





PMR: This challenge program wasn’t really a challenge and I enjoyed creating it. The past three movie assignments helped create this assignment since it was basically the same concept. I enjoyed this program since it was easy to build since I could reuse code from previous programs.

/\*\*

\* Item Interface

\*

\* @author Anika Jallipalli

\* @version 4/11/2020

\*/

public class Item

{

private String itemID;

private String itemName;

private int inStore;

private double price;

public Item(String itemID, String itemName, int inStore, double price)

{

this.itemID = itemID;

this.itemName = itemName;

this.inStore = inStore;

this.price = price;

}

public String toString()

{

String string = itemID + " " + itemName + " " + inStore + " $ " + price;

return string;

}

public String getID()

{

return itemID;

}

public int getStore()

{

return inStore;

}

public String getName()

{

return itemName;

}

public double getPrice()

{

return price;

}

}

/\*\*

\* run the program

\*

\* @author Anika Jallipalli

\* @version 4/11/2020

\*/

import java.lang.\*;

public class TestItem

{

public static void main(String[] args)

{

Item airfilter = new Item("1011","Air Filters",200,10.5);

Item doorknobs = new Item("1034","Door Knobs",60,21.5);

Item hammers = new Item("1101","Hammers",90,9.99);

Item levels = new Item("1600","Levels",80,19.99);

Item ceilingfans = new Item("1500","Ceiling Fans",100,59);

Item wrenchsets = new Item("1201","Wrench Sets",55,80);

Item[] items = new Item[] {hammers, doorknobs, airfilter,

levels, ceilingfans, wrenchsets};

Item[] newitems = new Item[6];

System.out.println("Original Array:");

System.out.println();

System.out.println("itemID itemName inStore price");

System.out.println("-------------------------------------");

printitems(items);

System.out.println();

System.out.println("Sorted by ID:");

System.out.println();

System.out.println("itemID itemName inStore price");

System.out.println("-------------------------------------");

sortID(items,1);

printitems(items);

System.out.println();

System.out.println("Sorted by Name:");

System.out.println();

System.out.println("itemID itemName inStore price");

System.out.println("-------------------------------------");

newitems = sortNames(items,2);

printitems(newitems);

System.out.println();

System.out.println("Sorted by Store:");

System.out.println();

System.out.println("itemID itemName inStore price");

System.out.println("-------------------------------------");

sortStores(newitems,1);

printitems(newitems);

System.out.println();

System.out.println("Sorted by Price:");

System.out.println();

System.out.println("itemID itemName inStore price");

System.out.println("-------------------------------------");

sortPrices(newitems,0,5);

printitems(newitems);

}

public static void printitems(Item[] a)

{

for(int i = 0; i < a.length; i++)

{

System.out.println(a[i].toString());

}

}

public static void sortID(Item[] a, int b)

{

if(b==1)

{

for (int i = 0; i < a.length - 1; ++i)

{

int minIndex = i;

for (int j = i + 1; j < a.length; ++j)

{

if ((a[j].getID()).compareTo(a[minIndex].getID()) < 0)

{

minIndex = j;

}

}

Item temp = a[i];

a[i] = a[minIndex];

a[minIndex] = temp;

}

}

else if (b==2)

{

for (int i = 0; i < a.length - 1; ++i)

{

int minIndex = i;

for (int j = i + 1; j < a.length; ++j)

{

if ((a[j].getID()).compareTo(a[minIndex].getID()) > 0)

{

minIndex = j;

}

}

Item temp = a[i];

a[i] = a[minIndex];

a[minIndex] = temp;

}

}

else

{

System.out.println("ERROR");

}

}

public static Item[] sortNames(Item[] a, int b)

{

Item[] newlist = new Item[a.length];

if(b==1)

{

for(int i=0;i<a.length;i++)

{

String next = a[i].getName();

int insert = 0;

int k =i;

while(k>0 && insert == 0)

{

if(next.compareTo( newlist[k-1].getName() ) > 0)

{

insert = k;

}

else

{

newlist[k] = newlist[k-1];

}

k--;

}

newlist[insert]=a[i];

}

return newlist;

}

else if (b==2)

{

for(int i=0;i<a.length;i++)

{

String next = a[i].getName();

int insert = 0;

int k =i;

while(k>0 && insert == 0)

{

if(next.compareTo( newlist[k-1].getName() ) < 0)

{

insert = k;

}

else

{

newlist[k] = newlist[k-1];

}

k--;

}

newlist[insert]=a[i];

}

return newlist;

}

else

{

return newlist;

}

}

public static void sortStores(Item[] a, int b)

{

Item temp;

int i;

int k;

int posmax;

if(b==1)

{

for ( i = a.length - 1 ; i > 0 ; i-- )

{

posmax = 0;

for ( k = 1 ; k <= i ; k++ )

{

if ( a[ k ].getStore() > a[ posmax ].getStore() )

{

posmax = k;

}

}

temp = a[ i ];

a[ i ] = a[posmax ];

a[ posmax ] = temp;

}

}

else if (b==2)

{

for ( i = a.length - 1 ; i > 0 ; i-- )

{

posmax = 0;

for ( k = 1 ; k <= i ; k++ )

{

if ( a[ k ].getStore() < a[ posmax ].getStore() )

{

posmax = k;

}

}

temp = a[ i ];

a[ i ] = a[posmax ];

a[ posmax ] = temp;

}

}

else

{

System.out.println("ERROR");

}

}

public static void sortPrices(Item[] a, int low, int high)

{

if ( low == high )

{

return;

}

int mid = ( low + high ) / 2;

sortPrices( a, low, mid );

sortPrices( a, mid + 1, high);

mergePrices( a, low, mid, high);

}

public static void mergePrices( Item[] a, int low, int mid, int high )

{

Item[] temp = new Item[ high - low + 1 ];

int i = low, j = mid + 1, n = 0;

while ( i <= mid || j <= high )

{

if ( i > mid )

{

temp[ n ] = a[ j ];

j++;

}

else if ( j > high )

{

temp[ n ] = a[ i ];

i++;

}

else if ( a[ i ].getPrice() > a[ j ].getPrice() )

{

temp[ n ] = a[ i ];

i++;

}

else

{

temp[ n ] = a[ j ];

j++;

}

n++;

}

for ( int k = low ; k <= high ; k++ )

{

a[ k ] = temp[ k - low ];

}

}

}