**Universidad Autónoma de Baja California**

**Grupo: 541**

**Docente:**

**Mayra Janeth Duran Rodriguez**

****

**Alumno:**

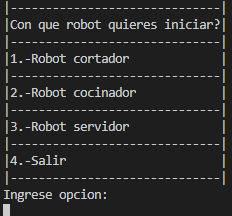
**Landa Luna Edgar Miguel 1263337**

**Programación orientada a objetos**

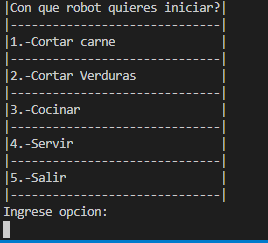
**Practica #7**

**Fecha de entrega: 24-noviembre-2020**

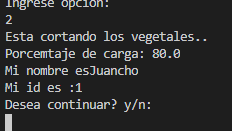
1. Menú



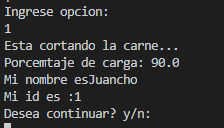
1. Menú 2



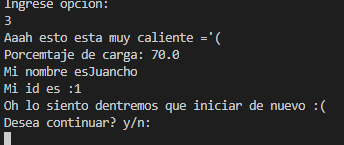
1. Opción 1
2. Opción cortar verduras



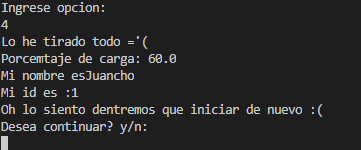
1. Opción cortar carnes



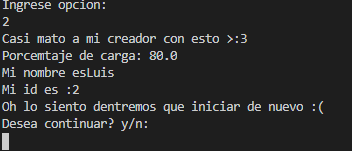
1. Opción hervir



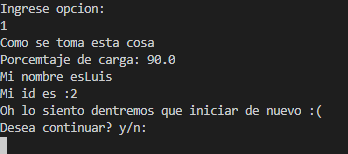
1. Opción servir



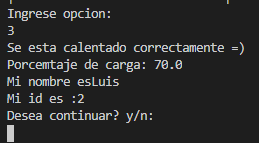
1. Opción 2
2. Opción cortar verduras



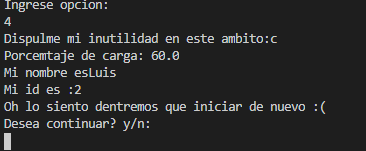
1. Opción cortar carnes



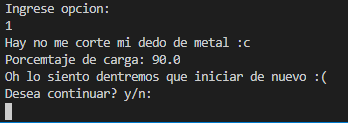
1. Opción hervir



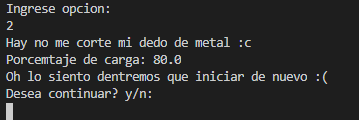
1. Opción servir



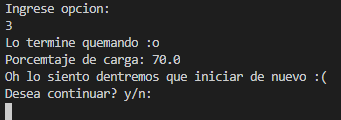
1. Opción 3
   1. Opción cortar verduras



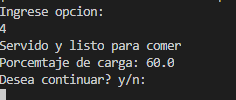
1. Opción cortar carnes



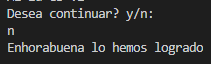
1. Opción hervir



1. Opción servir



1. Completado



1. Practica7.java

class Practica7{

    public static void main(String[] args) {

        selectionR();// inciar la seleccion

    }

    public static void selectionR(){

        Robot[] robot = new Robot[3];//inicializar

        robot[0] = new RobotCortado("Juancho",100,1,0,1,20,true);//robot cortador

        robot[1] = new RobotHervir("Luis",100,2,0);/// robot cocinero

        robot[2] = new RobotServir("Paco",100,3,0);/// robot servir

        char resp = 'n';/// solo para continuar

        int cont = 0;// contador que permite saber si se completo o no

        do{

            Menu.MenuPrincipal();// menu principal

            int op = CapturaEntrada.capturaEntero("Ingrese opcion");//op menu

            switch(op){

                case 1:// robot cortador

                    if(selectionAction(robot[0],cont) > 0){// con esto comprobamos si se hizo correcto

                        cont++;// se le suma al contador

                    }else{

                        cont = 0;// se reinicio el contador

                    }

                    robot[0].battery(10);//descarga la bateria

                    robot[0].setUtility(-1);/// le quita utilidad al cuchillo

                    if(robot[0].getUtility() == 0){//// saber si sigue sirviendo o no

                        System.out.println("Oh no se ha roto tomare otro");//mensaje

                        robot[0].setUtility(20);//rcupera la utibilidad

                    }

                    System.out.println("Porcemtaje de carga: "+robot[0].getBattery());// mostrar porcentaje

                    System.out.println("Mi nombre es"+ robot[0].name()+ "\n"+"Mi id es :"+ robot[0].id());//datos

                    if(robot[0].getBattery() == 0){///saber si se desargdo

                        System.out.println("Se termino la bateria favor de regargar");

                        if(CapturaEntrada.capturaChar("Desea recargar ? y/n") == 'y')//recargar?

                            robot[0].chargeBattery();//recarga

                        break;// vuelve a reiniciar menu

                    }

                    break;

                case 2:// aqui se repite todo lo del anterior pero sin utilidad

                    if(selectionAction(robot[1],cont) > 0){

                        cont++;

                    }else{

                        cont = 0;

                    }

                    robot[1].battery(10);

                    System.out.println("Porcemtaje de carga: "+robot[1].getBattery());

                    System.out.println("Mi nombre es"+ robot[1].name()+ "\n"+"Mi id es :"+ robot[1].id());

                    if(robot[1].getBattery() == 0){

                        System.out.println("Se termino la bateria favor de regargar");

                        if(CapturaEntrada.capturaChar("Desea recargar ? y/n") == 'y'){

                            robot[1].chargeBattery();

                        }

                        break;

                    }

                    break;

                case 3:

                    if(selectionAction(robot[2],cont) > 0){

                        cont++;

                    }else{

                        cont = 0;

                    }

                    robot[2].battery(10);

                    System.out.println("Porcemtaje de carga: "+robot[2].getBattery());

                    if(robot[2].getBattery() == 0){

                        System.out.println("Se termino la bateria favor de recargar");

                        System.out.println("Mi nombre es"+ robot[2].name()+ "\n"+"Mi id es :"+ robot[2].id());

                        if(CapturaEntrada.capturaChar("Desea recargar ? y/n") == 'y'){

                            robot[2].chargeBattery();

                        }

                        break;

                    }

                    break;

                case 4:

                    System.exit(0);

                    break;

                default:

                    break;

            }

            if(cont == 0)/// si se reinica el contador se empeiza de nuevo

                System.out.println("Oh lo siento dentremos que iniciar de nuevo :(");

            resp = CapturaEntrada.capturaChar("Desea continuar? y/n");

        }while(resp != 'n');

        if(cont > 4){

            System.out.println("Enhorabuena lo hemos logrado");//se logro

        }

    }

    public static int selectionAction(Robot robot,int cont){// sleciona que quieres que haga

        Menu.MenuAction();

        int op = CapturaEntrada.capturaEntero("Ingrese opcion");

        switch(op){

            case 1:

                if(robot.cutMeat()){///si es verdadero aumenta contador

                    cont++;

                }else{

                    cont=0;// reinicia contador

                }

                break;

            case 2:

                if(robot.cutVegetable()){

                    cont++;

                }else{

                    cont = 0;

                }

                break;

            case 3:

                if(robot.hotFood()){

                    cont++;

                }else{

                    cont = 0;

                }

                break;

            case 4:

                if(robot.serve()){

                    cont++;

                }else{

                    cont = 0;

                }

                break;

            default:

                break;

        }

        robot.actionLast(op);/// con esto le da como ultima accion

        return cont;

    }

}

1. Menú

public class Menu {// menus "Graficos"

    public static void MenuPrincipal(){

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

        System.out.println("|Con que robot quieres iniciar?|");

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

        System.out.println("|1.-Robot cortador             |");

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

        System.out.println("|2.-Robot cocinador            |");

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

        System.out.println("|3.-Robot servidor             |");

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

        System.out.println("|4.-Salir                      |");

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

    }

    public static void MenuAction(){

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

        System.out.println("|Con que robot quieres iniciar?|");

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

        System.out.println("|1.-Cortar carne               |");

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

        System.out.println("|2.-Cortar Verduras            |");

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

        System.out.println("|3.-Cocinar                    |");

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

        System.out.println("|4.-Servir                     |");

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

        System.out.println("|5.-Salir                      |");

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

    }

}

1. Captura

import java.util.\*;

public class CapturaEntrada{

  public static float capturaFloat(String msg){

    Scanner sc= new Scanner(System.in);

    System.out.print(""+ msg + ": ");

    return(sc.nextFloat());

    }

  public static String capturaString(String msg){

    Scanner sc= new Scanner(System.in);

    System.out.print(""+ msg + ": ");

    return(sc.nextLine());

    }

    public static int capturaEntero(String msg){

        Scanner sc =new Scanner(System.in);

        System.out.println(""+ msg + ": ");

        return(sc.nextInt());

  }

  public static char capturaChar(String msg){

        Scanner sc =new Scanner(System.in);

        System.out.println(""+ msg + ": ");

        return(sc.next().charAt(0));

  }

  public static boolean capturaBoolean(String msg){

    Scanner sc =new Scanner(System.in);

    System.out.println(""+ msg + ": ");

    return(sc.nextBoolean());

  }

  public static void Limpiar() {

    try {

        new ProcessBuilder("cmd", "/c", "cls").inheritIO().start().waitFor();

    } catch (Exception e) {

        /\*No hacer nada\*/

    }

  }

}

1. Robot

public abstract class Robot implements Interfaz{/// clase padre

    String name;

    float battery\_porcent;

    int id\_num;

    int action;

    public Robot(String name, float battery\_porcent, int id\_num, int action ){

        this.name=name;

        this.battery\_porcent=battery\_porcent;

        this.id\_num=id\_num;

        this.action=action;

    }

    protected abstract boolean cutMeat();

    protected abstract boolean cutVegetable();

    protected abstract boolean hotFood();

    protected abstract boolean serve();

    protected abstract float getBattery();

    protected abstract void chargeBattery();

    protected abstract void setUtility(int num);

    protected abstract int getUtility();

    public String name(){

        return name;

    }

    public void battery(){}

    public int id(){

        return 0;

    }

    public void actionLast(){}

}

1. RobotCortador

public class RobotCortado extends Robot{

    int speedCut;

    public int utility;

    boolean edge =true;

    public RobotCortado(String name, float battery\_porcent, int id\_num, int action, int speedCut, int utility, boolean edge){

        super(name,battery\_porcent,id\_num,action);

        this.speedCut = speedCut;

        this.utility = utility;

        this.edge = edge;

    }

    public void setUtility(int resta) {

        utility +=resta;

    }

    @Override

    protected boolean cutMeat() {

        System.out.println("Esta cortando la carne...");

        return true;

    }

    @Override

    protected boolean cutVegetable() {

        System.out.println("Esta cortando los vegetales..");

        return true;

    }

    @Override

    protected boolean hotFood(){

        System.out.println("Aaah esto esta muy caliente ='( ");

        return false;

    }

    @Override

    protected boolean serve(){

        System.out.println("Lo he tirado todo ='(");

        return false;

    }

    @Override

    public void actionLast(int num){

        action = num;

    }

    @Override

    public void battery(int discharge) {

        battery\_porcent-= discharge;

    }

    @Override

    public String name() {

        return name;

    }

    @Override

    public int id() {

        return id\_num;

    }

    public float getBattery(){

        return battery\_porcent;

    }

    public int getUtility(){

        return utility;

    }

    @Override

    protected void chargeBattery() {

        for (int i = 0; i < 10; i++)

            System.out.println("Canrgado ..... "+(battery\_porcent+=10)+"%");

        System.out.println("CARGA COMPLETA");

    }

}

1. RobotHervir

public class RobotHervir extends Robot {

    public RobotHervir(String name, float battery\_porcent, int id\_num, int action){

        super(name,battery\_porcent,id\_num,action);

    }

    @Override

    protected boolean cutMeat() {

        System.out.println("Como se toma esta cosa");

        return false;

    }

    @Override

    protected boolean cutVegetable() {

        System.out.println("Casi mato a mi creador con esto >:3");

        return false;

    }

    @Override

    protected boolean hotFood(){

        System.out.println("Se esta calentado correctamente =)");

        return true;

    }

    @Override

    protected boolean serve(){

        System.out.println("Dispulme mi inutilidad en este ambito:c");

        return false;

    }

    @Override

    public void actionLast(int num){

        action = num;

    }

    @Override

    public void battery(int discharge) {

        battery\_porcent-= discharge;

    }

    @Override

    public String name() {

        return name;

    }

    @Override

    public int id() {

        return id\_num;

    }

    public float getBattery(){

        return battery\_porcent;

    }

    @Override

    protected void chargeBattery() {

        for (int i = 0; i < 10; i++)

            System.out.println("Canrgado ..... "+(battery\_porcent+=10)+"%");

        System.out.println("CARGA COMPLETA");

    }

    @Override

    protected void setUtility(int num){}

    @Override

    protected int getUtility() {

        return 0;

    }

}

1. RobotServir

public class RobotServir extends Robot{

    public RobotServir(String name, float battery\_porcent, int id\_num, int action){

        super(name,battery\_porcent,id\_num,action);

    }

    @Override

    protected boolean cutMeat() {

        System.out.println("Hay no me corte mi dedo de metal :c");

        return false;

    }

    @Override

    protected boolean cutVegetable() {

        System.out.println("Hay no me corte mi dedo de metal :c");

        return false;

    }

    @Override

    protected boolean hotFood(){

        System.out.println("Lo termine quemando :o");

        return false;

    }

    @Override

    protected boolean serve(){

        System.out.println("Servido y listo para comer");

        return true;

    }

    @Override

    public void actionLast(int num){

        action = num;

    }

    @Override

    public void battery(int discharge) {

        battery\_porcent-= discharge;

    }

    @Override

    public String name() {

        return name;

    }

    @Override

    public int id() {

        return id\_num;

    }

    public float getBattery(){

        return battery\_porcent;

    }

    @Override

    protected void chargeBattery() {

        for (int i = 0; i < 10; i++)

            System.out.println("Canrgado ..... "+(battery\_porcent+=10)+"%");

        System.out.println("CARGA COMPLETA");

    }

    @Override

    protected void setUtility(int num) {

    }///no hace nada pero al meomento de hacerlas abstracta la clase principal me pedia usarla aqui

    ////y en el caso de donde es me pedi ponerla en la clase padre

    @Override

    protected int getUtility() {

        return 0;

    }

}