REPORT

Week 10

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 과목명 : | | | 객체지향프로그래밍 | | | | | | | | |
| 담당교수 : | | | 엄진영 | | | | | | | 교수님 | |
| 제출일 : | 2021 | | | 년 | | 05 | 월 | | 18 | | 일 |
| 공과 | | 대학 | | | 컴퓨터공학 | | | 과 | | | |
| 학번 : | | 2016112154 | | | 이름: | | | 정동구 | | | |

1.문제 1

텍스트이(가) 표시된 사진

자동 생성된 설명

텍스트이(가) 표시된 사진

자동 생성된 설명

각각 클래스는 implement한 interface가 가진 추상 메소드를 전부 구현하고 2개씩 추가로 각각 구현되었다. 구현된 함수는 코드 설명에서 설명하겠다.

2.코드

텍스트이(가) 표시된 사진

자동 생성된 설명

텍스트이(가) 표시된 사진

자동 생성된 설명

텍스트이(가) 표시된 사진

자동 생성된 설명

Life, move, on\_off interface이다. 각각 추상메소드를 가지고 있다.

class TV implements on\_off{  
  
 public int channel=0;  
 public String channel\_name;  
 public boolean on\_off = false;  
 public int prev\_channel=0;  
 public String prev\_channel\_name ="";  
 public void On()  
 {  
 if(on\_off = true)  
 {  
 this.on\_off = false;  
 System.*out*.println("TV가 켜졌습니다.");  
 }  
 else  
 {  
 System.*out*.println("TV가 이미 켜져있습니다.");  
 }  
 }  
 public void Off()  
 {  
 if(on\_off = true)  
 {  
 this.on\_off = false;  
 System.*out*.println("TV가 꺼졌습니다.");  
 }  
 else  
 {  
 System.*out*.println("TV가 이미 꺼져있습니다.");  
 }  
 }  
 public void channelUp()  
 {  
 this.prev\_channel = this.channel;  
 this.channel ++;  
 System.*out*.println("channel is "+channel);  
 }  
 public void channelDown()  
 {  
 this.prev\_channel = this.channel;  
 this.channel --;  
 System.*out*.println("channel is "+channel);  
 }  
}

TV클래스로 on\_off를 implement한다. 총 다섯개의 변수를 넣었고, on\_off에 없는 두가지의 추가 함수로 채널을 올리고 내리는 channel up, down을 구현했다.

class dog implements move,life{  
  
 public boolean eating = false;  
 public boolean sleeping = false;  
 public int age=0;  
 public String name="choco";  
 public String sex = "male";  
  
 public void front()  
 {  
 System.*out*.println("Dog moves front");  
 }//앞  
 public void back()  
 {  
 System.*out*.println("Dog moves back");  
 }//뒤  
 public void left()  
 {  
 System.*out*.println("Dog moves left");  
 }//왼  
 public void right()  
 {  
 System.*out*.println("Dog moves right");  
 }//오  
 public void eat()//식사중 여부  
 {  
 if(eating == true)  
 System.*out*.println("Dog is eating");  
 else  
 System.*out*.println("Dog is not eating");  
 }  
 public void sleep()//취침 여부  
 {  
 if(sleeping == true)  
 System.*out*.println("Dog is sleeping");  
 else  
 System.*out*.println("Dog is not sleeping");  
 }  
  
 public void changeEat()//식사 여부 변경  
 {  
 if(eating == true)  
 this.eating = false;  
 else  
 this.eating = true;  
 }  
  
 public void changeSleep() // 취침여부 변경  
 {  
 if(sleeping == true)  
 this.sleeping = false;  
 else  
 this.sleeping = true;  
 }  
}

move와 life interface를 implement하는 Dog Class이다. Move와 life interface가 가진 추상 메소드를 모두 구현하고 추가로 변수중 boolean변수인 sleeping과 eating의 T/F를 변경하는 changeSleep과 change Eat를 구현하였다. Cat과 Human도 동일하게 구현하였다.

class Airplane implements move{  
  
 public String model ="707";  
 public String destination = "Korea";  
 public String airline = "Korean Airline";  
 public boolean flying = false;  
 public boolean charging\_gas = false;  
  
 public void front()  
 {  
 System.*out*.println("airplane moves front");  
 }//앞  
 public void back()  
 {  
 System.*out*.println("airplane moves back");  
 }//뒤  
 public void left()  
 {  
 System.*out*.println("airplane moves left");  
 }//왼  
 public void right()  
 {  
 System.*out*.println("airplane moves right");  
 }//오  
  
 public void getAirline()//항공사 출력  
 {  
 System.*out*.println(airline);  
 }  
  
 public void getModel()//항공기 모델 출력  
 {  
 System.*out*.println(model);  
 }  
  
}

move interface를 implement하는 Airplane Class이다. Move interface의 추상메소드를 모두 구현하고 항공사이름과 항공기 모델명을 출력하는 getter 함수를 추가로 구현하였다.

class microwave implements on\_off{  
  
 public boolean on\_off = false;  
 public int time = 0 ; //sec  
 public int power = 0 ; // 0-100%  
 public String mode = "";  
 public boolean backlight = false;  
  
 public void On()//전원 키기  
 {  
 if(on\_off = true)  
 {  
 this.on\_off = false;  
 System.*out*.println("micorwave가 켜졌습니다.");  
 }  
 else  
 {  
 System.*out*.println("micorwave가 이미 켜져있습니다.");  
 }  
 }  
 public void Off()//전원 끄기  
 {  
 if(on\_off = true)  
 {  
 this.on\_off = false;  
 System.*out*.println("micorwave가 꺼졌습니다.");  
 }  
 else  
 {  
 System.*out*.println("micorwave가 이미 꺼져있습니다.");  
 }  
 }  
  
 public void setTime(int input)//시간 설정  
 {  
 this.time = input;  
 }  
  
 public void setPower(int power)//출력 설정  
 {  
 if(power < 0 || power >=100)  
 System.*out*.println("Poewer out of range");  
 else  
 this.power = power;  
 }  
  
}

on\_off 를 implement하는 microwave Class이다. On\_off interface의 On함수와 Off함수를 구현하고, 전자레인지 시간과 출력을 결정하는 setTime,setPower 함수를 추가로 구현하였다.