1) Write a program to display values of enums using a constructor & getPrice() method (Example display house & their prices)

enum En {

*twoBHK*(2000),

*threeBHK*(3000),

*fourBHK*(4000),

*fiveBHK*(5000);

private int price;

En(int price){

this.price=price;

}

public int getPrice() {

return price;

}

}

public class Main {

public static void main(String[] args) {

System.*out*.println("All Flat Prices");

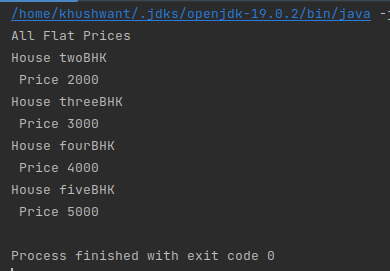
for(En n : En.*values*()){

System.*out*.println("House "+ n + "\n"+" Price " + n.getPrice());

}

}

}



2) Create a User class with fields: firstname, lastname, age, phonenumber. Write a program which accepts values of user fields, create object and **append** that to a text file. After every user creation the program should prompt: "Do you want to continue creating users? (Type QUIT to exit)" and keep on accepting values and writing to file unitl user quits.

import java.io.FileWriter;

import java.io.IOException;

import java.util.Scanner;

public class User {

private String firstname;

private String lastname;

private int age;

private String phonenumber;

public User(String firstname, String lastname, int age, String phonenumber) {

this.firstname = firstname;

this.lastname = lastname;

this.age = age;

this.phonenumber = phonenumber;

}

public static void main(String[] args) {

Scanner scanner = new Scanner(System.*in*);

String input = "";

try {

FileWriter writer = new FileWriter("users.txt", true);

while (!input.equals("QUIT")) {

System.*out*.print("Enter firstname: ");

String firstname = scanner.next();

System.*out*.print("Enter lastname: ");

String lastname = scanner.next();

System.*out*.print("Enter age: ");

int age = scanner.nextInt();

System.*out*.print("Enter phone number: ");

String phonenumber = scanner.next();

User user = new User(firstname, lastname, age, phonenumber);

writer.write(user.firstname + "," + user.lastname + "," + user.age + "," + user.phonenumber + "\n");

System.*out*.print("Do you want to continue creating users? (Type QUIT to exit) ");

input = scanner.next();

}

writer.close();

} catch (IOException e) {

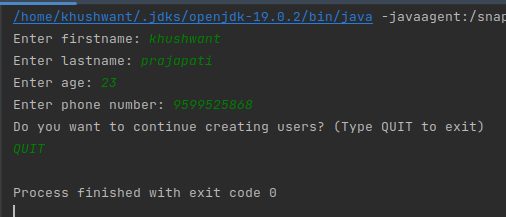
e.printStackTrace();

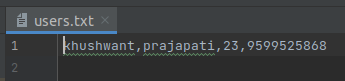
}

scanner.close();

}

}





3) Write a program to count number of occurrences of a word in a file. The file name and word should be supplied through commandline.

import java.io.BufferedReader;

import java.io.FileReader;

import java.io.IOException;

public class WordCounter {

public static void main(String[] args) throws IOException {

String filename = args[0];

String wordToCount = args[1];

int count = 0;

BufferedReader reader = new BufferedReader(new FileReader(filename));

String line;

while ((line = reader.readLine()) != null) {

String[] words = line.split("\\s+");

for (String word : words) {

if (word.equalsIgnoreCase(wordToCount)) {

count++;

}

}

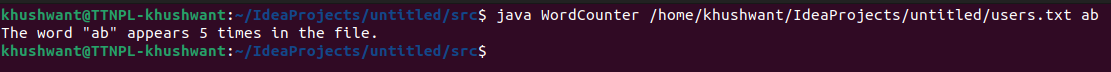
}

reader.close();

System.*out*.println("The word \"" + wordToCount + "\" appears " + count + " times in the file.");

}

}



4) Write a program to show application of Factory Design Pattern.

import java.util.Currency;

import java.util.Scanner;

interface Cur{

String getSymbol();

}

public class Q4 {

public static void main(String[] args) {

Scanner sc = new Scanner(System.*in*);

System.*out*.print("Enter Country: ");

String country = sc.nextLine();

Cur c = CurrentCountry.*current*(country);

System.*out*.println("Currency of "+country+" is : "+c.getSymbol());

}

}

class Rupee implements Cur{

@Override

public String getSymbol() {

return "Rs";

}

}

class Pounds implements Cur{

@Override

public String getSymbol() {

return "Pounds";

}

}

class USDollar implements Cur{

@Override

public String getSymbol() {

return "USD";

}

}

class CurrentCountry{

public static Cur current(String country){

if(country.equalsIgnoreCase("India")){

return new Rupee();

}else if (country.equalsIgnoreCase("UK")){

return new Pounds();

}else if (country.equalsIgnoreCase("US")){

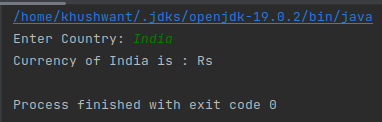
return new USDollar();

}

throw new IllegalArgumentException("No such currency");

}

}



5) Write a program to show application of Singleton Design Pattern.

public class Bulb {

public static Bulb *instance* = new Bulb();

private Bulb(){}

public static Bulb getInstance(){

return *instance*;

}

public void lightOn(){

System.*out*.println("Light is ON");

}

public void lightOff(){

System.*out*.println("Light is OFF");

}

}

public class Sc {

public static void main(String[] args) {

Bulb ob = Bulb.*getInstance*();

ob.lightOff();

ob.lightOn();

}

}

