

HANGMAN



Mohsin Ali Mirza

20K-0353

Ahmad Aleem

20K-0169

Syed Muhammad Faheem

20K-1054

COMPUTER ORGANIZATION AND LANGUAGES

**NATIONAL UNIVERSITY OF COMPUTER AND EMERGING
SCIENCES - FAST**

INTRODUCTION:

Implementing assembly language to make, childhood classic, hangman on a console based platform. The player has to guess the word before his/her lives run out and then loses the game.

LITERATURE REVIEW:

It is a very famous game, and we took inspiration from different string operations and tried to implement them in our game. Moreover, we also decided to use different animations to make the game more appealing and interesting.

PROBLEM DEFINITION:

Our program takes character inputs as guesses for the word needed to be guessed.

METHODOLOGY/SOLUTION STATEMENT:

The program will compare the input character with the word string. If any letters match it will output the letters onto the user's guess.

DETAILED DESIGN AND ARCHITECTURE:

main PROC	//All The Main Part
find_str PROC	//Finds The String In The Word List
make_array_dash PROC	//Makes The dashes for the word to be guessed
make_array_guess_letter PROC	//Makes The dashes with our guessed character
print_hangman_live PROC	//Printing/Animating The Current Status Of Hangman


```

31         BYTE "|           ",endl
32         BYTE "|      O      ",endl
33         BYTE "|     /|\     ",endl
34         BYTE "|     / \     ",endl
35         BYTE "+-----+ ",endl
36         BYTE "| YOU   WIN | ",endl
37         BYTE "+-----+ ",endl
38 messageSizeGoodGame DWORD ($-HANGMAN_GOODGAME_00)
39
40 HANGMAN_GOODGAME_01 LABEL BYTE
41         BYTE "+-----+ ",endl
42         BYTE "|      |      ",endl
43         BYTE "|      |      ",endl
44         BYTE "|      O_      ",endl
45         BYTE "|     /|      ",endl
46         BYTE "|     / \      ",endl
47         BYTE "+-----+ ",endl
48         BYTE "| YOU   WIN | ",endl
49         BYTE "+-----+ ",endl
50
51 HANGMAN_GOODGAME_02 LABEL BYTE
52         BYTE "+-----+ ",endl
53         BYTE "|      |      ",endl
54         BYTE "|      |      ",endl
55         BYTE "|      O/      ",endl
56         BYTE "|     /|      ",endl
57         BYTE "|     / \      ",endl
58         BYTE "+-----+ ",endl
59         BYTE "| YOU   WIN | ",endl
60         BYTE "+-----+ ",endl

```

```

61
62 HANGMAN_GOODGAME_03 LABEL BYTE
63     BYTE "+-----+      ",endl
64     BYTE "|          |      ",endl
65     BYTE "|          |      ",endl
66     BYTE "|          o_      ",endl
67     BYTE "|          /|      ",endl
68     BYTE "|          / \      ",endl
69     BYTE "+-----+      ",endl
70     BYTE "| YOU   WIN |      ",endl
71     BYTE "+-----+      ",endl
72
73 HANGMAN_GAMEOVER_00 LABEL BYTE
74     BYTE "+-----+      ",endl
75     BYTE "|          |      ",endl
76     BYTE "|          o      ",endl
77     BYTE "|          /|\      ",endl
78     BYTE "|          / \      ",endl
79     BYTE "|          |      ",endl
80     BYTE "+-----+      ",endl
81     BYTE "| YOU   DIE |      ",endl
82     BYTE "+-----+      ",endl
83
84 HANGMAN_GAMEOVER_01 LABEL BYTE
85     BYTE "+-----+      ",endl
86     BYTE "|          /      ",endl
87     BYTE "|          _o      ",endl
88     BYTE "|          _/\      ",endl
89     BYTE "|          \      ",endl
90     BYTE "|          |      ",endl

```

```

91         BYTE "+-----+ ",endl
92         BYTE "| YOU DIE | ",endl
93         BYTE "+-----+ ",endl
94
95     HANGMAN_GAMEOVER_02 LABEL BYTE
96         BYTE "+-----+ ",endl
97         BYTE "|         | ",endl
98         BYTE "|         O ",endl
99         BYTE "|        /|\ ",endl
100        BYTE "|        / \ ",endl
101        BYTE "|         ",endl
102        BYTE "+-----+ ",endl
103        BYTE "| YOU DIE | ",endl
104        BYTE "+-----+ ",endl
105
106    HANGMAN_GAMEOVER_03 LABEL BYTE
107        BYTE "+-----+ ",endl
108        BYTE "|         \ ",endl
109        BYTE "|         O_ ",endl
110        BYTE "|        /\_ ",endl
111        BYTE "|         / ",endl
112        BYTE "|         ",endl
113        BYTE "+-----+ ",endl
114        BYTE "| YOU DIE | ",endl
115        BYTE "+-----+ ",endl
116
117    HANGMAN_LIVES_06 LABEL BYTE
118        BYTE "+-----+ ",endl
119        BYTE "|         | ",endl
120        BYTE "|         | ",endl

```

```

121         BYTE "|",endl
122         BYTE "|",endl
123         BYTE "|",endl
124         BYTE "+-----+",endl
125         BYTE "|",endl
126         BYTE "+-----+",endl
127
128     HANGMAN_LIVES_05 LABEL BYTE
129         BYTE "+-----+",endl
130         BYTE "|",endl
131         BYTE "| 0",endl
132         BYTE "|",endl
133         BYTE "|",endl
134         BYTE "|",endl
135         BYTE "+-----+",endl
136         BYTE "|",endl
137         BYTE "+-----+",endl
138
139     HANGMAN_LIVES_04 LABEL BYTE
140         BYTE "+-----+",endl
141         BYTE "|",endl
142         BYTE "| 0",endl
143         BYTE "|",endl
144         BYTE "|",endl
145         BYTE "|",endl
146         BYTE "+-----+",endl
147         BYTE "|",endl
148         BYTE "+-----+",endl
149
150     HANGMAN_LIVES_03 LABEL BYTE

```

```

151             BYTE "+-----+      ",endl
152             BYTE "|          |      ",endl
153             BYTE "|          O      ",endl
154             BYTE "|          /|      ",endl
155             BYTE "|          ",endl
156             BYTE "|          ",endl
157             BYTE "+-----+      ",endl
158             BYTE "|          |      ",endl
159             BYTE "+-----+      ",endl
160
161  HANGMAN_LIVES_02 LABEL BYTE
162             BYTE "+-----+      ",endl
163             BYTE "|          |      ",endl
164             BYTE "|          O      ",endl
165             BYTE "|          /|\      ",endl
166             BYTE "|          ",endl
167             BYTE "|          ",endl
168             BYTE "+-----+      ",endl
169             BYTE "|          |      ",endl
170             BYTE "+-----+      ",endl
171
172  HANGMAN_LIVES_01 LABEL BYTE
173             BYTE "+-----+      ",endl
174             BYTE "|          |      ",endl
175             BYTE "|          O      ",endl
176             BYTE "|          /|\      ",endl
177             BYTE "|          /      ",endl
178             BYTE "|          ",endl
179             BYTE "+-----+      ",endl
180             BYTE "|          |      ",endl

```



```

181             BYTE "+-----+ ",endl
182
183 HANGMAN_LIVES_00 LABEL BYTE
184             BYTE "+-----+ ",endl
185             BYTE "|      | ",endl
186             BYTE "|      0 ",endl
187             BYTE "|      /|\ ",endl
188             BYTE "|      / \ ",endl
189             BYTE "|      ",endl
190             BYTE "+-----+ ",endl
191             BYTE "|      | ",endl
192             BYTE "+-----+ ",endl
193
194 ; random number what we generete
195 ranNum DWORD ?
196
197 ;All words what is posible to guess.
198 ;Pick by random generartor and put in selectedWords
199 manyWords  BYTE "BICYCLE", 0
200             BYTE "CANOE", 0
201             BYTE "SCATEBOARD", 0
202             BYTE "OFFSIDE", 0
203             BYTE "TENNIS", 0
204             BYTE "SOFTBALL", 0
205             BYTE "KNOCKOUT", 0
206             BYTE "CHALLENGE", 0
207             BYTE "SLALOM", 0
208             BYTE "MARATHON", 0
209             BYTE 0 ; End of list
210 len equ $ - manyWords

```

```

211
212 ; number what we make to know where are you in game
213 statusGameLive DWORD ?
214
215 ;Words what we select by random code
216 selectedWords BYTE " ", 0
217 ;Use as variable in function for length of Array
218 lengthArray DWORD ?
219
220 ;Letter what we guess, input from keyboard
221 guessLetter BYTE ?
222 ;Word what we print with -----,0
223 guessWords BYTE 50 DUP (?)
224 ;Array of guess Letter
225 guessLetterArray BYTE 50 DUP (?)
226 chardelete BYTE 'A'
227 ;Letter what are unknowns, change with -
228 letterDash BYTE '-'
229
230 drawDelay = 1000 ; delay 1 sec
231 var_loop BYTE 15 ; repeat 15 times
232
233 .code
234
235 main PROC
236
237 ; Get the console output handle:
238 INVOKE GetStdHandle, STD_OUTPUT_HANDLE
239 mov consoleHandle,eax

```

```

241  jump_game_start_again:
242
243      ; Write a string to the console:
244      INVOKE WriteConsole,
245          consoleHandle,          ;console output handle
246          ADDR message,          ; string pointer
247          messageSize,           ; string length
248          ADDR bytesWritten,     ; returns num bytes written
249          0                      ; not used
250
251      ;Part of code to generate random number from 0 until 9
252      mov  eax,10                ;get random 0 to 9
253      call Randomize            ;re-seed generator
254      call RandomRange
255      mov  ranNum,eax           ;save random number
256
257      ;call WriteDec
258      call Crlf                ;new line
259
260      ;Find a selectedWords base on generate ranNum from manyWords
261      mov  edx, ranNum          ;Index
262      call find_str             ;Returns EDI = pointer to string, we pick world
263
264      ;Copy find world in variable selectedWords
265      INVOKE Str_copy,
266          ADDR [edi],
267          ADDR selectedWords
268
269      ;Print selectedWords on screen
270      ;mov edx, offset selectedWords

```

```

271     ;call WriteString
272     ;call Crlf           ;new line
273
274     ;Make array of dash. It would be world what we guess
275     call make_array_dash
276
277     ;InicIALIZATION number of life what you have
278     mov statusGameLive, 6
279
280 again_input_world:
281
282     ;Print figure depending on the number of lives
283     call print_hangman_live
284
285     ;Check if you have more live. If player lost all lives, game is over
286     cmp statusGameLive, 0
287     je loop_game_over
288
289     mov eax,green+(black*16)
290     call SetTextColor
291
292     mWrite <"Guess a letter: ">
293
294     call readChar ;User inputs char
295     cmp al, 27    ;Check if is press ESC
296     je exit_main  ;YES, end game
297     cmp al, 32    ;Check if is press SPACE
298     je restart_game ;YES, restart game
299     and al, 0DFH  ;Convert lowercase input to uppercase.
300
301     ;If uppercase, it remains uppercase
302     push eax
303     sub al, 'A'   ;checks if it is a letter
304     cmp al, 'Z'-'A'
305     jbe uppercase
306     jmp again_input_world
307 uppercase:
308     pop eax
309     mov guessLetter, al
310     call WriteChar
311     call Crlf     ;new line
312     call Crlf     ;new line
313
314     mov eax,white+(black*16)
315     call SetTextColor
316
317
318     ;Check if letter is already guessed
319     mov ecx, LENGTHOF guessLetterArray
320     mov edi, offset guessLetterArray
321     mov al, guessLetter ; Load character to find
322     repne scasb         ; Search
323     je loop_guess_letter_exists ; Letter already exist
324
325     call make_array_guess_letter
326
327
328     ;Check if letter is in selectedWords. If not take life
329     mov ecx, LENGTHOF selectedWords
330

```

```

331     mov edi, offset selectedWords
332     mov al, guessLetter           ; Load character to find
333     repne scasb                   ; Search
334     jne loop_take_live           ; Letter exist take life
335
336
337     ; We are making new array, guess letter whange dash on right pleave
338     mov esi, offset selectedWords ; Source
339     mov edi, offset guessWords    ; Destination
340     mov ecx, LENGTHOF selectedWords ; Number of bytes to check
341     mov al, guessLetter           ; Search for that character
342     xor ebx, ebx                  ; Index EBX = 0
343
344     ride_hard_loop:
345         cmp [esi+ebx], al         ; Compare memory/register
346         jne @F                   ; Skip next line if no match
347         mov [edi+ebx], al         ; Hang 'em lower
348         @@:
349         inc ebx                   ; Increment pointer
350         dec ecx                   ; Decrement counter
351         jne ride_hard_loop        ; Jump if ECX != 0
352
353
354     ;Is there more letter to guess of we finish
355     mov ecx, LENGTHOF guessWords
356     mov edi, offset guessWords
357     mov al, letterDash           ; Load character to find
358     repne scasb                   ; Search
359     jne loop_game_win            ; No more letter
360     jmp again_input_world        ; Guess next world

```

```

363 exit_main:
364
365     INVOKE ExitProcess,0
366
367 loop_guess_letter_exists:
368
369     mov  eax,red+(black*16)
370     call SetTextColor
371
372     mWrite <"Sorry, you already guessed letter, ">
373     mov  al, guessLetter
374     call WriteChar
375     call Crlf                ; new line
376     mWrite <"I repeat you one more time the letter what you guessed. ">
377     call Crlf                ; new line
378     mWrite <"Guessed letter are: ">
379     mov  edx, offset guessLetterArray
380     call WriteString          ; write a string pointed to by EDX
381     call Crlf                ; new line
382     call Crlf                ; new line
383
384     mov  eax,white+(black*16)
385     call SetTextColor
386
387     jmp  again_input_world    ; Guess next letter
388
389 loop_take_live:
390

```

```

391         dec statusGameLive
392         jmp again_input_world             ; Guess next letter
393
394 restart_game:
395
396         INVOKE Str_trim, ADDR guessLetterArray, ','
397
398         mov  edx, OFFSET guessLetterArray
399         call StrLength
400         mov  lengthArray, eax
401
402         mov  edi, offset guessLetterArray ; Destination
403         add  edi, lengthArray
404         dec  edi
405         ;INVOKE Str_trim, ADDR guessLetterArray, guessLetter
406         INVOKE Str_trim, ADDR guessLetterArray, [edi]
407
408         cmp  edi, offset guessLetterArray
409         jne  restart_game
410
411         ;Return white color again
412         mov  eax,white+(black*16)
413         call SetTextColor
414         call Crlf             ;new line
415
416         jmp  jump_game_start_again       ; Guess next letter
417
418 loop_game_win:
419
420         mGotoxy 0, 15

```

```

421
422 ; Write a string to the console:
423 INVOKE WriteConsole,
424     consoleHandle,           ;console output handle
425     ADDR HANGMAN_GOODGAME_00, ; string pointer
426     messageSizeGoodGame,     ; string length
427     ADDR bytesWritten,       ; returns num bytes written
428     0                       ; not used
429
430     mov eax, drawDelay
431     call Delay
432     mGotoxy 0, 15
433     mov  eax,green+(black*16)
434     call SetTextColor
435
436 ; Write a string to the console:
437 INVOKE WriteConsole,
438     consoleHandle,           ;console output handle
439     ADDR HANGMAN_GOODGAME_01, ; string pointer
440     messageSizeGoodGame,     ; string length
441     ADDR bytesWritten,       ; returns num bytes written
442     0                       ; not used
443
444     mov eax, drawDelay
445     call Delay
446     mGotoxy 0, 15
447     mov  eax,yellow+(black*16)
448     call SetTextColor
449
450 ; Write a string to the console:

```



```

451     INVOKE WriteConsole,
452         consoleHandle,           ;console output handle
453         ADDR HANGMAN_GOODGAME_02, ; string pointer
454         messageSizeGoodGame,     ; string length
455         ADDR bytesWritten,       ; returns num bytes written
456         0                       ; not used
457
458     mov eax, drawDelay
459     call Delay
460     mGotoxy 0, 15
461     mov  eax,cyan+(black*16)
462     call SetTextColor
463
464 ; Write a string to the console:
465     INVOKE WriteConsole,
466         consoleHandle,           ;console output handle
467         ADDR HANGMAN_GOODGAME_03, ; string pointer
468         messageSizeGoodGame,     ; string length
469         ADDR bytesWritten,       ; returns num bytes written
470         0                       ; not used
471
472     mov eax, drawDelay
473     call Delay
474     mGotoxy 0, 15
475     mov  eax,red+(black*16)
476     call SetTextColor
477
478     dec var_loop
479     cmp var_loop, 0
480     jne loop_game_win

```

```

481
482     ;restart game after 4*15sekunds
483     jmp jump_game_start_again
484
485
486 loop_game_over:
487
488     mGotoxy 0, 15
489
490     ; Write a string to the console:
491     INVOKE WriteConsole,
492         consoleHandle,           ;console output handle
493         ADDR Hangman_GameOver_00, ; string pointer
494         messageSizeGoodGame,      ; string length
495         ADDR bytesWritten,        ; returns num bytes written
496         0                        ; not used
497
498     mov eax, drawDelay
499     call Delay
500     mGotoxy 0, 15
501     mov eax, green+(black*16)
502     call SetTextColor
503
504     ; Write a string to the console:
505     INVOKE WriteConsole,
506         consoleHandle,           ;console output handle
507         ADDR Hangman_GameOver_01, ; string pointer
508         messageSizeGoodGame,      ; string length
509         ADDR bytesWritten,        ; returns num bytes written
510         0                        ; not used

```

```

512     mov eax, drawDelay
513     call Delay
514     mGotoxy 0, 15
515     mov  eax,yellow+(black*16)
516     call SetTextColor
517
518     ; Write a string to the console:
519     INVOKE WriteConsole,
520         consoleHandle,                ;console output handle
521         ADDR HANGMAN_GAMEOVER_02,    ; string pointer
522         messageSizeGoodGame,         ; string length
523         ADDR bytesWritten,           ; returns num bytes written
524         0                             ; not used
525
526     mov eax, drawDelay
527     call Delay
528     mGotoxy 0, 15
529     mov  eax,cyan+(black*16)
530     call SetTextColor
531
532     ; Write a string to the console:
533     INVOKE WriteConsole,
534         consoleHandle,                ;console output handle
535         ADDR HANGMAN_GAMEOVER_03,    ; string pointer
536         messageSizeGoodGame,         ; string length
537         ADDR bytesWritten,           ; returns num bytes written
538         0                             ; not used
539
540     mov eax, drawDelay
541     call Delay

```

```

542     mGotoxy 0, 15
543     mov  eax,red+(black*16)
544     call SetTextColor
545
546     dec  var_loop
547     cmp  var_loop, 0
548     jne  loop_game_over
549
550     ;restar game after 4*15sekunds
551     mov  eax,white+(black*16)
552     call SetTextColor
553     jmp  jump_game_start_again
554
555
556 main ENDP
557
558 find_str PROC                                ; ARG: EDX = index
559     lea  edi, manyWords                    ; Address of string list
560
561     mov  ecx, len                          ; Maximal number of bytes to scan
562     xor  al, al                            ; Scan for 0
563
564     @@:
565     sub  edx, 1
566     jc  done                               ; No index left to scan = string found
567     repne scasb                            ; Scan for AL
568     jmp  @B                                ; Next string
569
570 done:
571     ret

```

```

573 find_str ENDP                ; RESULT: EDI pointer to string[edx]
574
575 make_array_dash PROC
576     mov     edx,OFFSET selectedWords
577     call    StrLength          ; Length of a null-terminated string pointed to by EDX
578     mov     lengthArray,eax
579
580     mov     al, '-'            ; Default character for guessWords
581     mov     ecx, lengthArray    ; REP counter
582     mov     edi, offset guessWords ; Destination
583     rep     stosb              ; Build guessWords
584     mov     BYTE PTR [edi], 0   ; Store the null termination
585
586     ret
587 make_array_dash ENDP
588
589 make_array_guess_letter PROC
590     mov     edx, OFFSET guessLetterArray
591     call    StrLength          ; Length of a null-terminated string pointed to by EDX
592     mov     lengthArray, eax
593
594     mov     edi, offset guessLetterArray ; Destination
595     add     edi, lengthArray
596     mov     al, guessLetter
597     mov     BYTE PTR [edi], al   ; Store guessLetter
598     inc     edi
599     mov     BYTE PTR [edi], ','   ; Store the null termination
600
601     ret
602 make_array_guess_letter ENDP

```

```

604 print_hangman_live PROC
605
606     mov eax, statusGameLive
607
608     cmp eax, 6
609     je live_6
610     cmp eax, 5
611     je live_5
612     cmp eax, 4
613     je live_4
614     cmp eax, 3
615     je live_3
616     cmp eax, 2
617     je live_2
618     cmp eax, 1
619     je live_1
620     cmp eax, 0
621     je live_0
622
623 live_6:    ; Write a string to the console:
624     INVOKE WriteConsole,
625         consoleHandle,          ; console output handle
626         ADDR HANGMAN_LIVES_06, ; string pointer
627         messageSizeGoodGame,    ; string length
628         ADDR bytesWritten,      ; returns num bytes written
629         0,                     ; not used
630     call Crlf                  ; new line
631     call Crlf                  ; new line
632     mov edx, offset guessWords
633     call WriteString           ; write a string pointed to by EDX

```

```

634     call Crlf                ; new line
635     call Crlf                ; new line
636     mWrite <"Guessed letter are: ">
637     mov edx, offset guessLetterArray
638     call WriteString          ; write a string pointed to by EDX
639     call Crlf                ; new line
640     call Crlf                ; new line
641     ret
642
643 live_5:    ; Write a string to the console:
644     INVOKE WriteConsole,
645         consoleHandle,        ; console output handle
646         ADDR HANGMAN_LIVES_05, ; string pointer
647         messageSizeGoodGame,   ; string length
648         ADDR bytesWritten,     ; returns num bytes written
649         0                     ; not used
650     call Crlf                ; new line
651     call Crlf                ; new line
652     mov edx, offset guessWords
653     call WriteString          ; write a string pointed to by EDX
654     call Crlf                ; new line
655     call Crlf                ; new line
656     mWrite <"Guessed letter are: ">
657     mov edx, offset guessLetterArray
658     call WriteString          ; write a string pointed to by EDX
659     call Crlf                ; new line
660     call Crlf                ; new line
661     ret
662
663 live_4:    ; Write a string to the console:

```

```

664     INVOKE WriteConsole,
665         consoleHandle,           ;console output handle
666         ADDR HANGMAN_LIVES_04,   ; string pointer
667         messageSizeGoodGame,     ; string length
668         ADDR bytesWritten,       ; returns num bytes written
669         0,                       ; not used
670     call Crlf                    ; new line
671     call Crlf                    ; new line
672     mov edx, offset guessWords
673     call WriteString             ; write a string pointed to by EDX
674     call Crlf                    ; new line
675     call Crlf                    ; new line
676     mWrite <"Guessed letter are: ">
677     mov edx, offset guessLetterArray
678     call WriteString             ; write a string pointed to by EDX
679     call Crlf                    ; new line
680     call Crlf                    ; new line
681     ret
682
683 live_3:    ; Write a string to the console:
684     INVOKE WriteConsole,
685         consoleHandle,           ;console output handle
686         ADDR HANGMAN_LIVES_03,   ; string pointer
687         messageSizeGoodGame,     ; string length
688         ADDR bytesWritten,       ; returns num bytes written
689         0,                       ; not used
690     call Crlf                    ; new line
691     call Crlf                    ; new line
692     mov edx, offset guessWords
693     call WriteString             ; write a string pointed to by EDX

```



```

694     call Crlf                ; new line
695     call Crlf                ; new line
696     mWrite <"Guessed letter are: ">
697     mov edx, offset guessLetterArray
698     call WriteString          ; write a string pointed to by EDX
699     call Crlf                ; new line
700     call Crlf                ; new line
701     ret
702
703 live_2: ; Write a string to the console:
704     INVOKE WriteConsole,
705         consoleHandle,        ;console output handle
706         ADDR HANGMAN_LIVES_02, ; string pointer
707         messageSizeGoodGame,   ; string length
708         ADDR bytesWritten,     ; returns num bytes written
709         0                     ; not used
710     call Crlf                ; new line
711     call Crlf                ; new line
712     mov edx, offset guessWords
713     call WriteString          ; write a string pointed to by EDX
714     call Crlf                ; new line
715     call Crlf                ; new line
716     mWrite <"Guessed letter are: ">
717     mov edx, offset guessLetterArray
718     call WriteString          ; write a string pointed to by EDX
719     call Crlf                ; new line
720     call Crlf                ; new line
721     ret
722
723 live_1: ; Write a string to the console:

```

```

724     INVOKE WriteConsole,
725         consoleHandle,          ;console output handle
726         ADDR HANGMAN_LIVES_01, ; string pointer
727         messageSizeGoodGame,    ; string length
728         ADDR bytesWritten,      ; returns num bytes written
729         0                      ; not used
730     call Crlf                   ; new line
731     call Crlf                   ; new line
732     mov edx, offset guessWords
733     call WriteString            ; write a string pointed to by EDX
734     call Crlf                  ; new line
735     call Crlf                  ; new line
736     mWrite <"Guessed letter are: ">
737     mov edx, offset guessLetterArray
738     call WriteString            ; write a string pointed to by EDX
739     call Crlf                  ; new line
740     call Crlf                  ; new line
741     ret
742
743 live_0: ; Write a string to the console:
744     INVOKE WriteConsole,
745         consoleHandle,          ;console output handle
746         ADDR HANGMAN_LIVES_00, ; string pointer
747         messageSizeGoodGame,    ; string length
748         ADDR bytesWritten,      ; returns num bytes written
749         0                      ; not used
750     call Crlf                   ; new line
751     call Crlf                   ; new line
752     mov edx, offset guessWords
753     call WriteString            ; write a string pointed to by EDX
754     call Crlf                   ; new line
755     call Crlf                   ; new line
756     mWrite <"Guessed letter are: ">
757     mov edx, offset guessLetterArray
758     call WriteString            ; write a string pointed to by EDX
759     call Crlf                  ; new line
760     call Crlf                  ; new line
761     ret
762
763 print_hangman_live ENDP
764
765 END main

```

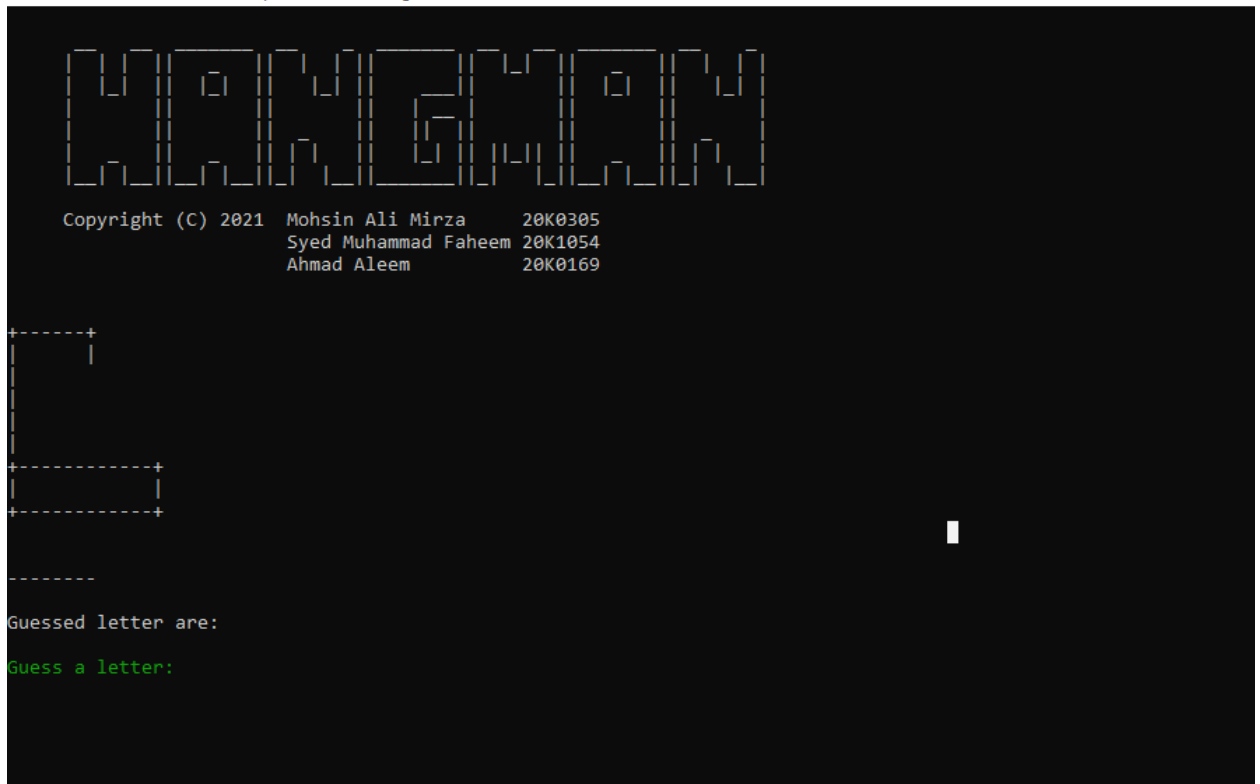
**RESULTS SOFTWARE SIMULATION AND DISCUSSION
(INCLUDE AT LEAST ALL POSSIBLE TEST CASES WITH
PICTURES OF YOUR RESULT):**

Word List:

```
manyWords BYTE "BICYCLE", 0
           BYTE "CANOE", 0
           BYTE "SCATEBOARD", 0
           BYTE "OFFSIDE", 0
           BYTE "TENNIS", 0
           BYTE "SOFTBALL", 0
           BYTE "KNOCKOUT", 0
           BYTE "CHALLENGE", 0
           BYTE "SLALOM", 0
           BYTE "MARATHON", 0
           BYTE 0 ; End of
len equ $ - manyWords
```

If You Lose:

Select C:\Users\acer\source\repos\Practice\Debug\Practice.exe



C:\Users\acer\source\repos\Practice\Debug\Practice.exe

```
|
+-----+
```

```
-----
```

Guessed letter are:

Guess a letter: A

```
+-----+
|         |
|         |
```

```
+-----+
|         |
+-----+
```

```
-----A--
```

Guessed letter are: A,

Guess a letter:

C:\Users\acer\source\repos\Practice\Debug\Practice.exe

Guessed letter are: A,E,

Guess a letter: Z

-----A-----


Guessed letter are: A,E,Z,

Guess a letter: P

-----A-----

Guessed letter are: A,E,Z,P,

Guess a letter: +-----+



```
Guess a letter: 0
```

A diagram showing a rectangular box with dashed lines. A vertical line segment is drawn inside the box, labeled '0'.

-0---A--

Guessed letter are: A,E,Z,P,O,

```
Guess a letter: L
```

A diagram showing a rectangular box with dashed lines. A vertical line segment is drawn inside the box, labeled with the number '0'.

-0---ALL

Guessed letter are: A,E,Z,P,O,L,

Guess a letter: Q

```

+-----+
|      |
|  O   |
| / \  |
| / \  |
|      |
+-----+
| YOU DIE |
+-----+

```

Gussed letter are:

 Select C:\Users\acer\source\repos\Practice\Debug\Practice.exe

```

+-----+
|       |
+-----+
|       |
+-----+
|       |
+-----+
|       |
+-----+
|       |
+-----+

```

Guessed letter are:

Guess a letter:

C:\Users\acer\source\repos\Practice\Debug\Practice.exe

```
|
|
+-----+
|         |
+-----+
```

Guessed letter are:

Guess a letter: A

```
+-----+
|         |
|         |
|         |
+-----+
|         |
+-----+
```

--A-----

Guessed letter are: A,

Guess a letter:

C:\Users\acer\source\repos\Practice\Debug\Practice.exe

```
+-----+
|       |
+-----+
```

Guessed letter are:

Guess a letter: A

```
+-----+
|       |
|       |
|       |
+-----+
|       |
+-----+
```

--A-----

Guessed letter are: A,

Guess a letter: C

```
+-----+
|       |
|       |
|       |
+-----+
|       |
+-----+
```

C-A-----

Guessed letter are: A,C,

C:\Users\acer\source\repos\Practice\Debug\Practice.ex

```
+-----+
|       |
+-----+
```

--A-----

Guessed letter are: A,

Guess a letter: C

```
+-----+
|       |
|       |
|       |
+-----+
|       |
+-----+
```

C-A-----

Guessed letter are: A,C,

Guess a letter: H

```
+-----+
|       |
|       |
|       |
+-----+
|       |
+-----+
```

CHA-----

Guessed letter are: A,C,H,

C:\Users\acer\source\repos\Practice\Debug\Practice.exe

```
-----+
|
-----+
```

C-A-----

Guessed letter are: A,C,

Guess a letter: H

```
-----+
|
-----+
```

```
-----+
|
-----+
```

CHA-----

Guessed letter are: A,C,H,

Guess a letter: L

```
-----+
|
-----+
```

```
-----+
|
-----+
```

CHALL----

Guessed letter are: A,C,H,L,

Guess a letter:

C:\Users\acer\source\repos\Practice\Debug\Practice.exe

```
+-----+  
|       |  
+-----+
```

CHA-----

Guessed letter are: A,C,H,

Guess a letter: L

```
+-----+  
|       |  
|       |  
|       |  
+-----+  
|       |  
+-----+
```

CHALL----

Guessed letter are: A,C,H,L,

Guess a letter: E

```
+-----+  
|       |  
|       |  
|       |  
+-----+  
|       |  
+-----+
```

CHALLE--E

Guessed letter are: A,C,H,L,E,

Guess a letter:

C:\Users\acer\source\repos\Practice\Debug\Practice.

```
+-----+  
|       |  
+-----+
```

CHALL----

Guessed letter are: A,C,H,L,

Guess a letter: E

```
+-----+  
|       |  
|       |  
|       |  
+-----+  
|       |  
+-----+
```

CHALLE--E

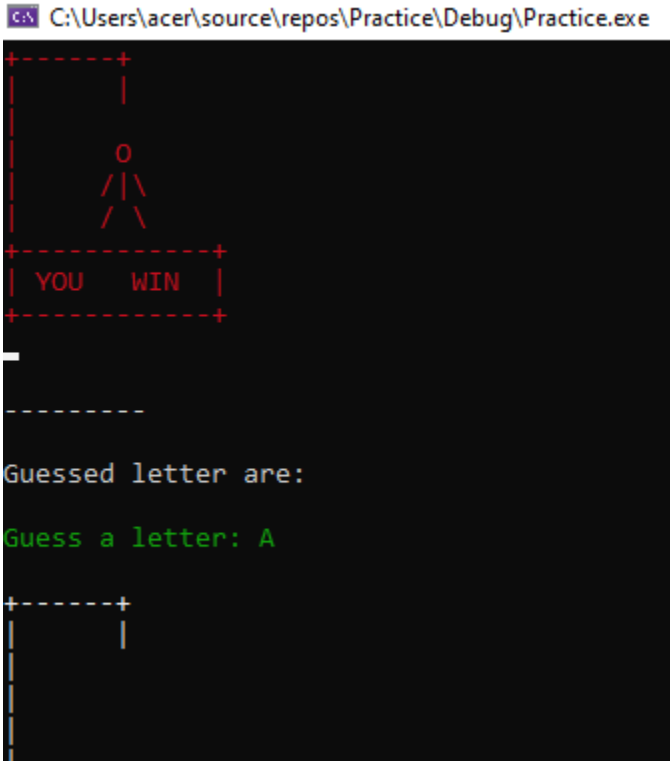
Guessed letter are: A,C,H,L,E,

Guess a letter: N

```
+-----+  
|       |  
|       |  
|       |  
+-----+  
|       |  
+-----+
```

CHALLENG--E

Guessed letter are: A,C,H,L,E,N,



CONCLUSION, COST AND FUTURE WORK:

We were able to achieve our core goal of our project which was to learn the basic fundamentals of game development and create a soothing and engaging game for the user using a low level language.

REFERENCES:

[https://en.wikipedia.org/wiki/Hangman_\(game\)](https://en.wikipedia.org/wiki/Hangman_(game))

<https://csc.csudh.edu/mmccullough/asm/help/index.html?page=source%2Fabout.htm>