TP3

EX1)

La classe Barque :

package TP.TP3;

public class Barque {

    private String nom;

    private String num;

    public Barque(String nom, String num) {

        this.nom = nom;

        this.num = num;

    }

//cette methode heriter de la classe mère Object

    @Override

    public String toString() {

        return "Nom : "+nom+"\nNumero d'autorisation : "+num;

    }

    public String getNum() {

        return num;

    }

}

La classe BarqueMoteur :

package TP.TP3;

//BarqueMoteur est une sous classe de Barque

public class BarqueMoteur extends Barque{

    private int puissance;

    public BarqueMoteur(String nom, String num, int puissance) {

        super(nom, num);//constructeur de la classe mere

        this.puissance = puissance;

    }

    public int getPuissance() {

        return puissance;

    }

//heritage depuis la classe mère Barque

    @Override

    public String toString() {

        return super.toString()+"\nPuissance : "+puissance; //le mot clé super

pour l’accés a les méthodes de la classe mere

    }

}

La classe Port :

package TP.TP3;

import java.util.ArrayList;

public class Port {

    private String nom;

    private int capacite;

    private float taxe;

    private int nbBarque;

    private ArrayList<Barque> listBarque;//ArrayList est une collection

    public Port(String nom, int capacite, float taxe) {

        this.nom = nom;

        this.capacite = capacite;

        this.taxe = taxe;

        this.nbBarque = 0;

        this.listBarque = new ArrayList<>();

    }

//methode ajouter un barque ayant comme parametre barque

    public void ajouterBarque(Barque b) {

        if (nbBarque < capacite) {

            listBarque.add(b);//methode add predefinie dans la classe ArrayList

pour ajouter des elements

            nbBarque++;

        } else {

            System.out.println("Port est plein");

        }

    }

//methode ajouter barque d’une autre façon ayant comme parametre les infos nécessaire pour créer un barque simple

    public void ajouterBarque(String num, String nom) {

        if (nbBarque < capacite) {

            Barque b = new Barque(nom, num);

            listBarque.add(b);

            nbBarque++;

        } else {

            System.out.println("Port est plein");

        }

    }

   //même methode du dernier la difference est cette methode pour créer un barque moteur

public void ajouterBarque(String num, String nom, int p) {

        if (nbBarque < capacite) {

            Barque b = new BarqueMoteur(nom, num, p);

            listBarque.add(b);

            nbBarque++;

        } else {

            System.out.println("Port est plein");

        }

    }

//supprimer un barque a travers l’instance

    public void suppBarque(Barque b) {

        if (listBarque.remove(b)) {//remove methode predefinie pour supprimer

            System.out.println("Barque est quitter");

            nbBarque--;

        } else {

            System.out.println("Barque n'est pas disponible");

        }

    }

//supprimer un barque a travers numero

    public void suppBarque(String num) {

        if (nbBarque>0) {

            for (Barque barque : listBarque) {

                if(barque.getNum().equals(num)){

                    listBarque.remove(barque);

                    nbBarque--;

                    break;

                }

            }

        } else {

            System.out.println("Barque n'est pas disponible");

        }

    }

    public float calculerTaxe(Barque b) {

        if (b instanceof BarqueMoteur) {

            if (((BarqueMoteur) b).getPuissance() < 5)

                return taxe \* 1.1f;

            else

                return taxe \* 1.2f;

        } else {

            return taxe;

        }

    }

    public float recetteTaxeAujour() {

        float som = 0;

        for (Barque barque : listBarque) {

            som += calculerTaxe(barque);

        }

        return som;

    }

    @Override

    public String toString() {

        String ch="Nom : " + nom + "\nCapacite : " + capacite + "\nTotal taxe : " + recetteTaxeAujour()+"\nNombre de barque : "+nbBarque+"\nLes barques : \n";

        for (Barque barque : listBarque) {

            ch+=barque+"\n";

        }

        return ch;

    }

}

La classe TestPort :

package TP.TP3;

public class TestPort {

    public static void main(String[] args) {

        Port p = new Port("Port sfax", 40, 100);

        p.ajouterBarque(new Barque("barque 1", "1"));//taxe 100

        p.ajouterBarque(new BarqueMoteur("barque 2", "2", 5));//taxe 120

        p.ajouterBarque("4", "barque 4", 4); //taxe 110

        p.ajouterBarque("3", "barque 3");

        p.suppBarque("3");

        System.out.println(p);

    }

}

Graphical user interface, text, application

Description automatically generated