Yakup Ozturk, Bozok Uygar Uluc

Prof. Shakil Ahmed

EE 2850

Date 5/3/2025

Project Report: MorseBridge

1. Overview of the Project

Project Title: MorseBridge

Team Members: Bozok Uygar Uluc, Yakup Ozturk

Objectives:

Morse Bridge is a C language-based desktop application that converts mouse clicks into Morse code. The purpose of this application is to enable disabled individuals to communicate with the outside world more easily. In order to do this, there is a simple interface that disabled individuals can easily use. Users' short mouse clicks are converted to dots, and long mouse clicks are converted to dashes. Then the program converts these clicks to letters by “translateMorecode()” function. In this way, disabled individuals can write with only 1 movement function. With the appropriate hardware, this code can use other kinds(eyelid closure, neuron signals or breathing movements).

Scope:

The project can help many disabled individuals who have difficulty speaking and writing to communicate. Below is a list and description of some of the diseases we can help:

1. **ALS (Amyotrophic Lateral Sclerosis)**
    a. ALS, or amyotrophic lateral sclerosis, is a progressive neurodegenerative disease that affects nerve cells in the brain and spinal cord.[1] As a result, motor abilities gradually decrease (inability to speak and move). MorseBridge can be adapted to the most easily used movement by the patient.
2. **Spinal Cord Injury (SCI)**
    a. It is usually caused by trauma that damages the spinal cord (car accident, fall, violence). As a result of this disease, temporary or permanent paralysis can occur. This paralysis can cover the whole body or it can be regional.
3. **Cerebral Palsy (CP)**
    a. CP can be due to abnormal brain development, brain damage, infection, or immature birth. This disease can affect motor movements, muscle control, speech, cognitive function, and reduce their productivity.

Our application can be a solution to many other similar diseases that make speaking difficult.

2. Related Work

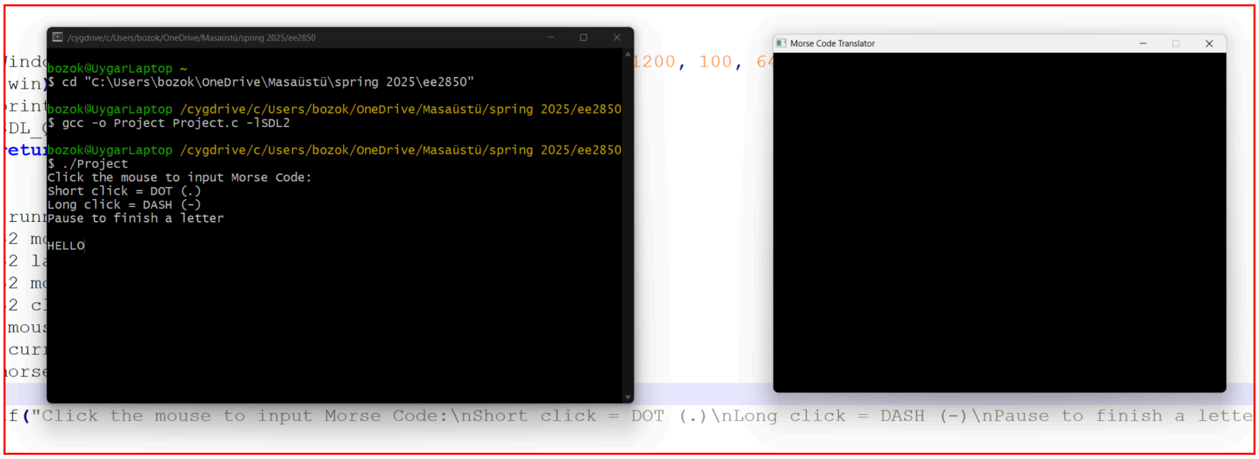
Morse code is a telecommunications method that encodes text characters as standardized sequences of two different signal durations, called dots and dashes. [2] This actually makes Morse code a language with two characters. In this way, Morse code can be used as a tool for communication when complex data transmission is not possible.

Modern Morse code in rehabilitation and education: new applications in assistive technology[3] For example, the Android operating system versions 5.0 and higher allow users to input text using Morse Code as an alternative to a keypad[4] Morse code has been shown to be a valuable tool in Assistive technology, augmentative and alternative communication, rehabilitation, and education, as well as adapted computer access methods via special software programs, hardware devices, and switches. After practicing on the proposed system repeatedly, three people with disabilities were able to familiarize themselves with Morse code operation. [5] There are also methods that enable disabled individuals to manage computers more easily, thanks to Morse code. [6] In addition, cheap and effective glasses that convert eye blinks into Morse code have also been developed. [7] As can be seen, Morse code was previously used to facilitate the communication of disabled individuals who had difficulty speaking.

In short, Morse code is a known system used to ease communication between disabled individuals. Thanks to this basic application we have created, we will be able to change it for various diseases and conditions and make a product specific to patients.

3. Project Technical Description

- **Technologies Used**
    a. Programming Language: C
    b. Libraries: SDL2/SDL.h[8], stdio.h, string.h, stdbool.h, time.h
- **Data processing**
    a. The application listens for mouse clicks through a window created with SDL. The created window can be seen in Figure 1 below.



```
bozok@UygarLaptop ~
$ cd "C:\Users\bozok\OneDrive\Masaüstü\spring 2025\ee2850"

bozok@UygarLaptop /cygdrive/c/Users/bozok/OneDrive/Masaüstü/spring 2025/ee2850
$ gcc -o Project Project.c -lSDL2

bozok@UygarLaptop /cygdrive/c/Users/bozok/OneDrive/Masaüstü/spring 2025/ee2850
$ ./Project
Click the mouse to input Morse Code:
Short click = DOT (.)
Long click = DASH (-)
Pause to finish a letter

HELLO
```

```
f("Click the mouse to input Morse Code:\nShort click = DOT (.)\nLong click = DASH (-)\nPause to finish a lette
```

/cygdrive/c/Users/bozok/OneDrive/Masaüstü/spring 2025/ee2850
Morse Code Translator

Figure-1

b. Clicks under 0.2 seconds are interpreted as a dot(.).

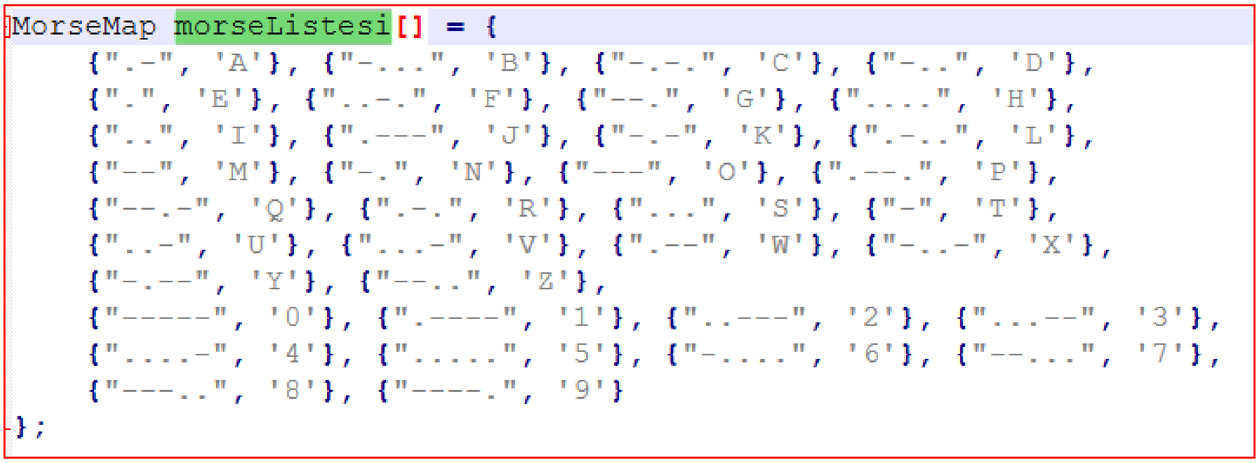
c. Clicks above 0.2 seconds are interpreted as dash (-).

d. Waiting longer than 0.6 seconds means that the letter has been written and you can move on to the next letter.

e. Waiting longer than 3 seconds creates a gap (“ “).

- **Converting Morse code to letters**

a. The array named morseListesi contains the morse code and its corresponding character, respectively. This situation can be seen from figure 2.



```
MorseMap morseListesi[] = {
    {".-", 'A'}, {"-...", 'B'}, {"-.-.", 'C'}, {"-..", 'D'},
    {".", 'E'}, {"..-.", 'F'}, {"--.", 'G'}, {"....", 'H'},
    {"..", 'I'}, {".---", 'J'}, {"-.-", 'K'}, {".-..", 'L'},
    {"--", 'M'}, {"-.", 'N'}, {"---", 'O'}, {".--.", 'P'},
    {"--.-", 'Q'}, {".-.", 'R'}, {"...", 'S'}, {"-", 'T'},
    {"..-", 'U'}, {"...-", 'V'}, {".--", 'W'}, {"-..-", 'X'},
    {"-.--", 'Y'}, {"--..", 'Z'},
    {"-----", '0'}, {".----", '1'}, {"..---", '2'}, {"...--", '3'},
    {"....-", '4'}, {".....", '5'}, {"-....", '6'}, {"--...", '7'},
    {"---..", '8'}, {"----.", '9'}
};
```

Figure-2

b. Thanks to the for loop we created, all characters are checked in input order.

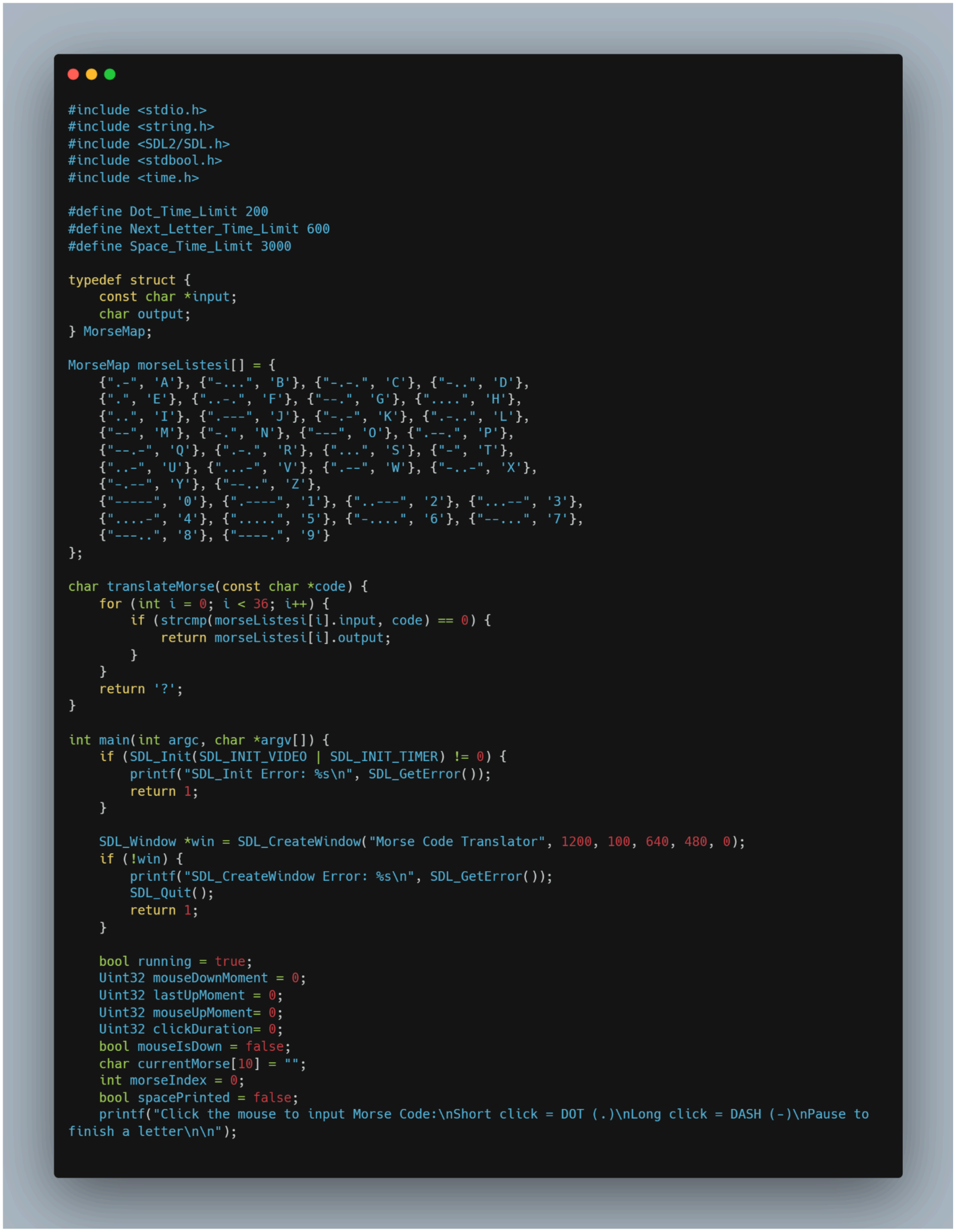
c. If the Morse code matches a character, that character is written to the screen. If the morse code is not processed with a character, “?” is displayed on the screen. is written.

- **Planned Future Features:**
  a. Audio output using a text-to-speech engine.
  b. File storage for translated messages.
  c. Interactive hints.
  d. AI-powered text completion

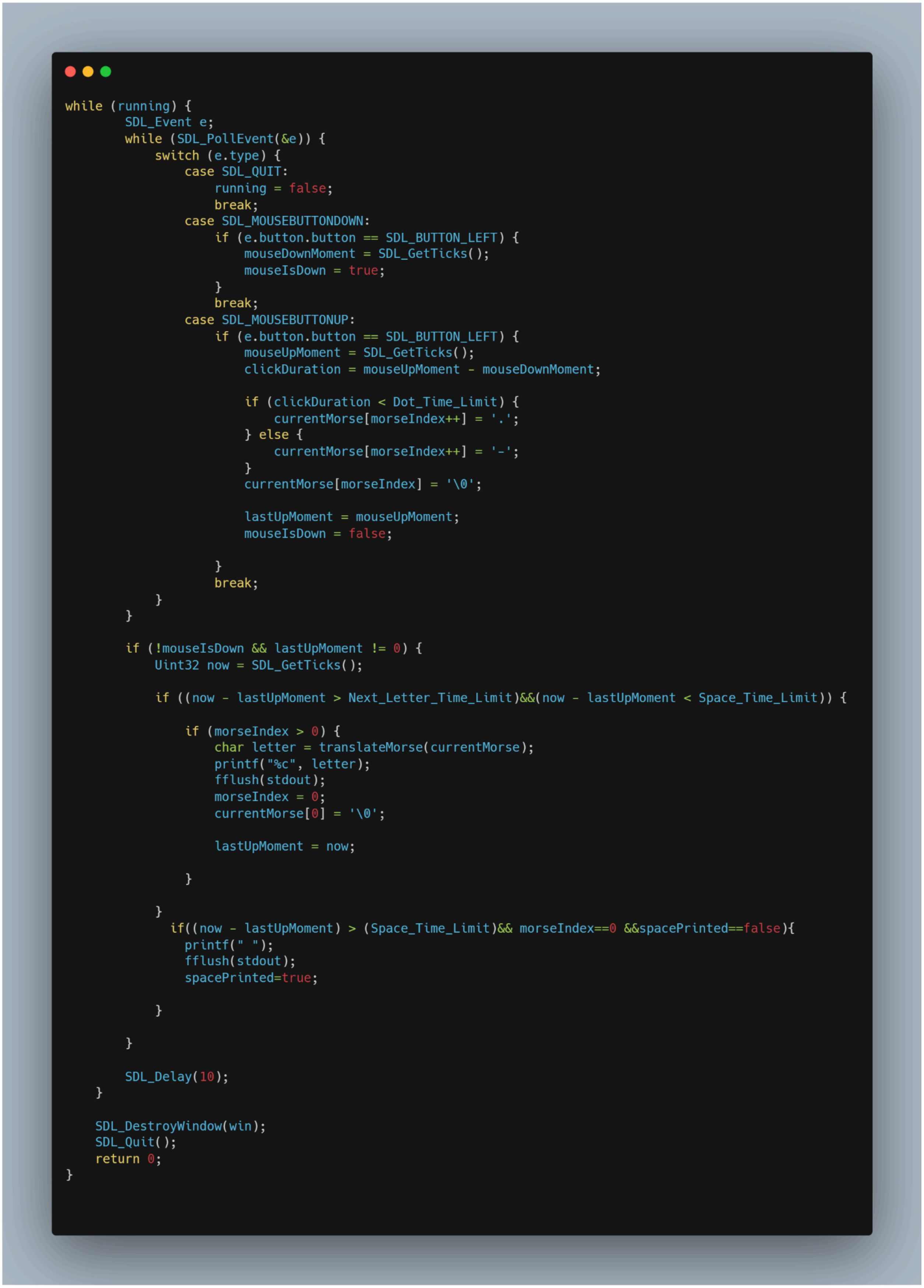
4. Results / Output

- **Functional Features**
  a. Our program can convert dot and dash data received from the user in real time into english characters.
  b. When it encounters a code that is not in the Morse code, it notifies the user that an error has been encountered.
    i. Example: "---...---.." ⇒ DNE ⇒ Does Not Exitst
- **Example Output**
  a. "....", ".", ".-..", "---" ⇒ HELLO
- **Testing Observations**
  a. The clicking mechanism works correctly.
  b. The window created by SDL works properly without any problems.
- **Limitations**
  a. There's no GUI feedback or audio output yet.
  b. There's no File storage.
  c. There is no complicated AI system that can correct the lines or express errors.

5. Code



```c
#include <stdio.h>
#include <string.h>
#include <SDL2/SDL.h>
#include <stdbool.h>
#include <time.h>

#define Dot_Time_Limit 200
#define Next_Letter_Time_Limit 600
#define Space_Time_Limit 3000

typedef struct {
    const char *input;
    char output;
} MorseMap;

MorseMap morseListesi[] = {
    {".-", 'A'}, {"-...", 'B'}, {"-.-.", 'C'}, {"-..", 'D'},
    {".", 'E'}, {"..-.", 'F'}, {"--.", 'G'}, {"....", 'H'},
    {"..", 'I'}, {".---", 'J'}, {"-.-", 'K'}, {".-..", 'L'},
    {"--", 'M'}, {"-.", 'N'}, {"---", 'O'}, {".--.", 'P'},
    {"--.-", 'Q'}, {".-.", 'R'}, {"...", 'S'}, {"-", 'T'},
    {"..-", 'U'}, {"...-", 'V'}, {".--", 'W'}, {"-..-", 'X'},
    {"-.--", 'Y'}, {"--..", 'Z'},
    {"-----", '0'}, {".----", '1'}, {"..---", '2'}, {"...--", '3'},
    {"....-", '4'}, {".....", '5'}, {"-....", '6'}, {"--...", '7'},
    {"---..", '8'}, {"----.", '9'}
};

char translateMorse(const char *code) {
    for (int i = 0; i < 36; i++) {
        if (strcmp(morseListesi[i].input, code) == 0) {
            return morseListesi[i].output;
        }
    }
    return '?';
}

int main(int argc, char *argv[]) {
    if (SDL_Init(SDL_INIT_VIDEO | SDL_INIT_TIMER) != 0) {
        printf("SDL_Init Error: %s\n", SDL_GetError());
        return 1;
    }

    SDL_Window *win = SDL_CreateWindow("Morse Code Translator", 1200, 100, 640, 480, 0);
    if (!win) {
        printf("SDL_CreateWindow Error: %s\n", SDL_GetError());
        SDL_Quit();
        return 1;
    }

    bool running = true;
    Uint32 mouseDownMoment = 0;
    Uint32 lastUpMoment = 0;
    Uint32 mouseUpMoment= 0;
    Uint32 clickDuration= 0;
    bool mouseIsDown = false;
    char currentMorse[10] = "";
    int morseIndex = 0;
    bool spacePrinted = false;
    printf("Click the mouse to input Morse Code:\nShort click = DOT (.)\nLong click = DASH (-)\nPause to
finish a letter\n\n");
```



```
while (running) {
        SDL_Event e;
        while (SDL_PollEvent(&e)) {
            switch (e.type) {
                case SDL_QUIT:
                    running = false;
                    break;
                case SDL_MOUSEBUTTONDOWN:
                    if (e.button.button == SDL_BUTTON_LEFT) {
                        mouseDownMoment = SDL_GetTicks();
                        mouseIsDown = true;
                    }
                    break;
                case SDL_MOUSEBUTTONUP:
                    if (e.button.button == SDL_BUTTON_LEFT) {
                        mouseUpMoment = SDL_GetTicks();
                        clickDuration = mouseUpMoment - mouseDownMoment;

                        if (clickDuration < Dot_Time_Limit) {
                            currentMorse[morseIndex++] = '.';
                        } else {
                            currentMorse[morseIndex++] = '-';
                        }
                        currentMorse[morseIndex] = '\0';

                        lastUpMoment = mouseUpMoment;
                        mouseIsDown = false;

                    }
                    break;
            }
        }

        if (!mouseIsDown && lastUpMoment != 0) {
            Uint32 now = SDL_GetTicks();

            if ((now - lastUpMoment > Next_Letter_Time_Limit)&&(now - lastUpMoment < Space_Time_Limit)) {

                if (morseIndex > 0) {
                    char letter = translateMorse(currentMorse);
                    printf("%c", letter);
                    fflush(stdout);
                    morseIndex = 0;
                    currentMorse[0] = '\0';

                    lastUpMoment = now;

                }

            }
              if((now - lastUpMoment) > (Space_Time_Limit)&& morseIndex==0 &&spacePrinted==false){
                printf(" ");
                fflush(stdout);
                spacePrinted=true;

            }

        }

        SDL_Delay(10);
    }

    SDL_DestroyWindow(win);
    SDL_Quit();
    return 0;
}
```

6. References

(1) ALS Association. “What Is ALS?” *ALS.org*, https://www.als.org/understanding-als/what-is-als. Accessed 5 May 2025.

(2) Wikipedia contributors. “Morse Code.” *Wikipedia: The Free Encyclopedia*, Wikimedia Foundation, https://en.wikipedia.org/wiki/Morse_code. Accessed 5 May 2025.

(3) Wikipedia contributors. “Morse Code.” *Wikipedia: The Free Encyclopedia*, Wikimedia Foundation, https://en.wikipedia.org/wiki/Morse_code#cite_note-51. Accessed 5 May 2025.

(4) Uyangoda, P. D., and N. Rajapakse. “An Interactive Morse Code Emulation Management System.” *ResearchGate*, https://www.researchgate.net/publication/248827374_An_Interactive_Morse_Code_Emulation_Management_System. Accessed 5 May 2025.

(5) Alagarsamy, V., et al. “Morse Code Input System for Mouse and Keyboard Emulation.” *International Journal of Advanced Engineering and Management*, https://ijaem.net/issue_dcp/Morse%20code%20Input%20System%20for%20Mouse%20and%20Keyboard%20Emulation.pdf?utm_source=chatgpt.com. Accessed 5 May 2025.

(6) De Marchis, C., et al. “The Role of Assistive Technologies in the Home Environment of Individuals with Neurological Diseases.” *Frontiers in Neurology*, vol. 12, 2021, Article 681689, https://pmc.ncbi.nlm.nih.gov/articles/PMC8202051/. Accessed 5 May 2025.

(7) “SDL Library in C/C++ with Examples.” *GeeksforGeeks*, https://www.geeksforgeeks.org/sdl-library-in-c-c-with-examples/. Accessed 5 May 2025.

(8) Wikipedia contributors. “Morse Code.” *Wikipedia: The Free Encyclopedia*, Wikimedia Foundation, https://en.wikipedia.org/wiki/Morse_code#cite_note-51. Accessed 5 May 2025. *(Duplicate citation if required by your formatting — otherwise, you can remove this.)*