range for guessing random number from user as an input and then it holds a random number between chosen range. After user tries to guess the number that computer holds program guides user to choose higher or lower number. When user guesses right number, program shows how many tries that user used to guess number.

Preparation Process of Project

While coding, we shared responsibilities of project equally. We worked together in all parts of project and helped each other respectively. The most difficult part of this project was finding the idea for coding program. We tried to use our creativity and adapt this with what we learned during this course.

We met four times to do project. After we found ideas and started to develop algorithms it becomes easier because getting along with coding helped us so much. We used various sources to develop algorithms and code. Facing with some errors, trying to understand what caused errors and debugging them taught us so many things, helped us to understand objectives and applications of this course better.

How Our Codes Works?

1) Uppercase Recognition

Our code firstly gets an input from user like “AhmetMehmetHakanSercan”. All words are back to back and first letters of them are uppercase letters. It recognizes uppercase letters by their ASCII codes. After recognizing first letter, it puts the other letters until the other uppercase letter that inside of ASCII code range. It stores words on an array and works inside a loop that doesn’t stop until input ends.

After loop faces with null space, it stops working and writes the words that computer holds in memory into a text file line by line. We used pointer and file operations commands for storing words, writing them into text file and etc. We used array types to store words, letters. We used loops and if-else statements for recognizing uppercase letters and defining conditions.

2) Prime Number Factorization

Firstly we declared an array and defined prime numbers in it. Our code simply gets an number input from user and calculates prime factors of this number. We defined numbers and dividors arrays to hold them into that. Then we are checking if we divide number to prime number from smallest, it give a remainder 0. If remainder equals 0, then it tries to divide it to this number again. If it is not zero, it passes to another prime number. After all that calculations, it gives a table that instructors and students use while solving problems and prints it into text file.

We used pointer and file operation commands for printing result table into text file. We used arrays to store prime numbers. We used loops and if-else statements for checking calculations and continuing to division operation.

3) Body Mass Index Calculator

Simply, we get two inputs from user that represents user's weight and height. After getting it from user we put values into BMI calculation formula and classify it depending on well accepted BMI table. It shows user if he/she's healthy or not with BMI value. For calculating BMI value we used Binary Calculator (bc) command because we need fractional numbers while classifying healthiness.

4) Number Guessing Game

Our number guessing game has short but tricky algorithm behind it. Firstly we get an input from user that determines range of number that computer holds. We used jot command to get random value. Then we converted random generator number to integer that we can use while comparing numbers. After comparing numbers, algorithm decides guessed number is equal to holded number or not. If they are not equal, it shows Higher or Lower message to guide user for guessing number. After several tries and user guessed number right, user can see how many tries that he/she take for guessing right number.

Test Runs of Codes

1) Uppercase Recognition

```
ENTER A COMMAND :
=================>> upper AhmetMehmetSercanHakanSeniz

Ahmet
Mehmet
Sercan
Hakan
Seniz
```

upperCase.txt

```
Ahmet
Mehmet
Sercan
Hakan
Seniz
```

2) Prime Number Factorization

```
ENTER A COMMAND :
================>> primef 21772800

Prime factorization of 21772800

 21772800|2
 10886400|2
  5443200|2
  2721600|2
  1360800|2
   680400|2
   340200|2
   170100|2
    85050|2
    42525|3
    14175|3
     4725|3
     1575|3
      525|3
      175|5
       35|5
        7|7
        1|
```

primef.txt

```
Prime factorization of 21772800

 21772800|2
 10886400|2
  5443200|2
  2721600|2
  1360800|2
   680400|2
   340200|2
   170100|2
    85050|2
    42525|3
    14175|3
     4725|3
     1575|3
      525|3
      175|5
       35|5
        7|7
        1|
```

3) Body Mass Index Calculator

```
ENTER A COMMAND :
================>> bmi 90 180
BMI: 27
OVERWEIGHT
```

bmi.txt

```
Weight: 90kg Height: 180cm
BMI: 27
RESULT: Overweight
```

```
ENTER A COMMAND :
================>> bmi 85 162
BMI: 32
OBESE
```

bmi.txt

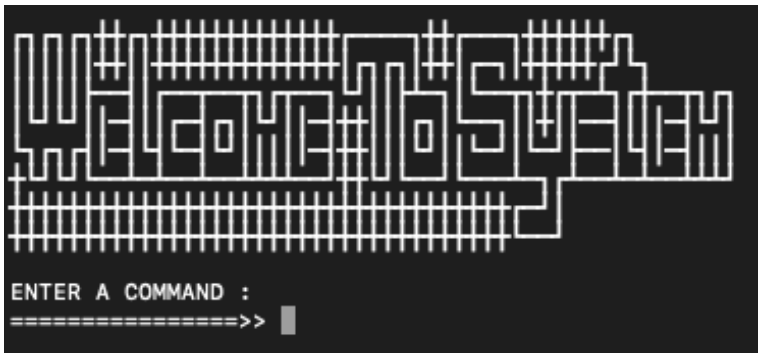
```
Weight: 85kg Height: 162cm
BMI: 32
RESULT: Obese
```

4) Number Guessing Game

```
ENTER A COMMAND :
================>> guess 50
Make a Guess
25
HIGHER!
Your guess: 25
Number of try: 1
40
LOWER!
Your guess: 40
Number of try: 2
30
HIGHER!
Your guess: 30
Number of try: 3
35
LOWER!
Your guess: 35
Number of try: 4
32
LOWER!
Your guess: 32
Number of try: 5
31

CORRECT!!!
Your guess: 31
Number of try: 6
You guessed my number 6 tries!
```

5) Terminal




```
ENTER A COMMAND :
==========>> 90 180
Given command is not valid => 90
COMMAND: guide ===> For more information.
```

```
ENTER A COMMAND :
==========>> bmi 100 50

 ERROR: Enter a valid argument!
 bmi *weight(kg) *height(cm)
 COMMAND: guide ===> For more information.
```

```
ENTER A COMMAND :
==========>> guide

bmi *weight(kg) *height(cm) => Calculates the body mass index of given input.

guess *positive_range =======> System picking a number between given range.

primef *positive_number =====> Prime factorization of given nnumber.

upper *str ==================> Seperating words by looking uppercase letters.

exit ========================> Quit from the system.
```

```
ENTER A COMMAND :
==========>> exit
logout
Saving session...
...copying shared history...
...saving history...truncating history files...
...completed.
```

Sources

http://www.asciitable.com/
https://www.tutorialsandyou.com/bash-shell-scripting/bash-bc-18.html
https://www.cyberciti.biz/faq/bash-for-loop/
https://www.computerhope.com/unix/bash/exec.htm
https://www.tutorialspoint.com/c_standard_library/c_function_strcmp.htm
https://stackoverflow.com/questions/3736210/how-to-execute-a-shell-script-from-c-in-linux
https://stackoverflow.com/questions/26508138/split-a-sentence-using-space-in-bash-script
https://stackoverflow.com/questions/4137262/is-there-an-easy-way-to-determine-if-user-input-is-an-integer-in-bash
http://tldp.org/LDP/Bash-Beginners-Guide/html/sect_09_02.html
http://tldp.org/LDP/abs/html/comparison-ops.html