# **Національний технічний університет України**

# **«Київський політехнічний інститут»**

# **Факультет інформатики та обчислювальної техніки**

# **Кафедра обчислювальної техніки**

# **Лабораторна робота №6**

### **З програмування**

**виконав студент першого курсу**

**група ІО-01**

**Шийка Владислав Володимирович**

**Тема:** Робота з абстрактними класами та інтерфейсами в мові програмування Java.

**Мета:** Здобуття навичок у створенні та використанні абстрактних класів та інтерфейсів в мові програмування Java.

1. Визначення варіанту та завдання

Номер залікової книжки – 129

С12=129 mod 2 = 1

Отже завдання має такий вигляд:

Абстрактний клас

1. Використовуючи варіант попередньої роботи створити батьківський клас (п.2) не менше ніж для 3 класів, які описують різновиди батьківського, та складаються не менше ніж з 7 полів (принаймні 2 поля повинні бути агрегованими об’єктами) та методів доступу до цих полів. В класі необхідно описати, використовуючи стандартні засоби мови програмування, не менше 2 методів, за допомогою яких можна було б сортувати масив об’єктів батьківського класу за одним із полів. Для перевірки необхідно створити клас, що складається з виконавчого методу. Всі початкові дані задаються у виконавчому методі.
2. Лістинг додактового класу для створення агрегованого об’єкту Date :

**public** **class** Date { //Additional class to use date fields

**private** **int** day;

**private** **int** month;

**private** **int** year;

Date(**int** day,**int** month,**int** year){

//full constructor

setDay(day);

setMonth(month);

setYear(year);

}

Date(){

//empty constructor

**this**(0,0,0);

}

//Access methods:

**public** **void** setDate(**int** day,**int** month,**int** year){

setDay(day);

setMonth(month);

setYear(year);

}

**public** **void** setDay(**int** day) {

**this**.day = day;

}

**public** **void** setMonth(**int** month) {

**this**.month = month;

}

**public** **void** setYear(**int** year) {

**this**.year = year;

}

//Getting methods:

**public** String getDate(){

**return** day+"/"+month+"/"+year;

}

**public** **int** getDay() {

**return** day;

}

**public** **int** getMonth() {

**return** month;

}

**public** **int** getYear() {

**return** year;

}

}

1. Лістинг батьківського класу:

**abstract** **class** AbsPlane { // Parent class

//Common fields:

**protected** **int** speed;

**protected** **int** distance;

**protected** **int** prodyear;

**protected** **int** places;

**protected** **int** height;

**protected** Name name = **new** Name();

**protected** Status status = **new** Status();

AbsPlane(**int** s,**int** d,**int** py,**int** pl,**int** h,String n,String st){

//full constructor

setSpeed(s);

setDistance(d);

setProdyear(py);

setPlaces(pl);

setHeight(h);

name.setName(n);

status.setStatus(st);

}

AbsPlane(){//empty constructor

**this**(0,0,0,0,0,"-/-","-/-");

}

AbsPlane(**int** s,**int** d,**int** y,**int** p){// with setting 4 field

**this**(s,d,y,p,0,"-/-","-/-");

}

//Access methods:

**public** **void** setSpeed(**int** speed) {

**this**.speed = speed;

}

**public** **void** setDistance(**int** distance) {

**this**.distance = distance;

}

**public** **void** setProdyear(**int** prodyear) {

**this**.prodyear = prodyear;

}

**public** **void** setPlaces(**int** places) {

**this**.places = places;

}

**public** **void** setHeight(**int** height) {

**this**.height = height;

}

**public** **void** setName(String namestr){

name.setName(namestr);

}

**public** **void** setStatus(String statstr){

status.setStatus(statstr);

}

//Getting methods:

**public** **int** getSpeed() {

**return** speed;

}

**public** **int** getDistance() {

**return** distance;

}

**public** **int** getProdyear() {

**return** prodyear;

}

**public** **int** getPlaces() {

**return** places;

}

**public** **int** getHeight() {

**return** height;

}

**public** String getName(){

**return** name.getName();

}

**public** String getStatus(){

**return** status.getStatus();

}

}

1. Лістинг першого дочірнього класу:

**public** **class** Techinfo **extends** AbsPlane {//Child class

//Special fields:

**private** Model model = **new** Model();

**private** Date dateofproduse = **new** Date();

Techinfo(){//empty constructor

**this**(0,0,0,0,0,"NO\_NAME","NO\_STATUS","NO\_MODEL",0,0,0);

}

Techinfo(**int** s ,**int** d,**int** py,**int** pl,**int** h,String n,String st,

String m,**int** day,**int** month,**int** year){

//full constructor

**super**(s ,d, py, pl, h, n, st);

setModel(m);

setDate(day, month, year);

}

//Access methods:

**public** **void** setModel(String modelstr){

model.setModel(modelstr);

}

**public** **void** setDate(**int** day,**int** month,**int** year){

dateofproduse.setDate(day,month,year);

}

//Getting methods:

**public** String getModel(){

**return** model.getModel();

}

**public** String getDate(){

**return** dateofproduse.getDate();

}

}

1. Лістинг другого дочірнього класу:

**public** **class** Airline **extends** AbsPlane {//Child class

//Special fields:

**private** **int** ticketprice;

**private** Date dateofflight = **new** Date();

Airline(**int** s ,**int** d,**int** py,**int** pl,**int** h,String n,String st,

**int** tp,**int** day,**int** month,**int** year){

//full constructor

**super**(s,d,py,pl,h,n,st);

setTicketPrice(tp);

setDateOfFlight(day,month,year);

}

Airline(){//empty constructor

**this**(0,0,0,0,0,"-/-","-/-",0,0,0,0);

}

//Access methods:

**public** **void** setTicketPrice(**int** ticketprice){

**this**.ticketprice=ticketprice;

}

**public** **void** setDateOfFlight(**int** day,**int** month,**int** year){

dateofflight.setDate(day, month, year);

}

//Getting methods:

**public** String getTicketPrice(){

**return** ticketprice+"$";

}

**public** String getDateOfFlight(){

**return** dateofflight.getDay()+"/"+dateofflight.getMonth()+"/"+dateofflight.getYear();

}

}

1. Лістинг третьго дочірнього класу:

**public** **class** Additional **extends** AbsPlane {

//Special fields:

**private** String country = **new** String();

**private** **int** planeprice;

Additional(**int** s,**int** d,**int** py,**int** pl,**int** h,

String n,String st,String c,**int** pp){

//full constructor

**super**(s,d,py,pl,h,n,st);

setPlanePrice(pp);

setCountryBuilder(c);

}

Additional(){//empty constructor

**this**(0,0,0,0,0,"NO\_NAME","NO\_STATUS","NO\_COUNTRY",0);

}

//Access methods:

**public** **void** setPlanePrice(**int** planeprice) {

**this**.planeprice = planeprice;

}

**public** **void** setCountryBuilder(String country){

**this**.country=country;

}

//Getting methods:

**public** String getPlanePrice() {

**return** planeprice+"$";

}

**public** String getCountryBuilder(){

**return** country;

}

}

1. Лістинг компараторів:

//Compare objects by field "Name"

**public** **class** CompareName **implements** Comparator <AbsPlane> {

**public** **int** compare(AbsPlane ob1,AbsPlane ob2){

**return** ob1.getName().compareTo(ob2.getName());

}

}

//Compare objects by field "Speed"

**public** **class** CompareSpeed **implements** Comparator <AbsPlane> {

**public** **int** compare(AbsPlane ob1, AbsPlane ob2) {

**return** ob1.getSpeed()>ob2.getSpeed()?1:-1;

}

}

1. Лістинг виконавчого класу:

**public** **class** ExecLab4 {

**public** **static** **void** main(String[] args) {

AbsPlane[] array= **new** AbsPlane[3];//Creating array of parent class

//Declare examples of children classes with all information

Techinfo tech = **new** Techinfo(650,18000,2008,660,8000,"T-777","NICE","Enhanced",24,12,2008);

Airline air = **new** Airline(800,15000,2001,750,8000,"T-777-M","GOOD",400,21,4,2011);

Additional add = **new** Additional(900,12000,2010,800,7000,

"ABC-47","PERFECT","Ukraine",4750000);

//Setting reference to array of parent class

array[0]=tech;

array[1]=air;

array[2]=add;

//Displaying info about planes:

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

System.*out*.println("Techinfo Plane:");

System.*out*.println("Speed: "+tech.getSpeed());

System.*out*.println("Distance: "+tech.getDistance());

System.*out*.println("Year: "+tech.getProdyear());

System.*out*.println("Places :"+tech.getPlaces());

System.*out*.println("Heigth :"+tech.getHeight());

System.*out*.println("Name :"+tech.getName());

System.*out*.println("Status :"+tech.getStatus());

System.*out*.println("Model :"+tech.getModel());

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

System.*out*.println("Airline Plane:");

System.*out*.println("Speed: "+air.getSpeed());

System.*out*.println("Distance: "+air.getDistance());

System.*out*.println("Year: "+air.getProdyear());

System.*out*.println("Places :"+air.getPlaces());

System.*out*.println("Heigth :"+air.getHeight());

System.*out*.println("Name :"+air.getName());

System.*out*.println("Status :"+air.getStatus());

System.*out*.println("Ticket Price :"+air.getTicketPrice());

System.*out*.println("Date Of Flight :"+air.getDateOfFlight());

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

System.*out*.println("Additional Plane:");

System.*out*.println("Speed: "+add.getSpeed());

System.*out*.println("Distance: "+add.getDistance());

System.*out*.println("Year: "+add.getProdyear());

System.*out*.println("Places :"+add.getPlaces());

System.*out*.println("Heigth :"+add.getHeight());

System.*out*.println("Name :"+add.getName());

System.*out*.println("Status :"+add.getStatus());

System.*out*.println("Country Builder :"+add.getCountryBuilder());

System.*out*.println("Plane Price :"+add.getPlanePrice());

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

//Sorting object by field "Speed"

Arrays.*sort*(array,**new** CompareSpeed());

System.*out*.println("Sorting by speed: ");

**for**(**int** i=0;i<3;i++){//Displaying sorted array

System.*out*.println(array[i].getSpeed());

}

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

//Sorting object by field "Speed"

Arrays.*sort*(array,**new** CompareName());

System.*out*.println("Sorting by name: ");

**for**(**int** i=0;i<3;i++){//Displaying sorted array

System.*out*.println(array[i].getName());

}

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

}

}

**Висновок:** під час виконання цієї лабораторної роботи я здобув навички у створенні та використанні абстрактних класів та інтерфейсів. Здобув навички у створенні та використанні компараторів,у використанні наслідування. Покращив навички у створенні методів доступу,та директив доступу (видимості). Навчився сортувати масив об’єктів за допомогою компараторів, у заданому порядку,за одним з полів.