Task 1:

Create a Graphical User Interface that enables a user to enter the data in the following data dictionary, and clears the file.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field** | **Data Type** | **Description** | **Identifier** | **Domain(range or values)** |
| Customer F Name | string | This is the customer’s first name. | strCustFirstName |  |
| Customer L Name | string | This is the customer’s last name. | strCustLastName |  |
| Customer house number | int | This is the house number. | intHouseNumber | Only a whole number. |
| Customer street name | string | This is the customer’s street name. | strStreetName |  |
| Customer suburb | string | This is the customer’s suburb. | strSuburb |  |
| Customer postcode | int | This is the customer’s postcode. | intPostcode | Between 3000-3999 |
| Customer phone number | string | This is the customer’s phone number | strPhoneNumber |  |

Task 2

Create a module that will swap two objects, when two values are declared and the module swaps the contents of each text based object.

Pseudocode

Begin (procedure Swap –ItemA, ItemB)

Temp temp A

ItemA ItemB

ItemB temp

End

Task 3 Create a program for either of the following sort algorithms:

1. Bubble Sort

begin BubbleSort(list)

for all elements of list

if list[i] > list[i+1]

swap(list[i], list[i+1])

end if

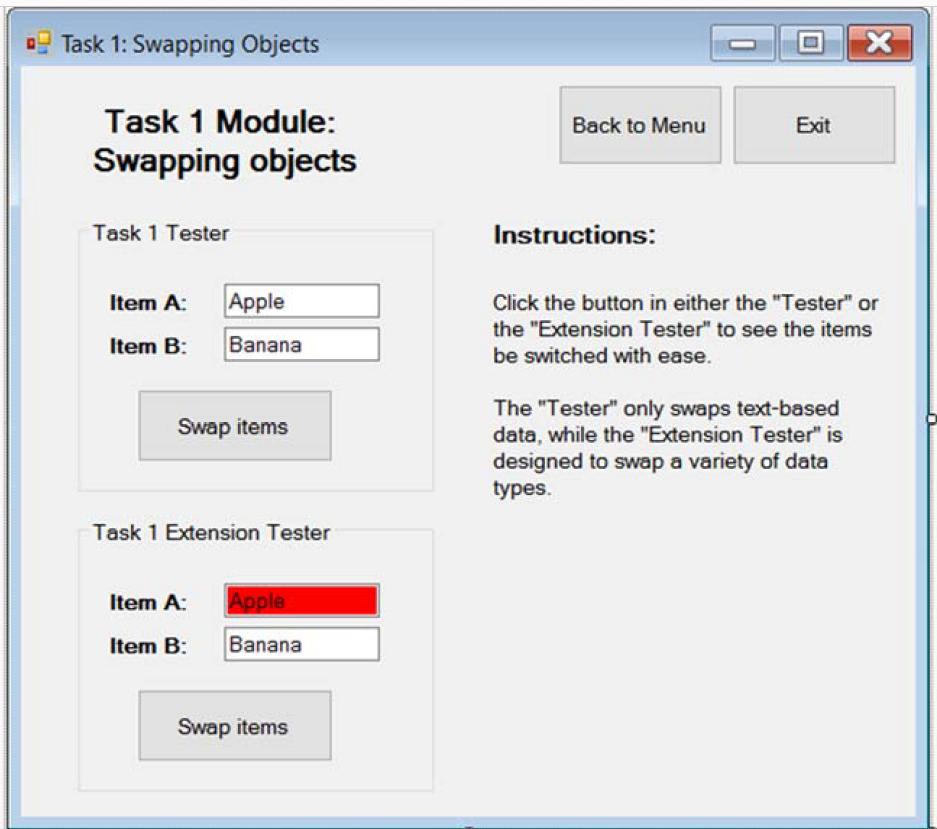
end for

return list

end BubbleSort

1. Quick Sort
2. procedure quickSort(left, right)
3. if right-left <= 0
4. return
5. else
6. pivot = A[right]
7. partition = partitionFunc(left, right, pivot)
8. quickSort(left,partition-1)
9. quickSort(partition+1,right)
10. end if
12. end procedure

Task 4: Create the code for the following design. You can use code from other tasks to help.



Task 5: