Name:-Tanzeem Hundekari

Roll No:-2401067

**Solve below given practical questions using Java. These questions are based on**  **Chapter 2.**

**1.Write an interface, TemperatureConverter. Add methods, convertToFarenheit() and convertToCelsius(). Add a static method about() which will print the purpose of this interface. Implement the interface into TemperatureConverterImplementation class and override the necessary methods. Create objects of the business class in TemperatureConverterDemo class and demonstrate the behaviour of all the methods.**

→**TemperatureConverter.java**   
**package** pkginterface;

**publicinterface** TemperatureConverter {   
 **publicdouble** convertToFahrenheit(**double** celsius);   
 **publicdouble** convertToCelsius(**double** fahrenheit);   
 **publicstaticvoid** about() {   
 System.***out***.println("This interface provides methods to convert temperature between Celsius and Fahrenheit.");   
 }

}   
→**TemperatureConverterImplementation.java package** pkginterface;

**publicclass** TemperatureConverterImplementation **implements** TemperatureConverter{   
 **publicdouble** convertToFahrenheit(**double** celsius) { **return**(celsius\*9/5)+32;   
 }   
 **publicdouble** convertToCelsius(**double** fahrenheit) { **return**(fahrenheit-32)\*5/9;   
 }   
}   
→**TemperatureConverterDemo.java**   
**package** pkginterface;

**import** java.util.Scanner;

**publicclass** TemperatureConverterDemo {   
 **publicstaticvoid** main(String[] args) {

//Call to static method   
TemperatureConverter.*about*();

//Create an object of implementation class TemperatureConverter converter=**new** TemperatureConverterImplementation();

//Create a Scanner object for user input   
Scanner scanner=**new** Scanner(System.***in***);

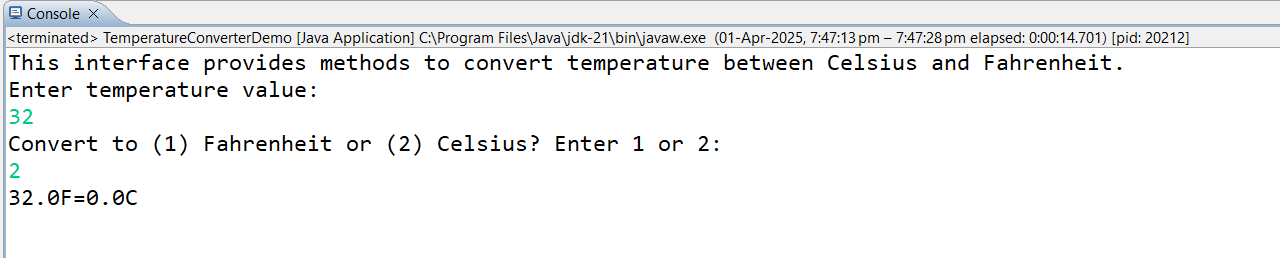
System.***out***.println("Enter temperature value:");   
**double** temperature=scanner.nextDouble();

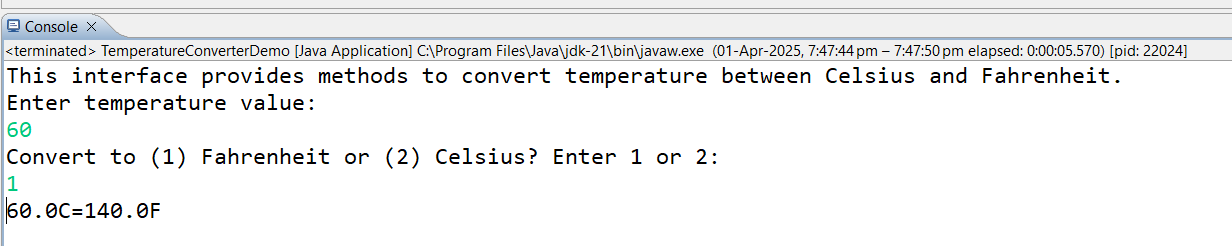
System.***out***.println("Convert to (1) Fahrenheit or (2) Celsius? Enter 1 or 2:");   
 **int** choice=scanner.nextInt();

**if**(choice==1) {   
**double** fahrenheit=converter.convertToFahrenheit(temperature); System.***out***.println(temperature+"C="+fahrenheit+"F");   
}**elseif**(choice == 2) {   
**double** celsius=converter.convertToCelsius(temperature); System.***out***.println(temperature+"F="+celsius+"C");   
}**else** {   
System.***out***.println("Invalid choice!");   
}   
scanner.close();

}

}   
**Output Screens:-**





**2.Write a Java programming to create a banking system with three classes - Bank, Account, SavingsAccount, and CurrentAccount. The bank should have a list of accounts and methods for adding them. Accounts should be an interface with methods to deposit, withdraw, calculate interest, and view balances.**

**SavingsAccount and CurrentAccount should implement the Account interface and have their own unique methods.**

→**Account.java**   
**package** pkgbank;

**publicinterface** Account {   
**void** deposit(**double** amount);   
**void** withdraw(**double** amount);   
**double** calculateInterest();   
**double** viewBalance();

}

→**Bank.java**   
**package** pkgbank;   
**import** java.util.ArrayList;

**publicclass** Bank {   
 **private** ArrayList<Account> accounts=**new** ArrayList<>();   
 **publicvoid** addAccount(Account account) {   
 accounts.add(account);   
 System.***out***.println("Account added to the bank");   
 }   
 **publicvoid** displayAllBalances() {   
 **for**(Account account:accounts) {   
 System.***out***.println("Account Balance: "+account.viewBalance()); }   
 }

}   
→**SavingsAccount.java**   
**package** pkgbank;

**publicclass** SavingsAccount **implements** Account{

**privatedouble** balance;   
**privatedouble** interestRate;

**public** SavingsAccount(**double** balance,**double** interestRate) {

**this**.balance=balance;   
 **this**.interestRate=interestRate;   
}

**publicvoid** deposit(**double** amount) {   
 **if** (amount > 0) {   
 balance += amount;   
 System.***out***.println("Deposited: " + amount);   
 } **else** {   
 System.***out***.println("Invalid deposit amount.");   
 }   
 }   
 **publicvoid** withdraw(**double** amount) {   
 **if** (balance>=amount) {   
 balance-=amount;   
 System.***out***.println("Withdrawn"+amount); }**else** {   
 System.***out***.println("Insufficient balance!"); }   
 }

**publicdouble** calculateInterest() {   
 **return** balance\*(interestRate/100);   
}

**publicdouble** viewBalance() {   
 System.***out***.println("SavingsAccount Balance:"); **return** balance;   
}

**publicvoid** applyInterest() {   
 **double** interest=calculateInterest();   
 deposit(interest);   
 System.***out***.println("Interest applied:"+interest); }

}   
→**CurrentAccount.java**   
**package** pkgbank;

**publicclass** CurrentAccount **implements** Account{ //private String accountNumber;   
**privatedouble** balance;   
**privatedouble** overdraftLimit;

**public** CurrentAccount(**double** balance,**double** overdraftLimit) { //this.accountNumber=accountNumber;   
 **this**.balance=balance;   
 **this**.overdraftLimit=overdraftLimit;   
}   
**publicvoid** deposit(**double** amount) {   
 **if** (amount > 0) {

balance += amount;   
 System.***out***.println("Deposited: " + amount);   
 } **else** {   
 System.***out***.println("Invalid deposit amount.");   
 }   
 }   
 **publicvoid** withdraw(**double** amount) {   
 **if**(amount <= balance + overdraftLimit) {   
 balance-= amount;   
 System.***out***.println("Withdrawn:"+amount); }**else** {   
 System.***out***.println("Overdraft limit exceeded."); }   
 }

**publicdouble** calculateInterest() {   
 System.***out***.println("Current account do not earn interest"); **return** 0;   
}

**publicdouble** viewBalance() {   
 System.***out***.println("Current Account Balance:");   
 **return** balance;   
}   
**publicvoid** setOverdraftLimit(**double** overdraftLimit) {   
 **this**.overdraftLimit=overdraftLimit;   
 System.***out***.println("Overdraft limit set to:"+overdraftLimit); }

}   
→**BankDemo.java**   
**package** pkgbank;

**publicclass** BankDemo {

**publicstaticvoid** main(String[] args) {   
 Bank bank=**new** Bank();

//Create accounts   
SavingsAccount savings=**new** SavingsAccount(1000,5); CurrentAccount current=**new** CurrentAccount(2000,500);

//Add accounts to bank   
bank.addAccount(savings);   
bank.addAccount(current);

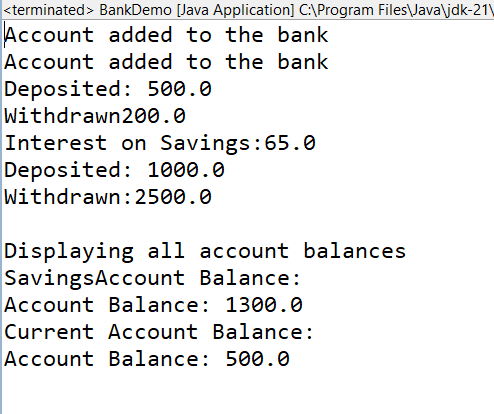
//Perform operations   
savings.deposit(500);

savings.withdraw(200);   
System.***out***.println("Interest on Savings:"+savings.calculateInterest()); current.deposit(1000);   
current.withdraw(2500);

//Display all account balances   
System.***out***.println("\nDisplaying all account balances"); bank.displayAllBalances();

}

}   
**Output Screens:-**



**3.Write an interface, SortingUtility having methods, ascendingSort() and**   
**descendingSort(). Implement this in SortArray class and SortString class which will inturn override the two abstract methods in the interface. Create necessary demo class and call relevant methods on the objects**   
**I/P : If the string input is Apple then the sorted ascending output should be Aelpp and descending output should be ppleA**

→**SortUtility.java**

**package** pkgSorting;

**publicinterface** SortingUtility {

**publicvoid** ascendingSort();

**publicvoid** descendingSort();

}   
→**SortArray.java**   
package pkgSorting;   
import java.util.Arrays;   
import java.util.\*;

public class SortArray implements SortingUtility{   
 String[] array;   
 public SortArray(String[] array) {   
 this.array=array;   
 }   
 public void ascendingSort() {   
 Arrays.sort(this.array);   
 for(String arr:this.array)   
 System.out.println(arr);   
 }   
 public void descendingSort() {   
 Arrays.sort(this.array,Collections.reverseOrder()); for(String arr:this.array)   
 System.out.println(arr);   
 }

}   
→**SortString.java**   
**package** pkgSorting;

**import** java.util.Arrays;

**publicclass** SortString **implements** SortingUtility{

**private** String str;   
**public** SortString(String str) {   
 **this**.str=str;

}   
**publicvoid** ascendingSort() {   
 **char**[] charArray=str.toCharArray();   
 Arrays.*sort*(charArray);   
 System.***out***.println("Ascending Order:"+**new** String(charArray));; }   
**publicvoid** descendingSort() {   
 **char**[] charArray=str.toCharArray();   
 Arrays.*sort*(charArray);   
 StringBuilder sb=**new** StringBuilder(**new** String(charArray)); System.***out***.println("Descending Order:"+sb.reverse().toString()); }

}   
→**SortDemo.java**   
**package** pkgSorting;

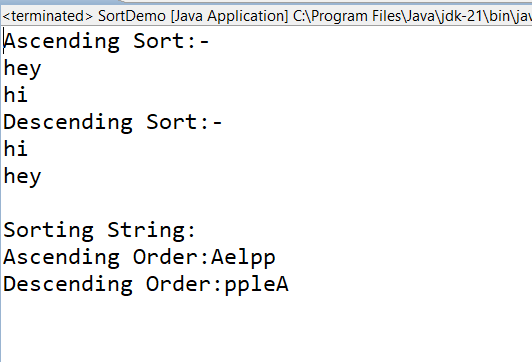
**publicclass** SortDemo {

**publicstaticvoid** main(String[] args) {   
//Sorting an array   
String str[]= {"hi","hey"};   
SortArray obj=**new** SortArray(str);   
System.***out***.println("Ascending Sort:-"); obj.ascendingSort();   
System.***out***.println("Descending Sort:-"); obj.descendingSort();

//Sorting a string   
String word="Apple";   
SortString sortString=**new** SortString(word);   
System.***out***.println("\nSorting String:");   
sortString.ascendingSort();   
sortString.descendingSort();

}

}   
**Output Screens:-**



**4.Write a Java program to create an interface Encryptable with methods encrypt (String data) and decrypt (String encryptedData) that define encryption and decryption operations. Create two classes AES and RSA that implement the Encryptable interface and provide their own encryption and decryption algorithms.**

→**Encryptable.java**   
**package** pkgEncryption;

**publicinterface** Encryptable {   
String encrypt(String data);   
String decrypt(String encryptedData);

}   
→**AES.java**   
**package** pkgEncryption;

**publicclass** AES **implements** Encryptable{   
 **public** String encrypt(String data)   
 {   
 //Using StringBuildere for AES encryption   
 //StringBuilder is a mutable class String objects are immutable   
 StringBuilder encryptedData=**new** StringBuilder();   
 **for**(**char** c:data.toCharArray())//converts string into character array {   
 //Shift each character by 1 placeholder for encryption   
 encryptedData.append((**char**)(c+1));   
 //shifts ASCII value by 1 & converts them back to char & stores it in object encryptedData   
 }   
 System.***out***.println("Encrypting data using AES algorithm");   
 **return** encryptedData.toString();   
 }   
 **public** String decrypt(String encryptedData)   
 {   
 StringBuilder decryptedData=**new** StringBuilder();   
 **for**(**char** c:encryptedData.toCharArray())   
 {   
 decryptedData.append((**char**)(c-1));   
 }   
 System.***out***.println("Decrypting data using AES algorithm");   
 **return** decryptedData.toString();   
 }

}   
→**RSA.java**   
**package** pkgEncryption;

**publicclass** RSA **implements** Encryptable{   
 **public** String encrypt(String data)   
 {   
 StringBuilder encryptedData=**new** StringBuilder();   
 **for**(**char** c:data.toCharArray())   
 {   
 encryptedData.append((**char**)(c+2));   
 }   
 System.***out***.println("Encrypting data using RSA algorithm"); **return** encryptedData.toString();

}

**public** String decrypt(String encryptedData)   
{   
 StringBuilder decryptedData=**new** StringBuilder();   
 **for**(**char** c:encryptedData.toCharArray())   
 {   
 decryptedData.append((**char**)(c-2));   
 }   
 System.***out***.println("Decrypting data using RSA algorithm"); **return** decryptedData.toString();   
}

}   
→**EncryptableDemo.java**   
**package** pkgEncryption;   
**import** java.util.\*;

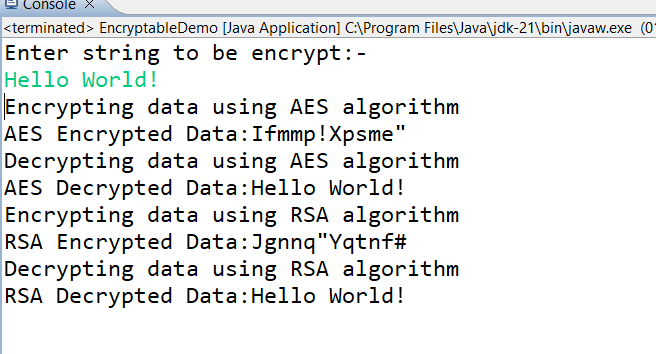
**publicclass** EncryptableDemo {

**publicstaticvoid** main(String[] args) {   
Scanner scanner=**new** Scanner(System.***in***);   
System.***out***.println("Enter string to be encrypt:-");   
String data=scanner.nextLine();   
AES obj1=**new** AES();   
String aesEncrypted=obj1.encrypt(data);   
System.***out***.println("AES Encrypted Data:"+aesEncrypted); String aesDecrypted=obj1.decrypt(aesEncrypted);   
System.***out***.println("AES Decrypted Data:"+aesDecrypted); RSA obj2=**new** RSA();

String rsaencryptedData=obj2.encrypt(data);   
System.***out***.println("RSA Encrypted Data:"+rsaencryptedData); String rsadecryptedData=obj2.decrypt(rsaencryptedData); System.***out***.println("RSA Decrypted Data:"+rsadecryptedData);

}

}   
**Output Screens:-**



**5.You have developed an e-commerce website for your client. The maximum no of units of a single prodcut that one user can add to the cart is 5. If the user adds more than 5 units of a single product, then your application is expected to throw, MaximumProductsLimitExceededException. Write a custom exception class to achieve this.**

→**MaxProdLimitExceededException**   
**package** pkgtablet;

**publicclass** MaxProdLimitExceededException **extends** Exception { **public** MaxProdLimitExceededException(String message) { **super**(message);   
 }

}   
→**ECommerceApp.java**   
package pkgProducts;   
import java.util.Scanner;

import pkgtablet.MaxProdLimitExceededException;

public class ECommerceApp {

public static void main(String[] args) {   
 Scanner scanner=new Scanner(System.in);   
 ShoppingCart cart=new ShoppingCart();   
 try {   
 System.out.println("Enter number of units you want to add to the cart:");   
 int units=scanner.nextInt();

cart.addProductToCart(units);   
 }catch(MaxProdLimitExceededException e) {   
 System.out.println("Error: "+e.getMessage());//Handles custom exception   
 }finally {   
 scanner.close();

}

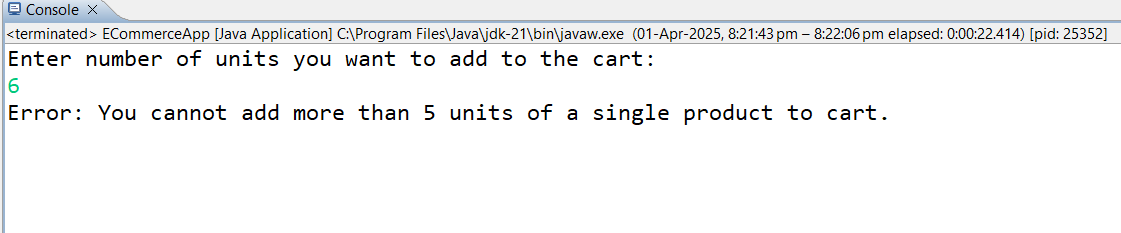
}

}   
→**Shopping Cart.java**   
**package** pkgProducts;

**import** pkgtablet.MaxProdLimitExceededException;

**publicclass** ShoppingCart {   
 **privatestaticfinalint*MAX\_PRODUCT\_LIMIT***=5;   
 **publicvoid** addProductToCart(**int** quantity)**throws** MaxProdLimitExceededException {   
 **if**(quantity>***MAX\_PRODUCT\_LIMIT***) {   
 **thrownew** MaxProdLimitExceededException(   
 "You cannot add more than "+***MAX\_PRODUCT\_LIMIT***+" units of a single product to cart."   
 );   
 }   
 //Logic to add product to cart   
 System.***out***.println(quantity+" units added to the cart.");   
 }

}   
 **Output Screens:-**



**6.The manufacturing of your medical company has very strict standards of product specifications. After each pill /tablet is ready, it is weighed. If the weight of the tablet exceeds the allowed limit, TabletWeightExceededException is raised. Using exception handling in Java, write the program to achieve the above business requirement.**

→**Tablet.java**   
**package** pkgtablet;

**publicclass** Tablet {   
**private** String name;   
**privatedouble** weight;   
**privatestaticfinaldouble*MAX\_WEIGHT***=500.0;

**public** Tablet(String name,**double** weight) **throws**   
TabletWeigthExceededException{   
 **if**(weight>***MAX\_WEIGHT***) {   
 **thrownew** TabletWeigthExceededException("Error: "+name+" exceeds the maximum allowed weight of "+***MAX\_WEIGHT***+" mg.");   
 }   
 **this**.name=name;   
 **this**.weight=weight;   
 System.***out***.println("Tablet "+name+" with weight "+weight+" mg is approved.");   
 }   
 **public** String getName() {   
 **return** name;   
 }   
 **publicdouble** getWeight() {   
 **return** weight;   
 }

}   
→**TabletWeightExceededException.java**   
**package** pkgtablet;

**publicclass**TabletWeigthExceededException **extends** Exception{ **public** TabletWeigthExceededException(String message) { **super**(message);

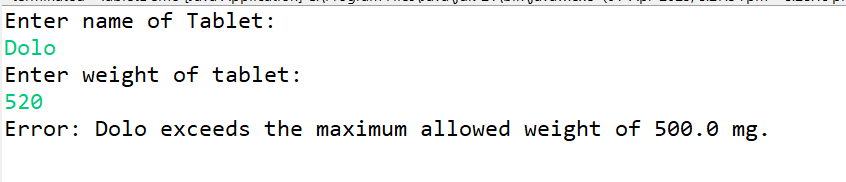
}

}   
→**TabletDemo.java**   
**package** pkgtablet;   
**import** java.util.\*;   
**publicclass** TabletDemo {

**publicstaticvoid** main(String[] args) {   
 Scanner scanner=**new** Scanner(System.***in***);   
 **try** {   
 System.***out***.println("Enter name of Tablet:");   
 String name=scanner.nextLine();   
 System.***out***.println("Enter weight of tablet:");   
 **double** weight=scanner.nextDouble();   
 Tablet tablet1=**new** Tablet(name,weight);   
 //Tablet tablet2=new Tablet("Dolo",500.0);   
 //Tablet tablet3=new Tablet("Aspirin",520.0);//This will throw exception   
 }   
 **catch**(TabletWeigthExceededException e) {   
 System.***out***.println(e.getMessage());   
 }

}

}   
**Output Screens:-**



**7.When the battery of your mobile phone is less than 20%, the system should generate, LowBatteryException to alert the user to start charging the device. If the battery goes lower than 10 then the system should raise,**   
**InsufficientChargeException and put the unit on power saver mode. Using exception handling in Java, write the program to achieve the above business requirement.**

→**InsufficientChargeException.java**   
**package** pkgBattery;

**publicclass**InsufficientCharge**extends** Exception { **public** InsufficientCharge(String message) {

**super**(message);   
}

}   
→**LowBatteryException.java**   
**package** pkgBattery;

**publicclass**LowBattery**extends** Exception {   
 **public** LowBattery(String message) {   
 **super**(message);   
 }

}   
→**MobilePhone.java**   
**package** pkgBattery;

**publicclass** MobilePhone {   
 **privateint** batteryLevel;

**public** MobilePhone(**int** batteryLevel) **throws** LowBattery,InsufficientCharge { **this**.batteryLevel=batteryLevel;   
 **if**(batteryLevel<10) {   
 **thrownew** InsufficientCharge("Battery critically low (" + batteryLevel + "%)! Switching to Power Saver Mode.");   
 }**elseif**(batteryLevel<20) {   
 **thrownew** LowBattery("Battery low (" + batteryLevel + "%)!

Please charge soon.");   
 }**else** {   
 System.***out***.println("No need to charge!"); }   
 }   
 **publicint** getBatteryLevel() {   
 **return** batteryLevel;   
 }

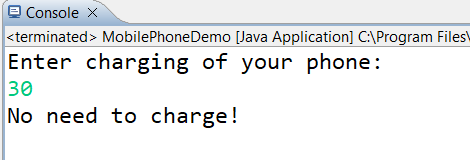
}   
→**MobilePhoneDemo.java**   
**package** pkgBattery;   
**import** java.util.\*;

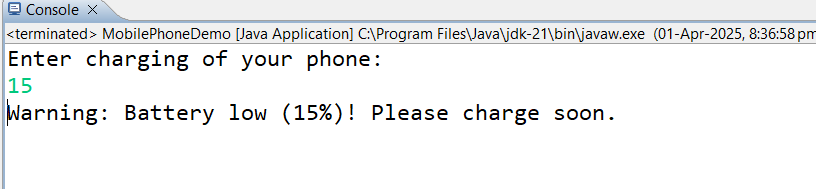
**publicclass** MobilePhoneDemo {

**publicstaticvoid** main(String[] args) {   
Scanner scanner=**new** Scanner(System.***in***);   
**try** {   
System.***out***.println("Enter charging of your phone:"); Integer batteryLevel=scanner.nextInt();   
MobilePhone phone=**new** MobilePhone(batteryLevel);

}   
 **catch**(LowBattery e) {   
 System.***out***.println("Warning: "+e.getMessage()); }**catch**(InsufficientCharge e) {   
 System.***out***.println("Critical: "+e.getMessage()); }   
}

}   
**Output Screen:-**





**Submission Details:**   
**Submit your soft copy of the executable code and relevant output screen shots in a single pdf file named. "JavaLab2\_RollNbr\_StudentNm.pdf"**