**Name (s): Kaumil Patel**

**Course Name:** Principles of Software Design

**Lab Section:** B01

**Course Code:** ENSF 480

**Assignment Number:** Lab 5

**Submission Date and Time:** Nov 4, 2021

**Exercise A - Design Pattern (15 marks):**

// Program Output

The original values in v1 object are:

44.3

58.7

28.5

84.4

90.8

The values in MyVector object v1 after performing BoubleSorter is:

28.5

44.3

58.7

84.4

90.8

The original values in v2 object are:

17.0

5.0

14.0

2.0

13.0

The values in MyVector object v2 after performing InsertionSorter is:

2.0

5.0

13.0

14.0

17.0

/\*

\* File Name: MyVector.java

\* Assignment: Lab 5

\* Lab Section: B01

\* Completed by: Kaumil Patel

\* Submission Date: Nov 4, 2021

\*/

import java.util.ArrayList;

import java.text.DecimalFormat;

public class MyVector< E extends Number & Comparable<E> > {

private ArrayList<Item<E>> storageM;

private Sorter<E> sorter;

MyVector(int n){

storageM = new ArrayList<Item<E>>(n);

}

MyVector(ArrayList<E> arr){

storageM = new ArrayList<Item<E>>(arr.size());

for(int i=0;i<arr.size();i++) {

storageM.add(new Item<E>(arr.get(i)));

}

}

public void add(Item<E> value) {

storageM.add(value);

}

public void setSortStrategy(Sorter <E> s) {

sorter = s;

}

public void performSort() {

sorter.sort(storageM);

}

public void display() {

for(int i=0;i<storageM.size();i++) {

DecimalFormat df = new DecimalFormat("#.0");

System.out.print(df.format(storageM.get(i).getItem()));

System.out.println();

}

}

}

/\*

\* File Name: Sorter.java

\* Assignment: Lab 5

\* Lab Section: B01

\* Completed by: Kaumil Patel

\* Submission Date: Nov 4, 2021

\*/

import java.util.ArrayList;

public abstract class Sorter< E extends Number & Comparable<E> > {

abstract public void sort(ArrayList<Item<E>> arr);

}

/\*

\* File Name: InsertionSorter.java

\* Assignment: Lab 5

\* Lab Section: B01

\* Completed by: Kaumil Patel

\* Submission Date: Nov 4, 2021

\*/

import java.util.ArrayList;

public class InsertionSorter< E extends Number & Comparable<E> > extends Sorter<E> {

@Override

public void sort(ArrayList<Item<E>> arr) {

for(int i=1;i<arr.size();i++) {

Item<E> key = arr.get(i);

int j=i-1;

while(j>=0 && key.lessThan(arr.get(j))) {

arr.set(j+1, arr.get(j));

j--;

}

arr.set(j+1, key);

}

}

}

/\*

\* File Name: BubbleSorter.java

\* Assignment: Lab 5

\* Lab Section: B01

\* Completed by: Kaumil Patel

\* Submission Date: Nov 4, 2021

\*/

import java.util.ArrayList;

public class BubbleSorter< E extends Number & Comparable<E> > extends Sorter<E>{

@Override

public void sort(ArrayList<Item<E>> arr) {

for(int i=0;i<arr.size()-1;i++) {

for(int j=i+1;j<arr.size();j++) {

if(arr.get(j).lessThan(arr.get(i))) {

Item<E> temp = arr.get(i);

arr.set(i, arr.get(j));

arr.set(j, temp);

}

}

}

}

}

/\*

\* File Name: Item.java

\* Assignment: Lab 5

\* Lab Section: B01

\* Completed by: Kaumil Patel

\* Submission Date: Nov 4, 2021

\*/

class Item <E extends Number & Comparable<E> >{

private E item;

public Item(E value) {

item = value;

}

public void setItem(E value){

item = value;

}

public E getItem(){

return item;

}

public boolean lessThan(Item<E> rhs){

if(item.compareTo(rhs.item) < 0)

return true;

else

return false;

}

}

**Exercise B (4 marks):**

// Program Output

The original values in v1 object are:

25.1

38.4

86.3

66.7

98.8

The values in MyVector object v1 after performing BoubleSorter is:

25.1

38.4

66.7

86.3

98.8

The original values in v2 object are:

38.0

17.0

47.0

1.0

19.0

The values in MyVector object v2 after performing InsertionSorter is:

1.0

17.0

19.0

38.0

47.0

The original values in v3 object are:

24.0

41.0

16.0

1.0

44.0

The values in MyVector object v3 after performing SelectionSorter is:

1.0

16.0

24.0

41.0

44.0

/\*

\* File Name: MyVector.java

\* Assignment: Lab 5

\* Lab Section: B01

\* Completed by: Kaumil Patel

\* Submission Date: Nov 4, 2021

\*/

import java.util.ArrayList;

import java.text.DecimalFormat;

public class MyVector< E extends Number & Comparable<E> > {

private ArrayList<Item<E>> storageM;

private Sorter<E> sorter;

MyVector(int n){

storageM = new ArrayList<Item<E>>(n);

}

MyVector(ArrayList<E> arr){

storageM = new ArrayList<Item<E>>(arr.size());

for(int i=0;i<arr.size();i++) {

storageM.add(new Item<E>(arr.get(i)));

}

}

public void add(Item<E> value) {

storageM.add(value);

}

public void setSortStrategy(Sorter <E> s) {

sorter = s;

}

public void performSort() {

sorter.sort(storageM);

}

public void display() {

for(int i=0;i<storageM.size();i++) {

DecimalFormat df = new DecimalFormat("#.0");

System.out.print(df.format(storageM.get(i).getItem()));

System.out.println();

}

}

}

/\*

\* File Name: Sorter.java

\* Assignment: Lab 5

\* Lab Section: B01

\* Completed by: Kaumil Patel

\* Submission Date: Nov 4, 2021

\*/

import java.util.ArrayList;

public abstract class Sorter< E extends Number & Comparable<E> > {

abstract public void sort(ArrayList<Item<E>> arr);

}

/\*

\* File Name: InsertionSorter.java

\* Assignment: Lab 5

\* Lab Section: B01

\* Completed by: Kaumil Patel

\* Submission Date: Nov 4, 2021

\*/

import java.util.ArrayList;

public class InsertionSorter< E extends Number & Comparable<E> > extends Sorter<E> {

@Override

public void sort(ArrayList<Item<E>> arr) {

for(int i=1;i<arr.size();i++) {

Item<E> key = arr.get(i);

int j=i-1;

while(j>=0 && key.lessThan(arr.get(j))) {

arr.set(j+1, arr.get(j));

j--;

}

arr.set(j+1, key);

}

}

}

/\*

\* File Name: BubbleSorter.java

\* Assignment: Lab 5

\* Lab Section: B01

\* Completed by: Kaumil Patel

\* Submission Date: Nov 4, 2021

\*/

import java.util.ArrayList;

public class BubbleSorter< E extends Number & Comparable<E> > extends Sorter<E>{

@Override

public void sort(ArrayList<Item<E>> arr) {

for(int i=0;i<arr.size()-1;i++) {

for(int j=i+1;j<arr.size();j++) {

if(arr.get(j).lessThan(arr.get(i))) {

Item<E> temp = arr.get(i);

arr.set(i, arr.get(j));

arr.set(j, temp);

}

}

}

}

}

/\*

\* File Name: Item.java

\* Assignment: Lab 5

\* Lab Section: B01

\* Completed by: Kaumil Patel

\* Submission Date: Nov 4, 2021

\*/

class Item <E extends Number & Comparable<E> >{

private E item;

public Item(E value) {

item = value;

}

public void setItem(E value){

item = value;

}

public E getItem(){

return item;

}

public boolean lessThan(Item<E> rhs){

if(item.compareTo(rhs.item) < 0)

return true;

else

return false;

}

}

/\*

\* File Name: SelectionSorter.java

\* Assignment: Lab 5

\* Lab Section: B01

\* Completed by: Kaumil Patel

\* Submission Date: Nov 4, 2021

\*/

import java.util.ArrayList;

public class SelectionSorter< E extends Number & Comparable<E> > extends Sorter<E> {

@Override

public void sort(ArrayList<Item<E>> arr) {

for(int i=0;i<arr.size()-1;i++) {

int lowest = i;

for(int j=i+1;j<arr.size();j++) {

if(arr.get(j).lessThan(arr.get(i))) {

lowest = j;

}

}

Item<E> temp = arr.get(i);

arr.set(i, arr.get(lowest));

arr.set(lowest, temp);

}

}

}

**Exercise C (15 marks):**

// Program Output

Creating object mydata with an empty list -- no data:

Expected to print: Empty List ...

mydata object is populated with: 10, 20, 33, 44, 50, 30, 60, 70, 80, 10, 11, 23, 34, 55

Now, creating three observer objects: ht, vt, and hl

which are immediately notified of existing data with different views.

Notification to Three-Column Table Observer: Data Changed:

10.0 20.0 33.0

44.0 50.0 30.0

60.0 70.0 80.0

10.0 11.0 23.0

34.0 55.0

Notification to Five-Rows Table Observer: Data Changed:

10.0 30.0 11.0

20.0 60.0 23.0

33.0 70.0 34.0

44.0 80.0 55.0

50.0 10.0

Notification to One-Row Observer: Data Changed:

10.0 20.0 33.0 44.0 50.0 30.0 60.0 70.0 80.0 10.0 11.0 23.0 34.0 55.0

Changing the third value from 33, to 66 -- (All views must show this change):

Notification to Three-Column Table Observer: Data Changed:

10.0 20.0 66.0

44.0 50.0 30.0

60.0 70.0 80.0

10.0 11.0 23.0

34.0 55.0

Notification to Five-Rows Table Observer: Data Changed:

10.0 30.0 11.0

20.0 60.0 23.0

66.0 70.0 34.0

44.0 80.0 55.0

50.0 10.0

Notification to One-Row Observer: Data Changed:

10.0 20.0 66.0 44.0 50.0 30.0 60.0 70.0 80.0 10.0 11.0 23.0 34.0 55.0

Adding a new value to the end of the list -- (All views must show this change)

Notification to Three-Column Table Observer: Data Changed:

10.0 20.0 66.0

44.0 50.0 30.0

60.0 70.0 80.0

10.0 11.0 23.0

34.0 55.0 1000.0

Notification to Five-Rows Table Observer: Data Changed:

10.0 30.0 11.0

20.0 60.0 23.0

66.0 70.0 34.0

44.0 80.0 55.0

50.0 10.0 1000.0

30.0 11.0

Notification to One-Row Observer: Data Changed:

10.0 20.0 66.0 44.0 50.0 30.0 60.0 70.0 80.0 10.0 11.0 23.0 34.0 55.0 1000.0

Now removing two observers from the list:

Only the remained observer (One Row ), is notified.

Notification to One-Row Observer: Data Changed:

10.0 20.0 66.0 44.0 50.0 30.0 60.0 70.0 80.0 10.0 11.0 23.0 34.0 55.0 1000.0 2000.0

Now removing the last observer from the list:

Adding a new value the end of the list:

Since there is no observer -- nothing is displayed ...

Now, creating a new Three-Column observer that will be notified of existing data:

Notification to Three-Column Table Observer: Data Changed:

10.0 20.0 66.0

44.0 50.0 30.0

60.0 70.0 80.0

10.0 11.0 23.0

34.0 55.0 1000.0

2000.0 3000.0

/\*

\* File Name: Subject.java

\* Assignment: Lab 5

\* Lab Section: B01

\* Completed by: Kaumil Patel

\* Submission Date: Nov 4, 2021

\*/

**public** **interface** Subject {

**public** **void** registerObserver(Observer observer);

**public** **void** removeObserver(Observer observer);

**public** **void** notifyAllObservers();

}

/\*

\* File Name: DoubleArrayListSubject.java

\* Assignment: Lab 5

\* Lab Section: B01

\* Completed by: Kaumil Patel

\* Submission Date: Nov 4, 2021

\*/

**import** java.util.ArrayList;

**public** **class** DoubleArrayListSubject **implements** Subject{

**private** ArrayList<Double> data;

**private** ArrayList<Observer> observers;

**public** DoubleArrayListSubject(){

data = **new** ArrayList<Double>();

observers = **new** ArrayList<Observer>();

}

**public** **void** addData(Double value) {

data.add(value);

notifyAllObservers();

}

**public** **void** setData(Double value, **int** index) {

data.set(index, value);

notifyAllObservers();

}

**public** ArrayList<Double> getData() {

**return** data;

}

**public** **void** populate(**double**[] arr) {

data = **new** ArrayList<Double>(arr.length);

**for**(**int** i=0;i<arr.length;i++) {

data.add(arr[i]);

}

notifyAllObservers();

}

**public** **void** display() {

**for**(**int** i=0;i<data.size();i++) {

System.***out***.print(data.get(i)+" ");

}

System.***out***.println();

}

**public** **void** remove(Observer observer) {

observers.remove(observer);

}

@Override

**public** **void** registerObserver(Observer observer) {

observers.add(observer);

}

@Override

**public** **void** removeObserver(Observer observer) {

observers.remove(observer);

}

@Override

**public** **void** notifyAllObservers() {

**for**(**int** i=0;i<observers.size();i++) {

observers.get(i).update(data);

}

}

}

/\*

\* File Name: Observer.java

\* Assignment: Lab 5

\* Lab Section: B01

\* Completed by: Kaumil Patel

\* Submission Date: Nov 4, 2021

\*/

**import** java.util.ArrayList;

**public** **interface** Observer {

**void** update(ArrayList<Double> arr);

}

/\*

\* File Name: FiveRowsTable\_Observer.java

\* Assignment: Lab 5

\* Lab Section: B01

\* Completed by: Kaumil Patel

\* Submission Date: Nov 4, 2021

\*/

**import** java.util.ArrayList;

**public** **class** FiveRowsTable\_Observer **implements** Observer{

**public** FiveRowsTable\_Observer(DoubleArrayListSubject mydata) {

mydata.registerObserver(**this**);

update(mydata.getData());

}

@Override

**public** **void** update(ArrayList<Double> arr) {

**int** i = 0;

System.***out***.println("\nNotification to Five-Rows Table Observer: Data Changed:");

**while**(i<arr.size()) {

**for**(**int** n=0;n<3;n++) {

**if**(i+n\*5>=arr.size()) {

System.***out***.println();

**return**;

}

System.***out***.print(arr.get(i+n\*5) + " ");

}

i++;

System.***out***.println();

}

}

}

/\*

\* File Name: ThreeColumnTable\_Observer.java

\* Assignment: Lab 5

\* Lab Section: B01

\* Completed by: Kaumil Patel

\* Submission Date: Nov 4, 2021

\*/

**import** java.util.ArrayList;

**public** **class** ThreeColumnTable\_Observer **implements** Observer{

**public** ThreeColumnTable\_Observer(DoubleArrayListSubject mydata) {

mydata.registerObserver(**this**);

update(mydata.getData());

}

@Override

**public** **void** update(ArrayList<Double> arr) {

**int** i = 0;

System.***out***.println("\nNotification to Three-Column Table Observer: Data Changed:");

**while**(i<arr.size()) {

**for**(**int** n=0;n<3 && i<arr.size();n++) {

System.***out***.print(arr.get(i) + " ");

i++;

}

System.***out***.println();

}

}

}

/\*

\* File Name: OneRow\_Observer.java

\* Assignment: Lab 5

\* Lab Section: B01

\* Completed by: Kaumil Patel

\* Submission Date: Nov 4, 2021

\*/

**import** java.util.ArrayList;

**public** **class** OneRow\_Observer **implements** Observer {

**public** OneRow\_Observer(DoubleArrayListSubject mydata) {

mydata.registerObserver(**this**);

update(mydata.getData());

}

@Override

**public** **void** update(ArrayList<Double> arr) {

**int** i = 0;

System.***out***.println("\nNotification to One-Row Observer: Data Changed:");

**while** (i < arr.size()) {

System.***out***.print(arr.get(i) + " ");

i++;

}

}

}