**COAL PROJECT**

**TASMIA ANWAR 131**

**AMINA ARIF 148**

**AMNA AMJAD 165**

**HAFSA RAZZAQ 107**

include irvine32.inc

include macros.inc

bufsize=500

.data

productID dd 100 dup(?)

productQuantity dd 100 dup(?)

totalItems dd 0

itemName byte 20 dup(?)

filename byte "inventorydata.txt",0

filehandle handle ?

buffer byte bufsize dup(?)

.code

main proc

call crlf

call menu

call crlf

exit

main endp

menu proc

call crlf

mwrite " ==============================================="

call crlf

mwrite " ------------------MAIN MENU--------------------"

call crlf

mwrite " ==============================================="

call crlf

mwrite " 1. Add Item"

call crlf

mwrite " 2. Remove Item"

call crlf

mwrite " 3. Update Item Quantity"

call crlf

mwrite " 4. Search Item"

call crlf

mwrite " 5. Display All Data"

call crlf

mwrite " 6. Clear All Data"

call crlf

mwrite " 7. View Total Items"

call crlf

mwrite " 8. Display Total Quantity of All Products"

call crlf

mwrite " 9. Sort Products"

call crlf

mwrite " 10. Check for Restock"

call crlf

mwrite " 11. Exit System"

call crlf

mwrite " Enter your choice: "

call readint

.if (eax == 1)

call addItem

.elseif (eax == 2)

call removeItem

.elseif (eax == 3)

call updateItem

.elseif (eax == 4)

call searchItem

.elseif (eax == 5)

call displayAllData

.elseif (eax == 6)

call clearAllData

.elseif (eax == 7)

call viewTotalItems

.elseif (eax == 8)

call displayTotalQuantity

.elseif (eax == 9)

call sortProducts

.elseif (eax == 10)

call checkRestock

.elseif (eax == 11)

call exitSystem

.else

call crlf

mwrite " Invalid choice, try again."

call crlf

call menu

.endif

menu endp

addItem proc

push ebp

mov ebp, esp

call clrscr

call crlf

mwrite " How many items would you like to add? "

call readint

mov ecx, eax

cmp ecx, 0

jle doneAdding

addItemLoop:

call crlf

mwrite " Enter Product ID: "

call readint

mov ebx, totalItems

mov [productID + ebx \* 4], eax

call crlf

mwrite " Enter Product Quantity of boxes: "

call readint

mov [productQuantity + ebx \* 4], eax

inc totalItems

dec ecx

jg addItemLoop

doneAdding:

call crlf

mwrite " Items added successfully!"

call crlf

call menu

pop ebp

ret

addItem endp

removeItem proc

push ebp

mov ebp, esp

call clrscr

call crlf

mwrite " Enter Product ID to remove: "

call readint

mov ebx, eax

mov ecx, 0

mov edx, totalItems

findItem:

cmp ecx, edx

jge notFound

mov eax, [productID + ecx \* 4]

cmp eax, ebx

je itemFound

inc ecx

jmp findItem

itemFound:

mov [productID + ecx \* 4], 0

mov [productQuantity + ecx \* 4], 0

dec totalItems

call crlf

mwrite " Item removed successfully!"

call crlf

call menu

pop ebp

ret

notFound:

call crlf

mwrite " Item ID not found!"

call crlf

call menu

pop ebp

ret

removeItem endp

updateItem proc

push ebp

mov ebp, esp

call clrscr

call crlf

mwrite " Enter Product ID to update quantity: "

call readint

mov ebx, eax

mov ecx, 0

mov edx, totalItems

findItemForUpdate:

cmp ecx, edx

jge notFoundForUpdate

mov eax, [productID + ecx \* 4]

cmp eax, ebx

je itemFoundForUpdate

inc ecx

jmp findItemForUpdate

itemFoundForUpdate:

call crlf

mwrite " Enter new quantity of boxes: "

call readint

mov [productQuantity + ecx \* 4], eax

call crlf

mwrite " Quantity updated successfully!"

call crlf

call menu

pop ebp

ret

notFoundForUpdate:

call crlf

mwrite " Item ID not found!"

call crlf

call menu

pop ebp

ret

updateItem endp

searchItem proc

push ebp

mov ebp, esp

call clrscr

call crlf

mwrite " Enter Product ID to search: "

call readint

mov ebx, eax

mov ecx, 0

mov edx, totalItems

findItemForSearch:

cmp ecx, edx

jge notFoundForSearch

mov eax, [productID + ecx \* 4]

cmp eax, ebx

je itemFoundForSearch

inc ecx

jmp findItemForSearch

itemFoundForSearch:

call crlf

mwrite " Item found. Quantity of boxes: "

mov eax, [productQuantity + ecx \* 4]

call writedec

call crlf

call menu

pop ebp

ret

notFoundForSearch:

call crlf

mwrite " Item ID not found!"

call crlf

call menu

pop ebp

ret

searchItem endp

displayAllData proc

push ebp

mov ebp, esp

call clrscr

call crlf

cmp totalItems, 0

je noData

mwrite " Displaying all data:"

call crlf

mov ecx, 0

displayLoop:

cmp ecx, totalItems

jge done

mwrite " Product ID: "

mov eax, [productID + ecx \* 4]

call writedec

call crlf

mwrite " Quantity of boxes: "

mov eax, [productQuantity + ecx \* 4]

call writedec

call crlf

inc ecx

jmp displayLoop

noData:

call crlf

mwrite " No items in inventory."

call crlf

done:

call menu

pop ebp

ret

displayAllData endp

clearAllData proc

push ebp

mov ebp, esp

call clrscr

mov totalItems, 0

mov productID, 0

mov productQuantity, 0

call crlf

mwrite " All data cleared!"

call crlf

call menu

pop ebp

ret

clearAllData endp

viewTotalItems proc

push ebp

mov ebp, esp

call clrscr

call crlf

mwrite " Total items in inventory: "

mov eax, totalItems

call writedec

call crlf

call menu

pop ebp

ret

viewTotalItems endp

displayTotalQuantity proc

push ebp

mov ebp, esp

call clrscr

call crlf

mwrite " Total quantity of all products: "

mov ecx, 0

mov edx, totalItems

xor eax, eax

displayTotalQuantityLoop:

cmp ecx, edx

jge doneTotalQuantity

mov ebx, [productQuantity + ecx \* 4]

add eax, ebx

inc ecx

jmp displayTotalQuantityLoop

doneTotalQuantity:

call writedec

call crlf

call menu

pop ebp

ret

displayTotalQuantity endp

sortProducts proc

push ebp

mov ebp, esp

call clrscr

call crlf

mwrite " Sorting products by Product ID..."

call crlf

cmp totalItems, 1

jle alreadySorted

mov ecx, totalItems

outerLoop:

mov esi, 0

mov edi, ecx

dec edi

cmp edi, 0

jle sorted

innerLoop:

mov eax, productID[esi \* 4]

mov ebx, productID[esi \* 4 + 4]

cmp eax, ebx

jle noSwap

mov productID[esi \* 4], ebx

mov productID[esi \* 4 + 4], eax

mov eax, productQuantity[esi \* 4]

mov ebx, productQuantity[esi \* 4 + 4]

mov productQuantity[esi \* 4], ebx

mov productQuantity[esi \* 4 + 4], eax

noSwap:

inc esi

dec edi

jnz innerLoop

dec ecx

jmp outerLoop

sorted:

call crlf

mwrite " Products sorted successfully!"

call crlf

alreadySorted:

call menu

pop ebp

ret

sortProducts endp

checkRestock proc

push ebp

mov ebp, esp

call clrscr

call crlf

mwrite " Checking items for restock..."

call crlf

mov ecx, 0

cmp totalItems, 0

je noItemsToCheck

checkLoop:

cmp ecx, totalItems

jge doneChecking

mov eax, productQuantity[ecx \* 4]

cmp eax, 10

jge skipItem

call crlf

mwrite " Product ID: "

mov eax, productID[ecx \* 4]

call writedec

call crlf

mwrite " Quantity of boxes: "

mov eax, productQuantity[ecx \* 4]

call writedec

call crlf

skipItem:

inc ecx

jmp checkLoop

noItemsToCheck:

call crlf

mwrite " No items in inventory to check for restock."

call crlf

jmp doneChecking

doneChecking:

call crlf

mwrite " Restock check complete!"

call crlf

call menu

pop ebp

ret

checkRestock endp

saveDataToFile proc

push ebp

mov ebp, esp

mov edx, OFFSET filename

call CreateOutputFile

mov fileHandle, eax

cmp fileHandle, INVALID\_HANDLE\_VALUE

je fileError

call crlf

mwrite " Debug: File created successfully."

call crlf

mov ecx, totalItems

mov esi, 0

writeLoop:

cmp esi, ecx

jge doneWriting

mov eax, fileHandle

mov edx, OFFSET buffer

mov ebx, OFFSET productID

add ebx, esi

call WriteToFile

mov eax,filehandle

mov edx, OFFSET buffer

mov ebx, OFFSET productQuantity

add ebx, esi

call WriteToFile

inc esi

jmp writeLoop

doneWriting:

call crlf

mwrite " Data saved to file successfully!"

call crlf

mov eax, fileHandle

call CloseFile

call crlf

mwrite " Debug: File closed successfully."

call crlf

jmp endSave

fileError:

call crlf

mwrite " Error creating file! Check file path and permissions."

call crlf

endSave:

pop ebp

ret

saveDataToFile endp

exitSystem proc

push ebp

mov ebp, esp

call crlf

mwrite " Exiting system..."

call crlf

exit

pop ebp

ret

exitSystem endp

end main