**PASSWORD STRENGTH CHECKER AND VALIDATOR**

|  |
| --- |
| .386  INCLUDE Irvine32.inc  .data  title\_msg db "==========================================", 0  welcome\_msg db " PASSWORD STRENGTH CHECKER AND VALIDATOR ", 0  line\_break db "==========================================", 0  username\_prompt db "Enter your username: ", 0  prompt db "Enter your password: ", 0  weak\_msg db "[Weak] Password is too simple! Enter a stronger one.", 0  medium\_msg db "[Medium] Your password is okay. Do you want to continue? (Y/N): ", 0  strong\_msg db "[Strong] Great! Your password is secure.", 0  confirm\_msg db "Confirm your password: ", 0  success\_msg db "Password matched! Account created.", 0  fail\_msg db "Passwords do not match. Try again!", 0  newline db 10, 13, 0  colon db ":", 0  user\_input db 20 dup(?)  confirmed\_password db 20 dup(?)  user\_username db 50 dup(?)  pass\_length DWORD 0  has\_upper DWORD 0  has\_lower DWORD 0  has\_digit DWORD 0  has\_special DWORD 0  starts\_with\_upper DWORD 0  user\_choice db ?  common\_passwords db "123456",0, "password",0, "qwerty",0, "abc123",0, "123456",0, "123123",0, "admin",0  filename db "accounts.txt", 0  fileHandle DWORD ?  bytesWritten DWORD ?  existing\_data db 2048 dup(?) ; buffer to hold existing contents  read\_handle DWORD ?  read\_bytes DWORD ?  exit\_prompt db "Do you want to create another account? (Y/N): ", 0  duplicate\_msg db "Username already exists! Try a different one.", 0  .code  main PROC  call Clrscr  mov edx, OFFSET title\_msg  call WriteString  call Crlf  mov edx, OFFSET welcome\_msg  call WriteString  call Crlf  mov edx, OFFSET line\_break  call WriteString  call Crlf  input\_username:  mov edx, OFFSET username\_prompt  call WriteString  mov edx, OFFSET user\_username  mov ecx, 50  call ReadString  ; === CHECK IF USERNAME ALREADY EXISTS ===  call check\_duplicate\_username  cmp eax, 1  je username\_exists  jmp get\_password  username\_exists:  call Crlf  mov edx, OFFSET duplicate\_msg  call WriteString  call Crlf  jmp input\_username  get\_password:  mov edx, OFFSET prompt  call WriteString  mov edx, OFFSET user\_input  mov ecx, 20  call ReadString  mov pass\_length, eax  mov has\_upper, 0  mov has\_lower, 0  mov has\_digit, 0  mov has\_special, 0  mov starts\_with\_upper, 0  mov esi, OFFSET user\_input  movzx eax, byte ptr [esi]  cmp eax, 'A'  jl check\_chars  cmp eax, 'Z'  jg check\_chars  mov starts\_with\_upper, 1  check\_chars:  mov esi, OFFSET user\_input  mov ecx, pass\_length  check\_loop:  cmp ecx, 0  je done\_check  mov al, [esi]    cmp al, 'A'  jl check\_lower  cmp al, 'Z'  jg check\_lower  mov has\_upper, 1  jmp next\_char  check\_lower:  cmp al, 'a'  jl check\_digit  cmp al, 'z'  jg check\_digit  mov has\_lower, 1  jmp next\_char  check\_digit:  cmp al, '0'  jl check\_special  cmp al, '9'  jg check\_special  mov has\_digit, 1  jmp next\_char  check\_special:  cmp al, ' '  jbe next\_char  cmp al, '/'  jbe mark\_special  cmp al, '9'  jbe next\_char  cmp al, '@'  jbe mark\_special  cmp al, 'Z'  jbe next\_char  cmp al, '`'  jbe mark\_special  cmp al, 'z'  jbe next\_char  cmp al, '~'  jbe mark\_special  jmp next\_char  mark\_special:  mov has\_special, 1  next\_char:  inc esi  dec ecx  jmp check\_loop  done\_check:  call check\_common\_passwords  cmp eax, 1  je handle\_weak  mov eax, pass\_length  cmp eax, 8  jl handle\_weak  cmp starts\_with\_upper, 1  jne check\_medium  cmp has\_digit, 1  jne check\_medium  cmp has\_special, 1  jne check\_medium  jmp handle\_strong  check\_medium:  cmp has\_upper, 1  jne handle\_weak  cmp has\_lower, 1  jne handle\_weak  cmp has\_digit, 1  jne handle\_weak  jmp handle\_medium  handle\_medium:  call Crlf  mov edx, OFFSET medium\_msg  call WriteString  call ReadChar  mov user\_choice, al  call Crlf  cmp user\_choice, 'Y'  je confirm\_password  cmp user\_choice, 'y'  je confirm\_password  jmp get\_password  handle\_strong:  call Crlf  mov edx, OFFSET strong\_msg  call WriteString  call Crlf  jmp confirm\_password  handle\_weak:  call Crlf  mov edx, OFFSET weak\_msg  call WriteString  call Crlf  jmp get\_password  confirm\_password:  call Crlf  mov edx, OFFSET confirm\_msg  call WriteString  mov edx, OFFSET confirmed\_password  mov ecx, 20  call ReadString  mov esi, OFFSET user\_input  mov edi, OFFSET confirmed\_password  compare\_confirm:  mov al, [esi]  mov bl, [edi]  cmp al, bl  jne password\_mismatch  cmp al, 0  je password\_match  inc esi  inc edi  jmp compare\_confirm  password\_mismatch:  call Crlf  mov edx, OFFSET fail\_msg  call WriteString  call Crlf  jmp confirm\_password  password\_match:  call Crlf  mov edx, OFFSET success\_msg  call WriteString  call Crlf  ; === READ EXISTING FILE CONTENT IF EXISTS ===  mov edx, OFFSET filename  call OpenInputFile  mov read\_handle, eax  cmp eax, INVALID\_HANDLE\_VALUE  je skip\_file\_read  ; Read contents into existing\_data  mov eax, read\_handle  mov edx, OFFSET existing\_data  mov ecx, 2048  call ReadFromFile  mov read\_bytes, eax  ; Close read file  mov eax, read\_handle  call CloseFile  skip\_file\_read:  ; === OPEN FILE FOR WRITING ===  mov edx, OFFSET filename  call CreateOutputFile  mov fileHandle, eax  ; === WRITE BACK OLD CONTENT ===  mov eax, fileHandle  mov edx, OFFSET existing\_data  mov ecx, read\_bytes  call WriteToFile  ; === WRITE NEW USERNAME ===  mov eax, fileHandle  mov edx, OFFSET user\_username  call WriteStringToFile  ; Write colon  mov edx, OFFSET colon  call WriteStringToFile  ; Write password  mov edx, OFFSET user\_input  call WriteStringToFile  ; Write newline  mov eax, fileHandle  mov edx, OFFSET newline  mov ecx, 2  call WriteToFile  ; Close file  mov eax, fileHandle  call CloseFile  ; Ask if user wants to add another account  call Crlf  mov edx, OFFSET exit\_prompt  call WriteString  call ReadChar  call Crlf  cmp al, 'Y'  je main  cmp al, 'y'  je main  jmp done  done:  call Crlf  mov edx, OFFSET line\_break  call WriteString  call Crlf  exit  WriteStringToFile PROC  pusha  mov ecx, 0  mov esi, edx  count\_loop:  cmp byte ptr [esi+ecx], 0  je count\_done  inc ecx  jmp count\_loop  count\_done:  call WriteToFile  popa  ret  WriteStringToFile ENDP  check\_common\_passwords PROC  mov esi, OFFSET common\_passwords  next\_common:  push esi  mov edi, OFFSET user\_input  compare\_loop:  mov al, [esi]  mov bl, [edi]  cmp al, bl  jne not\_match  cmp al, 0  je weak\_found  inc esi  inc edi  jmp compare\_loop  not\_match:  pop esi  skip\_to\_next:  cmp byte ptr [esi], 0  je after\_skip  inc esi  jmp skip\_to\_next  after\_skip:  inc esi  cmp byte ptr [esi], 0  je not\_found  jmp next\_common  weak\_found:  pop esi  mov eax, 1  ret  not\_found:  mov eax, 0  ret  check\_common\_passwords ENDP  check\_duplicate\_username PROC  ; Check if user\_username exists in existing\_data  ; eax = 1 if exists, 0 if not  mov esi, OFFSET existing\_data ; buffer with all lines  mov edi, OFFSET user\_username ; username to check  search\_loop:  ; Check end of buffer  cmp byte ptr [esi], 0  je not\_found  ; Compare current string with username  push esi  push edi  call compare\_strings  pop edi  pop esi  cmp eax, 1  je found  ; Skip to next line  skip\_line:  cmp byte ptr [esi], 0  je not\_found  cmp byte ptr [esi], 10  je skip\_crlf  inc esi  jmp skip\_line  skip\_crlf:  inc esi  cmp byte ptr [esi], 13  je skip\_inc  inc esi  jmp search\_loop  skip\_inc:  inc esi  jmp search\_loop  found:  mov eax, 1  ret  not\_found:  mov eax, 0  ret  check\_duplicate\_username ENDP  compare\_strings PROC  ; ESI = ptr to current line in file  ; EDI = ptr to username  ; EAX = 1 if match, 0 if not  push ecx  mov ecx, 0  compare\_loop:  mov al, [esi+ecx]  cmp al, ':'  je match\_check  cmp al, 0  je no\_match  mov bl, [edi+ecx]  cmp al, bl  jne no\_match  inc ecx  jmp compare\_loop  match\_check:  mov al, [edi+ecx]  cmp al, 0  jne no\_match  mov eax, 1  pop ecx  ret  no\_match:  mov eax, 0  pop ecx  ret  compare\_strings ENDP  main ENDP  END main |