Project Name:

SCRIPT VELOCITY

COURSE CODE : CT-175

Group Name:

KREKHEDS

Group Members:	Roll Numbers
Muhammad Subhan Khan	CT 24075
Danyal Abbas	CT 24084
Moazzam Farooqui	CT 24068

Project Documentation:

Typing Speed Calculator in C Language

1. Project Description:

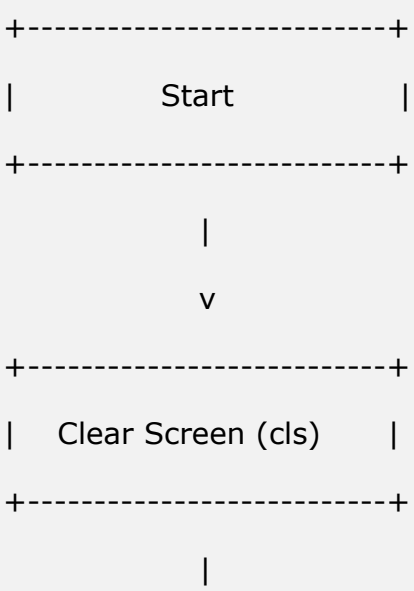
This project (ScriptVelocity) is a **Typing Speed Calculator** implemented in C. The program measures the user's typing speed in words per minute (WPM), accuracy, and the total number of errors while typing a randomly selected sentence. The program also provides real-time feedback on typing accuracy and speed, enhancing the user's typing skills.

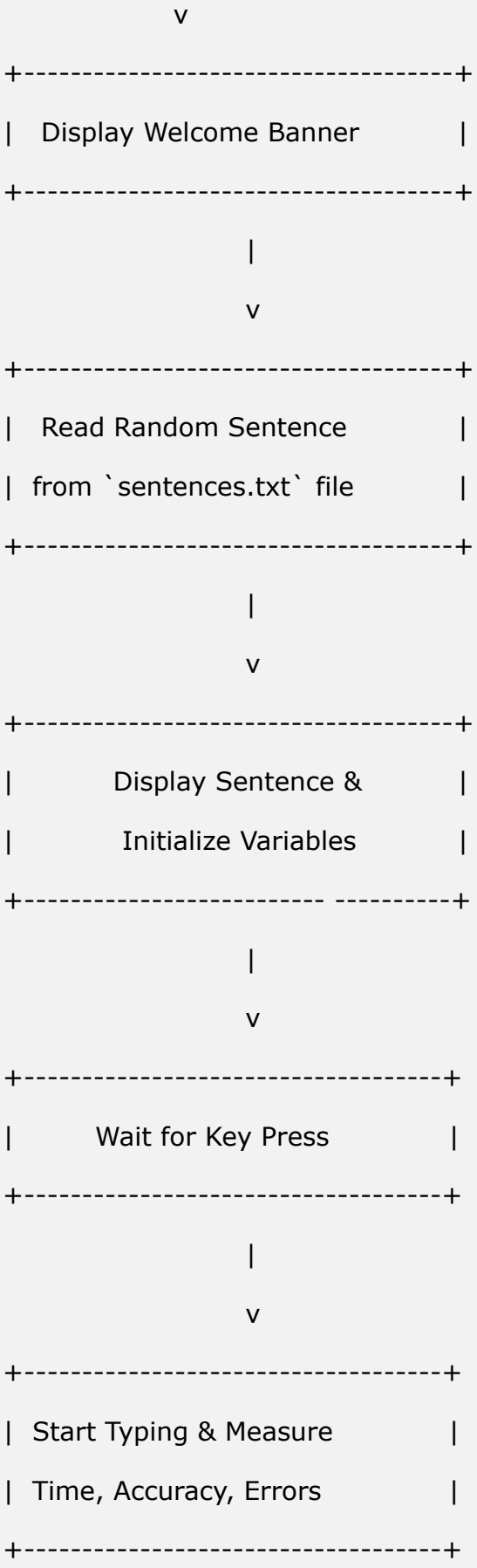
2. Project Functionalities/Features:

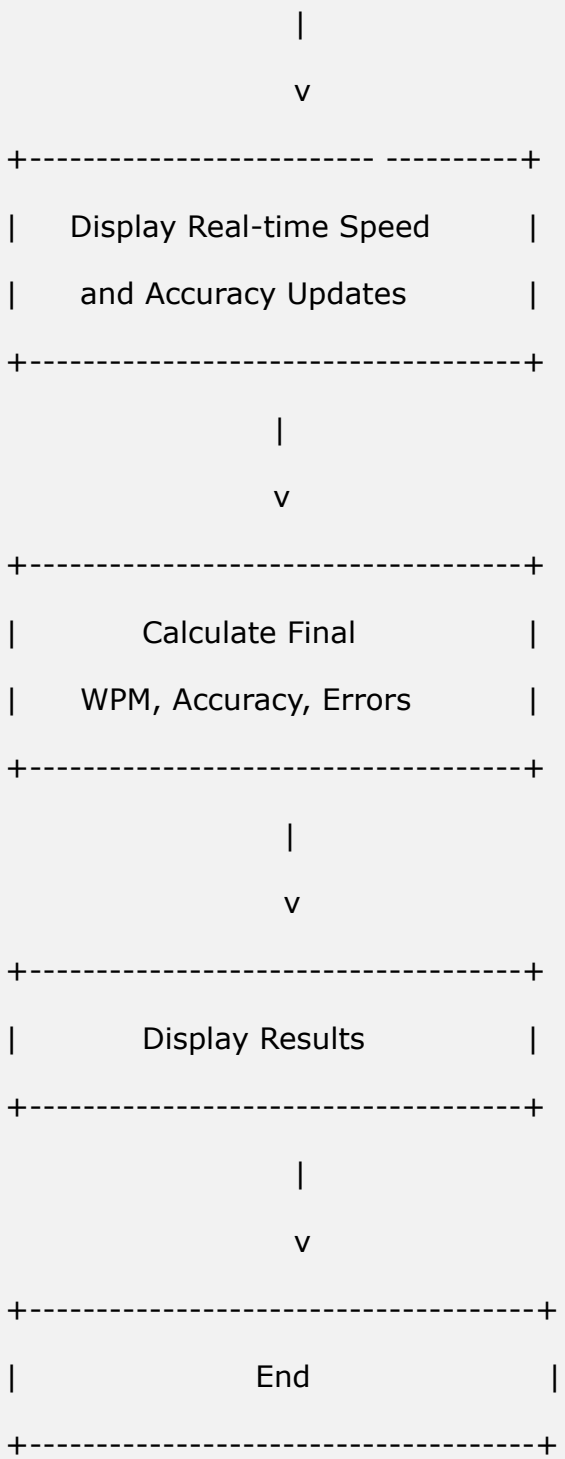
The main features of the project include:

1. **Random Sentence Selection:**
   The program reads a random sentence from a text file (`sentences.txt`).
2. **Typing Speed Calculation:**
   Calculates Words Per Minute (WPM) based on the total number of characters typed and the elapsed time.
3. **Real-Time Accuracy Monitoring:**
   Displays real-time accuracy and adjusts dynamically as the user types.
4. **Error Tracking:**
   Keeps track of the number of errors and updates the accuracy percentage accordingly.
5. **Console-Based Interface:**
   Utilizes a console-based interface with cursor control (`gotoxy()` function) for real-time updates.

3. Project Flowchart:




v
Display Welcome Banner
v
Read Random Sentence
from `sentences.txt` file
v
Display Sentence &
Initialize Variables
v
Wait for Key Press
v
Start Typing & Measure
Time, Accuracy, Errors


Display Real-time Speed
and Accuracy Updates
Calculate Final
WPM, Accuracy, Errors
Display Results
End

4. Datatype Description:

Datatype	Variable	Description
FILE *	File	Pointer to access the file containing sentences.
Char	c	Used to store user input during typing.
char[]	Sentence	Stores the selected random sentence.
long	LettersLength	Stores the total number of characters in the sentence.
long	LetterCount	Counts the correctly typed letters.
float	Accuracy	Stores the accuracy percentage.
float	Speed	Stores the calculated typing speed in WPM.
double	StartTime,StopTime	Track time taken by the user to type.
int	Errors	Tracks the total number of typing errors.
double	LettertoWord	Average letters per word, calculated from spaces.

5. Functions Description

5.1 int main()

Description:
The main function that drives the program, coordinating between reading the file, displaying the sentence, and calculating speed and accuracy.

- **Inputs:**
  None (User input is taken during runtime).
- **Outputs:**
  Displays typing speed, accuracy, and total errors on the console.

5.2 gotoxy(int x, int y)

Description:
Positions the cursor at a specified location `(x, y)` on the console.

- **Inputs:**
  `int x` - Horizontal position.
  `int y` - Vertical position.
- **Outputs:**
  Moves the cursor to the specified position using `SetConsoleCursorPosition()`.

- **Code:**

```
void gotoxy(int x,int y)
{

    COORD coord; // COORD STRUCT

    coord.X = x ;
    coord.Y = y ;

    SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE),coord);
}
```

5.3 File Handling: sentences.txt

Description:

The program reads a random line from a file `sentences.txt` containing multiple sentences.

- **Code:**

```
file = fopen("sentences.txt", "r"); // reading the file sentences.txt

    char c, Sentence[10000]; // Store a sentence from a file using file
management in C language

    for (int i = 0; i <= random_line; i++)
        fgets(Sentence, sizeof(Sentence), file); // adding the random sentence
in the variable
    fclose(file);
```

5.4 Accuracy Calculation

The program calculates accuracy by reducing the percentage based on errors:

- **Code:**

```
Accuracy = ((float)100/LettersLength)*(LettersLength-Errors);
```

6. Source Code:

```
7. // SCRIPT VELOCITY - A Typing Speed Checker CLI Program
```

```
8. // By Danyal Abbas (CT-24084), Moazzam Farooqui (CT-24068), and Subhan
   Khan (CT-24075)
9.
10.#include <windows.h> // For gotoxy function
11.#include <string.h> // For String inputs and functions
12.#include <time.h> // For Time calculations
13.#include <stdio.h> // For printing statements
14.#include <stdlib.h> // For File Management
15.
16.void gotoxy(int x,int y); // Function for managing cursor on CLI
17.
18.int main()
19.{
20.    system("cls"); // To first clear out the screen
21.
22.    // ###SUBHAN'S AREA TO CODE###
23.
24.    printf("\n\n");
25.    printf("  _________             .__        __ ____   ____     .__
               .__  __          \n");
26.    printf(" /   _____/ ____________|__|______/  |\\   \\ /   /____ |  |
   ____   ____ |__|/  |_ ___.__.\n");
27.    printf(" \\_____  \\_/ ___\\_  __ \\  \\____ \\   __\\   Y   // __
   \\|  |  /  _ \\_/ ___\\|  \\   __<   |  |\n");
28.    printf(" /        \\  \\___|  | \\/  |  |_> >  |  \\     /\\  ___/|
    |_(  <_> )  \\___|  ||  |  \\___  |\n");
29.    printf("/_______  /\\___  >__|  |__|   __/|__|   \\___/  \\___
    >____/\\____/ \\___  >__||__|  / ____|\n");
30.    printf("        \\/     \\/         |__|                    \\/
              \\/          \\/     \n");
31.    printf("\n\n");
32.
33.    // ############################
34.
35.    // ###MOAZZAM'S AREA TO CODE###
36.    FILE *file;
37.    int total_sentences = 10;
38.    int random_line;
39.    srand(time(NULL));
40.
41.    random_line = rand() % total_sentences; // random function to pick a
   random sentence from the file
42.
43.    file = fopen("sentences.txt", "r"); // reading the file sentences.txt
44.
45.    char c, Sentence[10000]; // Store a sentence from a file using file
   management in C language
46.
```

```
47.     for (int i = 0; i <= random_line; i++)
48.         fgets(Sentence, sizeof(Sentence), file); // adding the random
    sentence in the variable
49.     fclose(file);
50.
51.     size_t len = strlen(Sentence);
52.     // Removing the "\n" from the sentence and replacing it with NULL
    character
53.     if (Sentence[len - 1] == '\n')
54.         Sentence[len - 1] = '\0';
55.
56.     // ############################
57.
58.     // ###DANYAL'S AREA TO CODE###
59.
60.     // main variable declaration
61.     long LettersLength = 0, LetterCount = 0;
62.     float Accuracy = 0 , Speed = 0;
63.     double StartTime , StopTime = 0, LetterToWord = 1;
64.
65.     // Basic print statements that will be manipulated further in the
    code
66.     printf("\n\nYOUR TEXT:    \n");
67.     puts(Sentence);
68.     printf( "\n                    SPEED: 0          ACCURACY: 0
     TOTAL ERROR: 0" );
69.
70.     // Length of the string variable "Sentence"
71.     LettersLength = strlen(Sentence);
72.
73.     // for-loop to calculate the number of spaces in the string
    "Sentence"
74.     for (int i = 0; i < LettersLength; i++)
75.         if (Sentence[i] == ' ')
76.             LetterToWord++;
77.
78.     // calculate average space per character rate
79.     LetterToWord /= LettersLength;
80.
81.     // printf("\n%ld", LetterToWord);
82.
83.     printf("\n-----------------------------------------------");
84.     printf("\n\nEnter Any Key To Start Typing.........");
85.
86.     // waits for user to press any key
87.     c = getch();
88.
```

```
    // change the cursor point to make "Accuracy : 100%" and going back to normal cursor position
    gotoxy(48, 15);
    printf("100%c", '%');
    gotoxy(0, 18);
    printf("ENTER THE ABOVE TEXT:                    \n\n");

    // Time variables using the time.h headerfile
    time_t t,t1;

    t = clock();
    gotoxy(0,13);

       // while-loop to traverse through the string variable "Sentence"
       int k = 0, Errors = 0;
       while(Sentence[k] != '\0')
       {
           // clock ticks
           t1 = clock();

           c = getch(); // takes an input character

           if(c != Sentence[k])
           {
               printf("\a");
               Errors++;

               Accuracy = ((float)100/LettersLength)*(LettersLength-Errors);

                   if(Accuracy <= 0)
                   {
                       gotoxy(48, 15);
                       printf("0.00%c    ",'%');
                   }
                   if(Accuracy > 0)
                   {
                       gotoxy(48, 15);
                       printf("%0.2f%c ", Accuracy, '%');
                   }

               gotoxy(70, 15);
               printf("%d", Errors);
               gotoxy(0, 13);

               for(int j = 0; j < k; j++)
                   printf("%c", Sentence[j]);
           }
```

```
135.          else
136.          {
137.              LetterCount++;
138.              gotoxy(0,20);
139.
140.              for(int j = 0; j < k; j++)
141.                  printf("%c", Sentence[j]);
142.              printf("%c", c);
143.              k++;
144.          }
145.
146.              t1 = clock() - t1; // total clock ticks through the loop
147.              StartTime=((double)t1/CLOCKS_PER_SEC); // converting the
    ticks to seconds
148.              StopTime = StopTime + StartTime;
149.
150.              gotoxy(27,15);
151.              printf("%0.2f WPM ",
    (float)(LetterToWord*LetterCount)/(StopTime/60)); // calculating WPM
    (Words Per Minute)
152.
153.              gotoxy(0,13);
154.              for(int j=0; j<k; j++)
155.                  printf("%c", Sentence[j]);
156.      }
157.
158.      t=clock()-t; // clock ticks outside the loop
159.
160.      double YourTime =((double)t/CLOCKS_PER_SEC); // converting to
    seconds
161.      YourTime -= (YourTime-StopTime);
162.
163.      gotoxy(0,20);
164.
165.      for(int j = 0; j < k; j++)
166.          printf("%c",Sentence[j]);
167.
168.      // printing out the results
169.      printf("\n\n-----------------------------------------------\n\n");
170.      printf("YOUR RESULT");
171.      printf("\n\nYOUR SPEED: %0.2f WPM  \n\n** ( Length Of One Word Is
    Taken As %0.2f Letters )\n",
    (LetterToWord*LettersLength)/(YourTime/60),(float)LettersLength/(LetterTo
    Word*LettersLength));
172.      printf("\nEnter Any Key to Exit......");
173.      getch();
174.      return 0;
175.
```

```
}

// A C-Language implementation of the C++ function gotoxy()
void gotoxy(int x,int y)
{

    COORD coord; // COORD STRUCT

    coord.X = x ;
    coord.Y = y ;

    SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE),coord);
}

```

CONTRIBUTION OF EACH GROUP MEMBER:

- ***Subhan Khan*** *has done the ASCII art work to make the presentation of the program appealing and user interface easier and used a time library to use functions like: clock() and CLOCK_PER_SEC for time tracking.*
- ***Moazzam Farooqi*** *has performed the implementation of file handling in the source code and has introduced the idea of LettersCount and LettertoWord.*
- ***Danyal Abbas*** *has written the logic of the typing speed checker which includes: accuracy checking,WPM ,error checking and also did the implementation of gotoxy function for cursor management.*