



A PBL PROJECT

On

HANGMAN GAME USING C

Submitted By

T. MANI SHEKER

218R1A1259

B. MALATHI

218R1A1213

R. DIVYA

218R1A1252

Y. KARTHIK

218R1A1229

M. DHEERAJ

218R1A1241

Under the Guidance of

Mrs. T. SARIKA

Assistant Professor, Department of IT

DEPARTMENT OF INFORMATION TECHNOLOGY

CMR ENGINEERING COLLEGE

UGC AUTONOMOUS (Approved by AICTE-New Delhi & J.N.T.U, Hyderabad)
Kandlakoya (v), Medchal Road, Hyderabad-501 401, Telangana State, India.



CERTIFICATE

This is to certify that the project entitled " **HANGMAN GAME USING C** " is a work carried out by

T. MANI SHEKER	218R1A1259
B. MALATHI	218R1A1213
R. DIVYA	218R1A1252
Y. KARTHIK	218R1A1229
M. DHEERAJ	218R1A1241

In the part of innovative teaching methodology **PBL (Project Based Learning)** of Operating Systems Laboratory under our guidance and supervision.

Project Coordinator

Mrs. T. Sarika

Assistant Professor,
Department of IT,
CMR Engineering college,
Hyderabad

Head of the Department

Dr. Madhavi Pingili

Professor & HOD,
Department of IT,
CMR Engineering College,
Hyderabad

DECLARATION

This is to certify that the work reported in the present project entitled “**HANGMAN GAME USING C**” is a record of bonafide work done by us in the Department of Information Technology, CMR Engineering College, Hyderabad. The reports are based on the project work done entirely by us and not copied from any other source. We submit our project for further development by any interested students who share similar interests to improve the project in the future. The results embodied in this project report have not been submitted to any other University or Institute for the award of any degree or diploma to the best of our knowledge and belief.

T. MANI SHEKER	218R1A1259
B. MALATHI	218R1A1213
R. DIVYA	218R1A1252
Y. KARTHIK	218R1A1229
M. DHEERAJ	218R1A1241

ACKNOWLEDGEMENT

We are extremely grateful to our Principal, **Dr. A. Srinivasula Reddy**, and our HOD, **Dr. Madhavi Pingili**, Department of IT, CMR Engineering College for their constant support. We are immensely thankful to **Mrs. T. SARIKA**, Assistant Professor, Project Coordinator, Department of IT, for her constant guidance, encouragement and moral support throughout the project.

T. MANI SHEKER	218R1A1259
B. MALATHI	218R1A1213
R. DIVYA	218R1A1252
Y. KARTHIK	218R1A1229
M. DHEERAJ	218R1A1241

TABLE OF CONTENTS

CONTENT	PAGE NUMBER
DECLARATION	3
ACKNOWLEDGEMENT	4
ABSTRACT	6
INTRODUCTION	7
FUNCTIONAL REQUIREMENTS	8
ALGORITHM	9
SOURCE CODE	10-13
SCREEN SHOTS	14-16
OUTPUT SCREEN	17-19
CONCLUSION	20
REFERENCES	21

ABSTRACT

Hangman is a guessing game for two or more players. One player thinks of a word, phrase or sentence and the other(s) tries to guess it by suggesting letters within a certain number of guesses. Originally a Paper-and-pencil game, there are now electronic versions.

The word to guess is represented by a row of dashes representing each letter of the word. Rules may permit or forbid proper nouns, such as names, places, brands, or slang. If the guessing player suggests a letter which occurs in the word, the other player writes it in all its correct positions. If the suggested letter does not occur in the word, the other player removes (or alternatively, adds) one element of a hanged stick figure as a tally mark. Generally, the game ends once the word is guessed, or if the stick figure is complete — signifying that all guesses have been used.

The player guessing the word may, at any time, attempt to guess the whole word. If the word is correct, the game is over and the guesser wins. Otherwise, the other player may choose to penalize the guesser by adding an element to the diagram. On the other hand, if the guesser makes enough incorrect guesses to allow the other player to complete the diagram, the guesser loses. However, the guesser can also win by guessing all the letters that appear in the word, thereby completing the word, before the diagram is completed

INTRODUCTION

In the English language, the 12 most commonly occurring letters in descending order are: e-t-a-o-i-n-s-h-r-d-l-u. This and other letter-frequency lists are used by the guessing player to increase the odds when it is their turn to guess. On the other hand, the same lists can be used by the puzzle setter to stump their opponent by choosing a word which deliberately avoids common letters or one that contains rare letters.

Another common strategy is to guess vowels first, as English only has six vowels (a,e,i,o,u, and y), and almost every word has at least one.

Thus the user wins if he can guess the word or else he is a loser. In this programming assignment I intend to implement the user interface by which the code takes input as letters of the word and checks for its presence. Also another task is to reduce the no. of chances (lifelines) one by one as the user keeps on guessing incorrect letters.

My mini project will obviously illustrate the above mentioned task, and if time permits I even intend to enhance the GUI front end by adding some more buttons , message box, and labels.

FUNCTIONAL REQUIREMENTS

SOFTWARE USED:

Operating Systems	- Windows 7/10/11
Language	- C
Software	- TURBO C
Hardware	- Quad core 2GHZ or Higher

ALGORITHM

Step 1: Start

Step2: Display the rules of the game.

Step 3: Create an array of encrypted words, and select one randomly as the target word.

Step 4: Create an array of spaces to represent the target word, with each space representing letter.

Step 5: Create an array to keep track of incorrect guesses.

Step 6: Start a loop to take input of the player's guess and update the target word array and incorrect guess array.

Step 7: Display the number of mistakes and a hangman figure based on the number of incorrect guesses.

Step 8: End the loop when the target word is completely filled or the number of incorrect guesses reaches 6.

Step 9: Display the result (win or lose) and the target word.

Step 10: End

SOURCE CODE

/*Program for inter-process communication (IPC) between two processes in C using pipes: */

```
#include <stdio.h>
#include <stdlib.h>
#include <stdbool.h>
#include <time.h>
#include <string.h>

#define WORDS 10
#define WORDLEN 40
#define CHANCE 6

bool srand_called = false;

int i_rnd(int i) {
    if (!srand_called) {
        srand(time(NULL) << 10);
        srand_called = true;
    }
    return rand() % i;
}

char* decrypt(char* code) {
    int hash = ((strlen(code) - 3) / 3) + 2;
    char* decrypt = malloc(hash);
    char* toFree = decrypt;
    char* word = code;
    for (int ch = *code; ch != '\0'; ch = *(++code))
    {
        if((code - word + 2) % 3 == 1){
            *(decrypt++) = ch - (word - code + 1) - hash;
        }
    }
    *decrypt = '\0';
    return toFree;
}

void printBody(int mistakes, char* body) {
    printf("\tMistakes :%d\n", mistakes);
}
```

```
switch(mistakes) {

    case 6: body[6] = '\\'; break;
    case 5: body[5] = '/'; break;
    case 4: body[4] = '\\'; break;
    case 3: body[3] = '|'; break;


    case 2: body[2] = '/'; break;
    case 1: body[1] = ')', body[0] = '('; break;
    default: break;

}

printf("\t _____\n"
       "\t|      |\n"
       "\t|    %c %c\n"
       "\t|    %c %c %c\n"
       "\t|    %c %c\n"
       "\t|      \n"
       "\t|      ", body[0], body[1], body[2],
       body[3], body[4], body[5], body[6]);
}

void printWord(char* guess, int len) {
    printf("\t");
    for (int i = 0; i < len; ++i)
    {
        printf("%c ", guess[i]);
    }
    printf("\n\n");
}

int main() {
    printf("\n\t Be aware you can be hanged!!.");

    printf("\n\n\t Rules : ");
```

```

printf("\n\t - Maximum 6 mistakes are allowed.");
printf("\n\t - All alphabet are in lower case.");
printf("\n\t - All words are name of very popular Websites. eg. Google");
printf("\n\t - If you enjoy continue, otherwise close it.");
printf("\n\t Syntax : Alphabet");
printf("\n\t Example : a \n\n");
char values[WORDS][WORDLEN] =
{"N~mqOlJ^tZletXodeYgs","gCnDIffQe^CdP^^B{hZpeLA^hv","7urtrtwQv{dt`>}FaR]i]X
Uug^GI",

"aSwfXsxOsWAlXScVQmjAWJG","cruD=idduvUdr=gmcauCmg"],"BQt`zncypFVj
vIaTl]u=_?Aa}F",

"iLvKdT`yu~mWj[^gcO|","jSiLyzJ=vPmnv^`N]^>ViAC^z_","xo|RqqhO|nNstjmfzfi
uoiFfhwdh~","OHkttvxdp|[nnW]Drgaomdq"};
char *body = malloc(CHANCE+1);

int id = i_rnd(WORDS);
char *word = decrypt(values[id]);

int len = strlen(word);
char *guessed = malloc(len);
char falseWord[CHANCE];

memset(body, ' ', CHANCE+1);
memset(guessed, '_', len);
char guess;
bool found;
char* win;

int mistakes = 0;
setvbuf(stdin, NULL, _IONBF, 0);

```

```
do {

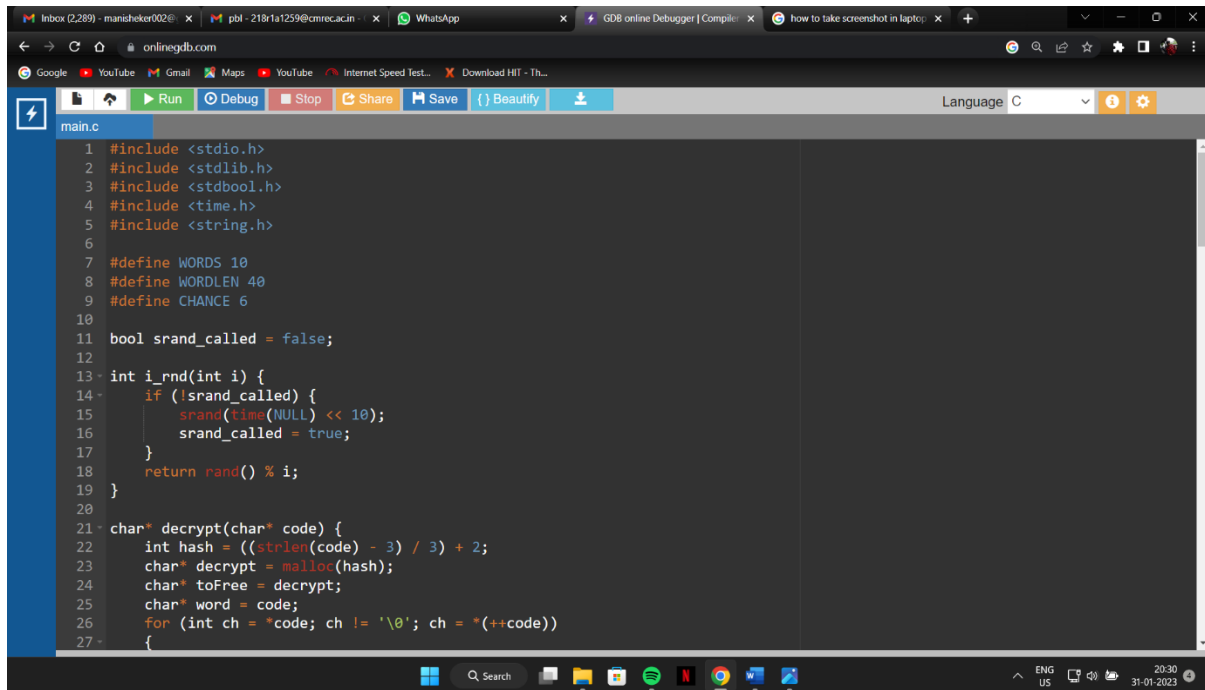
    found = false;
    printf("\n\n");
    printBody(mistakes, body);
    printf("\n\n");
    printf("\tFalse Letters : ");
    if(mistakes == 0) printf("None\n");
    for (int i = 0; i < mistakes; ++i)
    {
        printf("%c", falseWord[i]);
    }
    printf("\n\n");
    printWord(guessed, len);
    printf("\tGive me a alphabet in lower case : ");
    do {scanf("%c",&guess);} while ( getchar() != '\n' );
    for (int i = 0; i < len; ++i)
    {
        if(word[i] == guess) {
            found = true;
            guessed[i] = guess;
        }
    }
    if(!found) {
        falseWord[mistakes] = guess;
        mistakes += 1;
    }

    win = strchr(guessed, '_');
}while(mistakes < CHANCE && win != NULL);

if(win == NULL) {
    printf("\n");
    printWord(guessed, len);
    printf("\n\tCongrats! You have won : %s\n\n", word);
```

```
    } else {  
        printf("\n");  
        printBody(mistakes, body);  
        printf("\n\n\tBetter try next time. Word was %s\n\n", word);  
    }  
  
    free(body);  
    free(word);  
    free(guessed);  
    return EXIT_SUCCESS;  
}
```

SCREENSHOTS

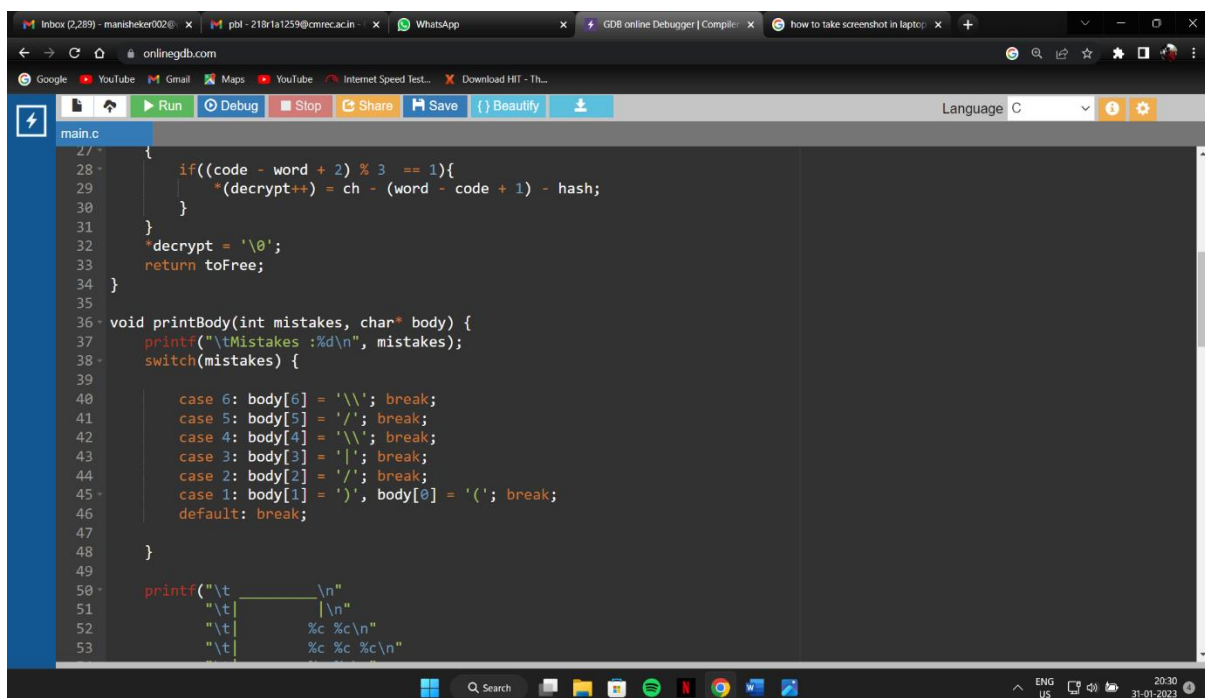


```

1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <stdbool.h>
4 #include <time.h>
5 #include <string.h>
6
7 #define WORDS 10
8 #define WORDLEN 40
9 #define CHANCE 6
10
11 bool srnd_called = false;
12
13 int i_rnd(int i) {
14     if (!srnd_called) {
15         srand(time(NULL) << 10);
16         srnd_called = true;
17     }
18     return rand() % i;
19 }
20
21 char* decrypt(char* code) {
22     int hash = ((strlen(code) - 3) / 3) + 2;
23     char* decrypt = malloc(hash);
24     char* toFree = decrypt;
25     char* word = code;
26     for (int ch = *code; ch != '\0'; ch = ++code)
27 {

```

Fig 1: Hangman Game code in C

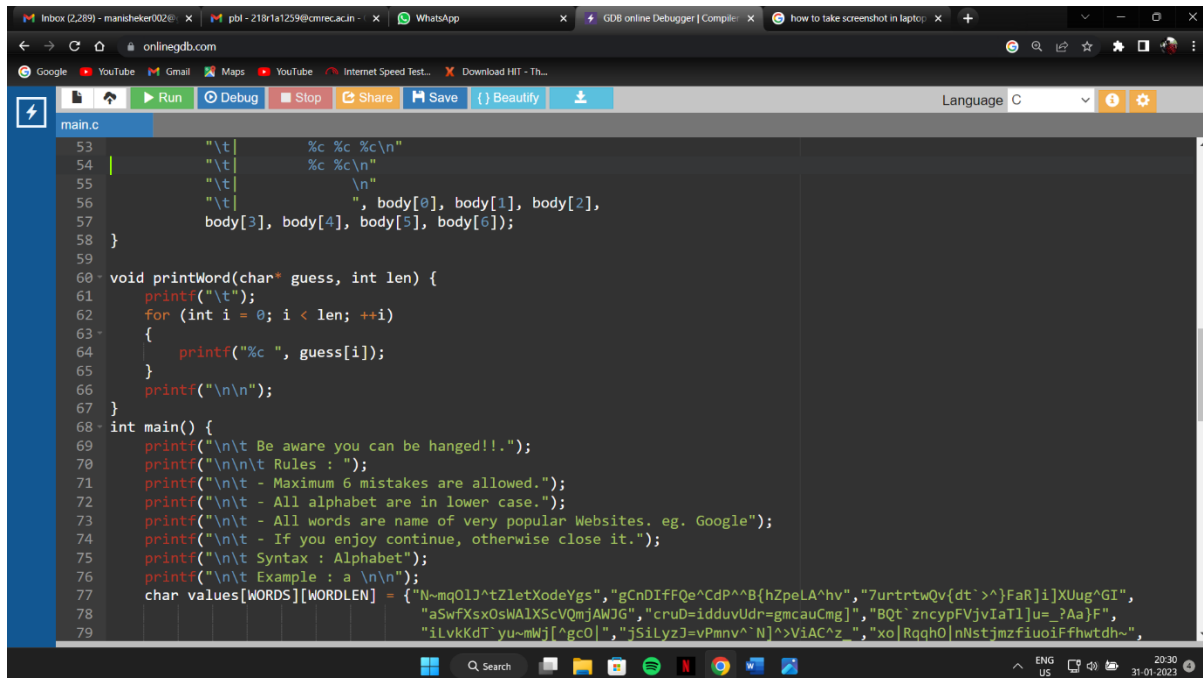


```

27 {
28     if ((code - word + 2) % 3 == 1) {
29         *(decrypt++) = ch - (word - code + 1) - hash;
30     }
31 }
32 *decrypt = '\0';
33 return toFree;
34 }
35
36 void printBody(int mistakes, char* body) {
37     printf("\tMistakes : %d\n", mistakes);
38     switch(mistakes) {
39
40         case 6: body[6] = '\\'; break;
41         case 5: body[5] = '/'; break;
42         case 4: body[4] = '\\'; break;
43         case 3: body[3] = '|'; break;
44         case 2: body[2] = '/'; break;
45         case 1: body[1] = ')', body[0] = '('; break;
46         default: break;
47     }
48 }
49
50 printf("\t\t\t\t\t\n"
51        "\t\t\t\t\t\n"
52        "\t\t\t\t\t\n"
53        "\t\t\t\t\t\n"

```

Fig 2: Hangman Game code in C

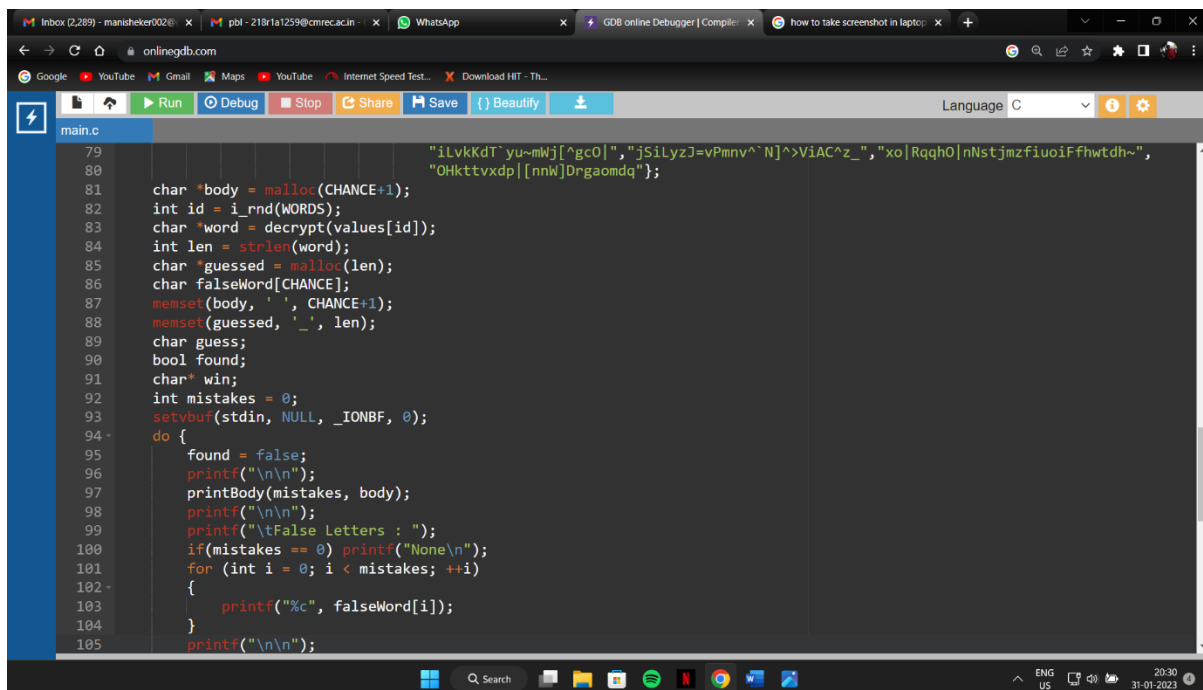


```

53     printf("\t\t\t\t\t%c %c %c\n",
54            "\t\t\t\t\t%c %c %c\n",
55            "\t\t\t\t\t", body[0], body[1], body[2],
56            body[3], body[4], body[5], body[6]);
57 }
58
59
60 void printWord(char* guess, int len) {
61     printf("\t\t\t\t\t");
62     for (int i = 0; i < len; ++i)
63     {
64         printf("%c ", guess[i]);
65     }
66     printf("\n\n");
67 }
68
69 int main() {
70     printf("\n\t\t\t\t\tBe aware you can be hanged!!.");
71     printf("\n\t\t\t\t\tRules : ");
72     printf("\n\t\t\t\t\t- Maximum 6 mistakes are allowed.");
73     printf("\n\t\t\t\t\t- All alphabet are in lower case.");
74     printf("\n\t\t\t\t\t- All words are name of very popular Websites. eg. Google");
75     printf("\n\t\t\t\t\t- If you enjoy continue, otherwise close it.");
76     printf("\n\t\t\t\t\tSyntax : Alphabet");
77     printf("\n\t\t\t\t\tExample : a \n\n");
78     char values[WORDS][WORDLEN] = {"N~mq0lJ^tZletXodeYgs", "gCnDlFQe^CdP^B{hZpeLA^hv", "7urtrtwQv{dt`>^}FaRji}XUug^GI",
79                                     "aSwfXsxOswAlXScVQmjAWJG", "cruD=idduvUdr=gmcuCMg]", "BQt`zncypFVjvIaTl]u= ?Aa}F",

```

Fig 3: Hangman Game code in C

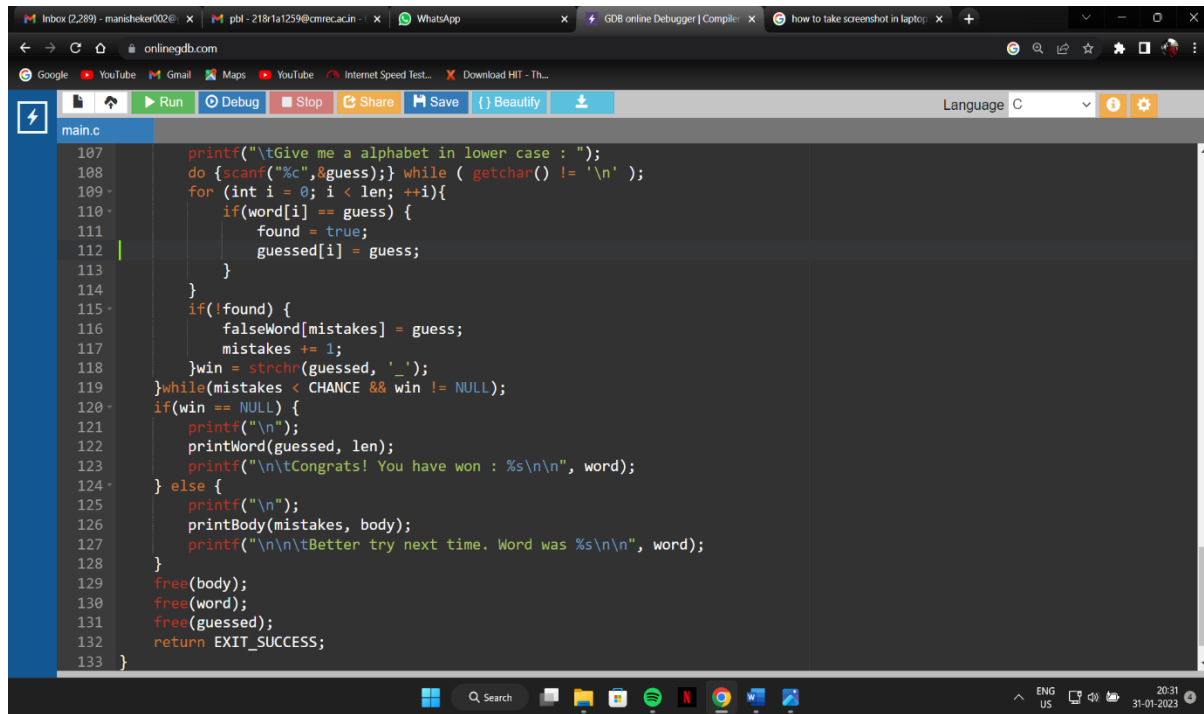


```

79     "iLvKdKdT`yu~mWj[^gcO|", "jSiLyZJ=vPmnv^N]^>ViAC^z_", "xo|RqqhO|nNstjmzfiuoiFfhwdh~",
80     "OHkttvxdp|[nnW]Drgaomdq");
81     char *body = malloc(CHANCE+1);
82     int id = i_rnd(WORDS);
83     char *word = decrypt(values[id]);
84     int len = strlen(word);
85     char *guessed = malloc(len);
86     char falseWord[CHANCE];
87     memset(body, ' ', CHANCE+1);
88     memset(guessed, ' ', len);
89     char guess;
90     bool found;
91     char* win;
92     int mistakes = 0;
93     setvbuf(stdin, NULL, _IONBF, 0);
94     do {
95         found = false;
96         printf("\n\n");
97         printBody(mistakes, body);
98         printf("\n\n");
99         printf("\t\t\t\t\tFalse Letters : ");
100         if(mistakes == 0) printf("None\n");
101         for (int i = 0; i < mistakes; ++i)
102         {
103             printf("%c", falseWord[i]);
104         }
105         printf("\n\n");

```

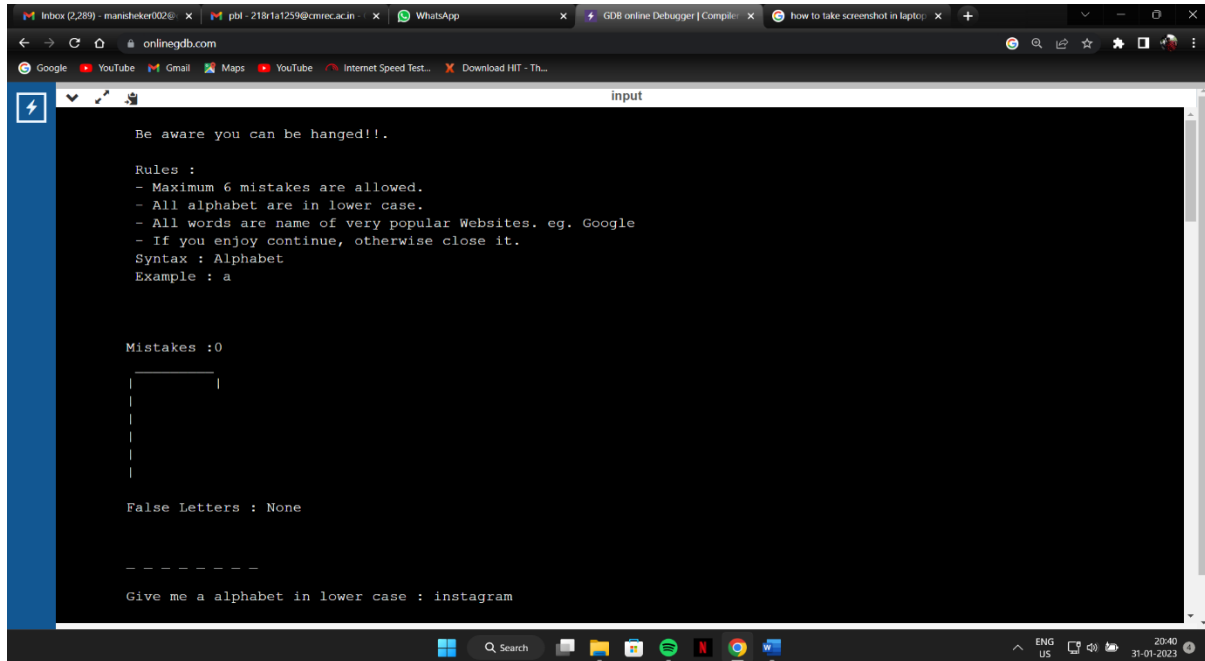
Fig 4: Hangman Game code in C



```
107 printf("\tGive me a alphabet in lower case : ");
108 do {scanf("%c",&guess);} while ( getchar() != '\n' );
109 for (int i = 0; i < len; ++i){
110     if(word[i] == guess) {
111         found = true;
112         guessed[i] = guess;
113     }
114 }
115 if(!found) {
116     falseWord[mistakes] = guess;
117     mistakes += 1;
118 }win = strchr(guessed, '_');
119 }while(mistakes < CHANCE && win != NULL);
120 if(win == NULL) {
121     printf("\n");
122     printWord(guessed, len);
123     printf("\n\tCongrats! You have won : %s\n\n", word);
124 } else {
125     printf("\n");
126     printBody(mistakes, body);
127     printf("\n\tBetter try next time. Word was %s\n\n", word);
128 }
129 free(body);
130 free(word);
131 free(guessed);
132 return EXIT_SUCCESS;
133 }
```

Fig 5: Hangman Game code in C

OUTPUT SCREEN



The screenshot shows a web browser window with the URL `onlinegdb.com`. The terminal output is as follows:

```
input

Be aware you can be hanged!!

Rules :
- Maximum 6 mistakes are allowed.
- All alphabet are in lower case.
- All words are name of very popular Websites. eg. Google
- If you enjoy continue, otherwise close it.
Syntax : Alphabet
Example : a

Mistakes :0

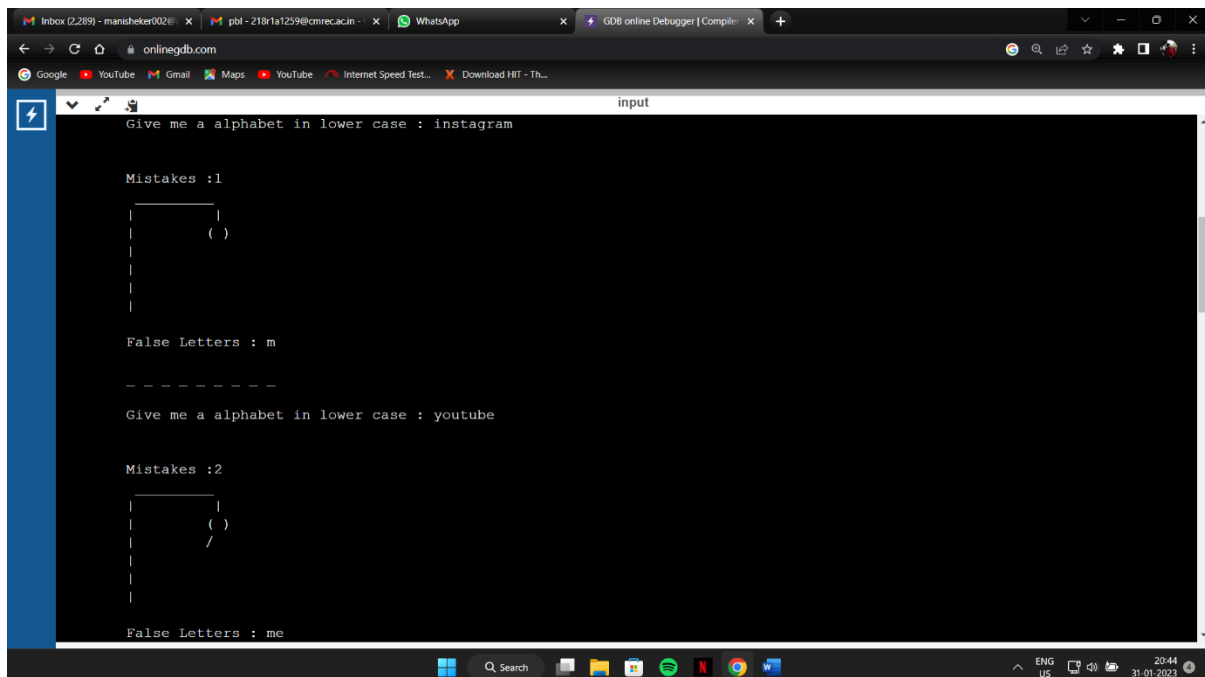
|
|
|

False Letters : None

-----

Give me a alphabet in lower case : instagram
```

Fig1: mistake 0



The screenshot shows the same web browser window, but the terminal output has progressed:

```
input

Give me a alphabet in lower case : instagram

Mistakes :1

|
|  ( )
|

False Letters : m

-----

Give me a alphabet in lower case : youtube

Mistakes :2

|
|  ( )
|  /
|

False Letters : me
```

Fig 2: mistake 1 and 2

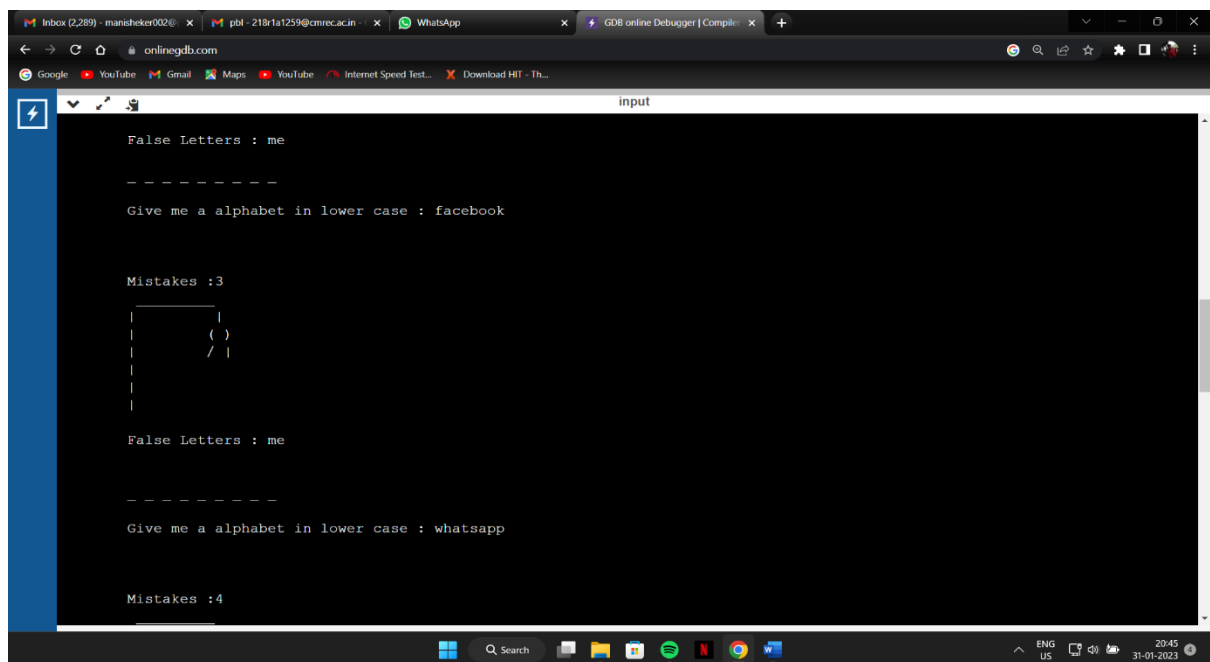


Fig 3: mistake 3 and 4

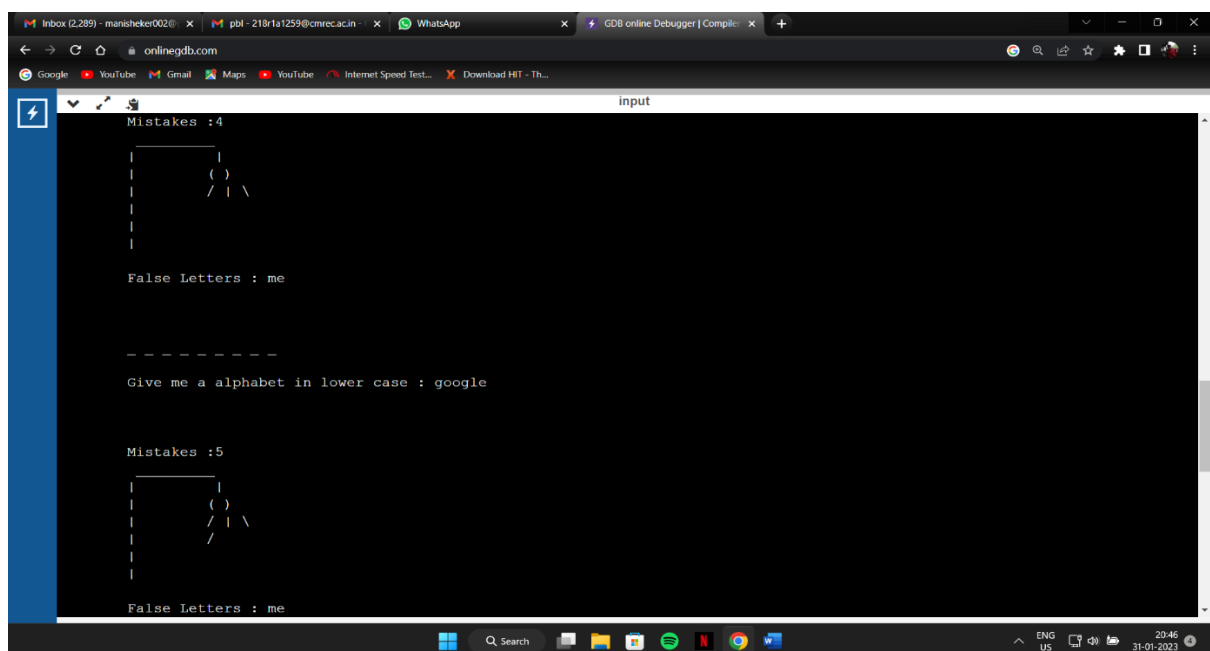


Fig 4: mistake 4 and 5

```
False Letters : me

Give me a alphabet in lower case : e

Mistakes :6

Better try next time. Word was skysports

...Program finished with exit code 0
Press ENTER to exit console.
```

Fig 5: mistake 6

CONCLUSION

The hangman program is a simple and classic word guessing game implemented in C programming language. The program uses the principles of encryption and decryption, random number generation, and input/output operations. The game has a set of rules and a maximum of 6 mistakes are allowed. The player has to guess the name of a popular website and if they exceed the maximum allowed mistakes, they lose the game. The game also features a visual representation of the hangman body, with each incorrect guess resulting in a part of the body being drawn. This program serves as an excellent educational tool for learning the basics of C programming, logic building and problem-solving skills.

REFERENCE

- <https://itsourcecode.com/free-projects/c-projects/hangman-game-in-c-with-source-code/>
- <https://www.cse.iitb.ac.in/~anwsha/cs699/mpStage1.html>
- <https://stackoverflow.com/questions/22877160/programming-hangman-in-c>
- <https://11points.com/11-strategies-dominating-hangman/>
- <https://gist.github.com/saroj22322/aa2f0849f33736395544c2d341ab3722>

THANK YOU!