**[Inheritance, Polymorphism, Abstract Class, Interface, Lambda](https://capgemini.tekstac.com/course/view.php?id=438" \l "sectionwrapper-7)**

1. Mother Teresa Endowment is one of the famous endowments in the city. In their last board meeting, board members have decided to automate their holder details for improvement and better maintenance of records

You, being their software consultant, have been approached to develop software to implement the functionality of generating the endowment amount based on the endowment type.

**Requirement 1: Calculation of Endowment amount**

The application needs to calculate the endowment amount to be paid for the holder according to the endowment type

**Component Specification: Endowment**

|  |  |  |  |
| --- | --- | --- | --- |
| **Type(Class)** | **Attributes** | **Methods** | **Responsibilities** |
| **Endowment** | String endowmentId  String holderName  String endowmentType  String registrationDate | Include the getter and setter method for all the attributes.  Include a public parameterized constructor of four arguments in the following order -endowmentId, holderName, endowmentType, registrationDate to initialize the values for the Endowment object |  |
| **Endowment** |  | public abstract double**calculateEndowment()** |  |

**Note:**The attributes of the Endowment class should be protected and the methods should be public

**Component Specification: EducationalEndowment**(Needs to be a child of the Endowment class)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Component Name** | **Type (Class)** | **Attributes** | **Methods** | **Responsibilities** |
|  | **EducationalEndowment** | String educationalInstitution  String educationalDivision | Include the getter and setter method for all the attributes.  Include a public   parameterized constructor of six arguments in the following order - endowmentId, holderName, endowmentType, registrationDate, educationalInstitution, educationalDivision to initialize the values for the EducationalEndowment object. |  |
| Calculation of the Endowment amount | **EducationalEndowment** |  | public double **calculateEndowment()** | This method should calculate the endowment amount to be paid for the holder based on the educationalDivision and return the calculated endowment amount    If the educationalDivision is "School" then the  Endowment amount should be 30000    If the educationalDivision is "UnderGraduate" then the  Endowment amount should be 60000    If the educationalDivision is "PostGraduate" then the  Endowment amount should be 90000    **Note:** educationalDivision is caseinsensitive |

**Note:**The methods and the constructor should be public, and the attributes of the class should be private

**Component Specification: HealthEndowment**(Needs to be a child of the Endowment class)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Component Name** | **Type (Class)** | **Attributes** | **Methods** | **Responsibilities** |
|  | **HealthEndowment** | String healthCareCenter  int holderAge | Include the getter and setter method for all the attributes.  Include a public parameterized constructor of six arguments in the following order - endowmentId, holderName, endowmentType, registrationDate, healthCareCenter, holderAge to initialize the values for the HealthEndowment object |  |
| Calculation of the Endowment amount | **HealthEndowment** |  | public double **calculateEndowment()** | This method should calculate the endowment amount to be paid for the holder based on the holderAge and return the calculated endowment amount    If the holderAge is less than or equal to 30 then the Endowment amount shoud be 120000    If the holderAge is greater than 30 and  less than 60 then the Endowment amount should be 200000    If the holderAge is greater than or equal to 60 then the Endowment amount should be 500000 |

**Note:**The attributes of HealthEndowment class should be private and methods as public

Use a **public class UserInterface** with the main method to test the application.  In the main method get the endowment holder details as shown in the sample input.

Get the endowment type from the user.

**Note:** Endowment type is case insensitive

If the endowment type is educational, then create an object for the EducationalEndowment and invoke the **calculateEndowment**(), and display the returned amount as output (refer to sample output)

If the endowment type is health, then create an object for the HealthEndowment and invoke the **calculateEndowment**(), and display the returned amount as output (refer to sample output)

If the endowment type is invalid then print "**<endowment type> is an invalid endowment type**" and terminate the program.

**Note:**

* Use System.out.printf("%.2f", endowmentamount).
* In the Sample Input / Output provided, the highlighted text in bold corresponds to the input given by the user and the rest of the text represents the output.
* Ensure to follow the object-oriented specifications provided in the question.
* Ensure to provide the names for classes, attributes and methods as specified in the question.
* Adhere to the code template, if provided.
* Please do not use **System.exit(0**) to terminate the program.

**Sample Input/Output 1:**

Enter Endowment Id

**EID101**

Enter Holder Name

**Babu**

Enter Endowment Type

**Educational**

Enter Registration Date

**28/5/2021**

Enter Educational Institution

**AMC**

Enter Educational Division

**undergraduate**

Endowment Amount 60000.00

**Sample Input/Output 2:**

Enter Endowment Id

**EID205**

Enter Holder Name

**Ezhil**

Enter Endowment Type

**health**

Enter Registration Date

**10/4/2021**

Enter Health Care Center

**Queens**

Enter Holder Age

**40**

Endowment Amount 200000.00

**Sample Input/Output 3:**

Enter Endowment Id

**EID205**

Enter Holder Name

**Ezhil**

Enter Endowment Type

**Grocery**

Grocery is an invalid endowment type

2. Vivek Furniture Store is one of the most famous furniture stores in the city. They want to automate their customer details and the total price. As an initiative, they want the system to store the customer details and help them generate the total price.

You, being their software consultant, have been approached to develop software to implement the functionality of generating the total price based on the bero type.

**Component Specification: CustomerDetails**

|  |  |  |  |
| --- | --- | --- | --- |
| **Type(Class)** | **Attributes** | **Methods** | **Responsibilities** |
| CustomerDetails | String customerName  long phoneNumber  String address | Include the getter and setter method for all the attributes.  Include a public parametrized constructor of three arguments in the following order - customerName, phoneNumber and address to initialize the values for the CustomerDetails object |  |

**Note:**The attributes of the CustomerDetails class should be private and the methods should be public

**Requirements 1:To calculate Total Price**

The application needs to calculate the total price to be paid by the customer according to the beroType

**Component Specification: Bero (Abstract Class)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Type(Class)** | **Attributes** | **Methods** | **Responsibilities** |
| Bero | String beroType  String beroColour  double price | Include the getter and setter method for all the attributes.  Include a parametrized constructor of two arguments in the following order - beroType, beroColourto intialize the values for the Bero object |  |
| Bero |  | public abstract void **calculatePrice**() |  |

**Note:**The attributes of the Bero class should be protected and the methods should be public

**Component Specification: SteelBero**(Needs to be a child of the Bero class)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Component Name** | **Type (Class)** | **Attributes** | **Methods** | **Responsibilities** |
| Calculating the Total Price | **SteelBero** | int beroHeight | Include the getter and setter method for all the attributes.  Include a public parametrized constructor of three arguments in the following order - beroType, beroColour, beroHeight to intialize the values for the SteelBero object |  |
| Calculating the Total Price | **SteelBero** |  | public void **calculatePrice()** | This method should calculate the total price to be paid by the customer based on the beroHeight and setPrice of the bero    If the beroHeight is 3 feet, then the price should be Rs 5000    If the beroHeight is 5 feet, then the price should be Rs 8000    If the beroHeight is 7 feet, then the price should be Rs 10000 |

**Note:**The attributes of the SteelBero class should be private and methods as public

**Component Specification: WoodenBero**(Needs to be a child of the Bero class)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Component Name** | **Type (Class)** | **Attributes** | **Methods** | **Responsibilities** |
| Calculating the Total Price | **WoodenBero** | String woodType | Include the getter and setter method for all the attributes.  Include a public parametrized constructor of three arguments in the following order - beroType, beroColour, woodType to intialize the values for the WoodenBero object |  |
| Calculating the Total Price | **WoodenBero** |  | public void **calculatePrice**() | This method should calculate the total price to be paid by the customer based on the woodType and the setPrice of the bero    If the woodType is "Ply Wood" then the price should be Rs 15000    If the woodType is "Teak Wood" then the price should be Rs 12000    If the woodType is "Engineered Wood" then the price should be Rs 10000    **Note :** woodType is case sensitive |

**Note:**The attributes of the WoodenClass class should be private and methods as public

**Requirements 2:** **To calculate the discount on the Bero**

The application needs to calculate the discount on the beroType

**Component Specification: Discount**

|  |  |  |  |
| --- | --- | --- | --- |
| **Component Name** | **Type (Class)** | **Methods** | **Responsibilities** |
| Calculating the discount on the bero | Discount | public double **calculateDiscount**(Bero bObj) | This method is used to calculate the discount for the bero objects and return the discount amount  If the object is SteelBero then the discount should be 10%  If the object is WoodenBero then the discount should be 15% |

**Example:**

If the object is SteelBero, price = 5000, then the discount will be:

Discount = (price \* 10) /100

               = (5000 \* 10) / 100

Discount = 500

Use a **public class UserInterface**with the main method to test the application.  In the main method get the customer details as shown in the sample input .

Get the bero type from the user.

**Note:**bero type is case sensitive

If the bero type is "Steel Bero" then create an object for the SteelBero and invoke the **calculatePrice**(), and display the returned amount as output (refer sample output)

If the bero type is "Wooden Bero" then create an object for the WoodenBero and invoke the **calculatePrice**(), and display the returned amount as output (refer sample output)

If the bero type is invalid then print "**<bero type> is an invalid bero type**" and terminate the program.

**Example :**

Total price calculation

If price = 5000, discount = 500, then the total price will be:

Total price = price - discount

                  = 5000 - 500

                  = 4500.00

The total price should end with two digits after the decimal points

Use System.out.printf("%.2f", totalprice).

**Note:**

* In the Sample Input / Output provided, the highlighted text in bold corresponds to the input given by the user and the rest of the text represents the output.
* Ensure to follow the object-oriented specifications provided in the question.
* Ensure to provide the names for classes, attributes and methods as specified in the question.
* Adhere to the code template, if provided.
* Please do not use System.exit(0) to terminate the program.

**Sample Input/Output1**

Enter Customer Name

**Barath**

Enter Phone Number

**9876543210**

Enter address

**North Street**

Enter Bero Type

**Wooden Bero**

Enter Bero Colour

**Brown**

Enter Wood Type

**Ply Wood**

Amount needs to be paid 12750.00

**Sample Input/Output 2**

Enter Customer Name

**Guru**

Enter Phone Number

**7894561230**

Enter address

**South Street**

Enter Bero Type

**Steel Bero**

Enter Bero Colour

**Blue**

Enter Bero Height

**7**

Amount needs to be paid 9000.00

**Sample Input/Output 3**

Enter Customer Name

**Guru**

Enter Phone Number

**7894561230**

Enter address

**South Street**

Enter Bero Type

**Cupboard**

Cupboard is an invalid bero type

3. Kannan Departmental Store has decided to open its online shopping portal. By automating they can easily enter the customer details, award bonus points and calculate the delivery charge as well.

You as their software consultant have been approached to develop software to implement the functionality of generating the bonus points and calculating the delivery charges.

**Component Specification: CustomerDetails**

|  |  |  |  |
| --- | --- | --- | --- |
| **Type(Class)** | **Attributes** | **Methods** | **Responsibilities** |
| **CustomerDetails** | String customerName  String phoneNumber  String streetName  double billAmount  int distance | Include the getter and setter methods for all the attributes.  Include a public parametrized constructor of five arguments in the following order - customerName, phoneNumber, streetName, billAmount, distance to intialize the values for the CustomerDetails object |  |

**Note:**The attributes of the CustomerDetails class should be private and the methods should be public.

Write a public interface **BonusPoints** with an abstract method "**double** **calculateBonusPoints**()".

Write a public interface **DoorDelivery** with an abstract method **"double deliveryCharge()".**

The above class **CustomerDetails** should implement the Interfaces **BonusPoints** and **DoorDelivery**.

**Requirement 1: To calculate the Bonus Points**

The application needs to calculate the bonus points to be awarded to the customer based on the bill amount.

**Component Specification: CustomerDetails**

|  |  |  |  |
| --- | --- | --- | --- |
| **Component Name** | **Type (Class)** | **Methods** | **Responsibilities** |
| Calculating bonus points for the customer | **CustomerDetails** | **public** **double** **calculateBonusPoints**() | This method should calculate the bonus points to be awarded to the customer based on the bill amount and return the bonus points  If the billAmount is greater than or equal to 250, the customer is eligible to get bonus points  So the bonus points will be calculated as billAmount/10  If the billAmount is less than 250 return 0 |

**Example:**

If billAmount = 480, then the Bonus Points will be:

Bonus Points   = billAmount / 10

                        = 480 / 10

                        = 48.0

**Requirement 2: To calculate the Delivery charges**

The application needs to calculate the delivery charges  to be paid by the customer according to the distance

**Component Specification: CustomerDetails**

|  |  |  |  |
| --- | --- | --- | --- |
| **Component Name** | **Type (Class)** | **Methods** | **Responsibilities** |
| Calculating delivery charges for the customer | **CustomerDetails** | **public double deliveryCharge()** | This method should calculate the delivery charges to be paid by the customer based on the distance and return the delivery charges.  If the distance is greater than or equal to 25, the delivery charges should be Rs 8/km.  If the distance is greater than or equal to 15 and less than 25, the delivery charges should be Rs 5/ km  If the distance is less than 15 km, the delivery charges should be Rs 2/ km  So the delivery charges should be calculated as distance\*delivery charges per km |

**Example:**

If distance = 10, then the Delivery  charges will be:

Delivery charges = distance \* delivery charges per km

                          = 10 \* 2

                          = 20.0

Use a **public class UserInterface**with the **main** method to test the application.  In the **main** method get the customer details as shown in the sample input .

Create an object for CustomerDetails and invoke the **calculateBonusPoints**() method from the CustomerDetails class to find the bonus points, and display the returned points as output(refer sample output).

Invoke the **deliveryCharge**() method from the CustomerDetails class to find the delivery charges, and display the returned amount as output(refer sample output).

**Note:**

* In the Sample Input / Output provided, the highlighted text in bold corresponds to the input given by the user and the rest of the text represents the output.
* Ensure to follow the object oriented specifications provided in the question.
* Ensure to provide the names for classes, attributes and methods as specified in the question.
* Adhere to the code template, if provided.
* Please do not use System.exit(0) to terminate the program.

**Sample Input/Ouput 1**

Enter the customer name

**Mikkel**

Enter the phone number

**9513578462**

Enter the street name

**Winden street**

Enter the bill amount

**670**

Enter the distance

**18**

Customer name: Mikkel

Phone number: 9513578462

Street name: Winden street

Bonus points: 67.0

Delivery charge: 90.0

.

**Sample Input/Ouput 2**

Enter the customer name

**Hannah**

Enter the phone number

**9784561230**

Enter the street name

**Newton street**

Enter the bill amount

**150**

Enter the distance

**10**

Customer name: Hannah

Phone number: 9784561230

Street name: Newton street

Bonus points: 0.0

Delivery charge: 20.0

4.

KIT Institutions have opened admissions for freshers. So they decided to automate their server to calculate the college fee based on the student's category (Hosteller or Dayscholar). As an initiative, the institute wants the system to store the student details and help them to generate the total college fee.

You as their software consultant have been approached to develop software to implement the functionality of generating the total college fee based on the category.

**Requirement 1:To calculate the Total Fee**

The application needs to calculate the total fee to be paid by the student according to the category

**Component Specification: Student (Abstract Class)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Type(Class)** | **Attributes** | **Methods** | **Responsibilities** |
| **Student** | int studentId  String studentName  String department  String gender  String category  double collegeFee | Include the getters and setters method for all the attributes.  Include a public parameterized constructor of six arguments in the following order - studentId,studentName,department,gender,category,collegeFee to initialize the values for the Student object |  |
| **Student** |  | **public abstract double calculateTotalFee()** |  |

**Note:The attributes of the Student class should be protected and the methods should be public**

**Component Specification: Hosteller**(Needs to be a child of the Student class)

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Component Name** | **Type (Class)** | **Attributes** | **Methods** | | | | | **Responsibilities** |
|  | **Hosteller** | int roomNumber  char blockName  String roomType | Include the getters and setters method for all the attributes.  Include a public parametrized constructor of nine arguments in the following order - studentId,studentName,department,gender,category,collegeFee, roomNumber, blockName, roomType to initialize the values for the Hosteller object | | | | |  |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | | **Component Name** | **Type (Class)** | **Attributes** | **Methods** | **Responsibilities** | |  | **DayScholar** | int busNumber  float distance | Include the getters and setters method for all the attributes.  Include a parametrized constructor of eight arguments in the following order - studentId,studentName,department,gender,category,collegeFee, busNumber,distance to intialize the values for the DayScholar object. |  | | Calculate the Total Fee | **DayScholar** |  | **public double calculateTotalFee()** | This method should calculate the total fee to be paid by the student based on the distance and return the calculated amount    If the distance is greater than 30 and less than or equal to 40 the busFee should be Rs 28000    If the distance is greater than 20 and less than or equal to 30 the busFee should be Rs 20000    If the distance is greater than 10 and less than or equal to 20 the busFee should be Rs 12000    If the distance is less than or equal to 10 the busFee should be Rs 6000 |     Calculate the Total Fee | | | | **Hosteller** |  | **public double calculateTotalFee()** | This method should calculate the total fee to be paid by the student based on the blockName and roomType and return the calculated amount  If the blockName is A then the hostelFee should be Rs 60000 and if the roomType is AC then the fee should be Rs 8000  If the blockName is B then the hostelFee should be Rs 50000 and if the roomType is AC then the fee should be Rs 5000  If the blockName is C then the hostelFee should be Rs 40000 and if the roomType is AC then the fee should be Rs 2500  Hostel fee is common for both AC and Non - AC category  For an AC room, the total amount will be **collegeFee+hostelFee+roomTypeAmount**  For a Non-AC room, the  total amount will be **collegeFee+hostelFee**  **Note: roomType is case sensitive**  **blockName is case sensitive** | |

**Note:The attributes of the Hosteller class should be private and methods should be public**

**Example**:

If the collegeFee = 70000, blockName = C, roomType = AC, then the total amount will be:

Total Amount    = collegeFee + blockName + roomType

                                = 70000 + 40000 + 2500

                                = 112500.0

**Component Specification: DayScholar**(Needs to be a child of Student class)

**Note:The attributes of the DayScholar class should be private and methods should be public**

**Example:**

If collegeFee = 60000, distance = 15, then the total amount will be:

Total Amount    = collegeFee + distance

                                = 60000 + 12000

                                = 72000.0

Use a **public class UserInterface** with the **main** method to test the application.  In the main method get the student details as shown in the sample input .

Get the category from the user.

**Note: Category is case sensitive**

If the student is a **Hosteller** then create an object for the Hosteller and invoke the **calculateTotalFee**(), and display the returned amount as output(refer sample output).

If the student is a **DayScholar** then create an object for the DayScholar and invoke the **calculateTotalFee**(), and display the returned amount as output(refer sample output).

**Note:**

* In the Sample Input / Output provided, the highlighted text in bold corresponds to the input given by the user and the rest of the text represents the output.
* Ensure to follow the object oriented specifications provided in the question.
* Ensure to provide the names for classes, attributes and methods as specified in the question.
* Adhere to the code template, if provided.
* Please do not use System.exit(0) to terminate the program.

**Sample Input/Output 1**

Enter Student Id

**45**

Enter Student name

**Arun**

Enter Department name

**ECE**

Enter gender

**Male**

Enter category

**DayScholar**

Enter College fee

**70000**

Enter Bus number

**7**

Enter the distance

**19**

Total College fee is 82000.0

**Sample Input/Output 2**

Enter Student Id

**102**

Enter Student name

**Geetha**

Enter Department name

**CSE**

Enter gender

**Female**

Enter category

**Hosteller**

Enter College fee

**80000**

Enter the room number

**8**

Enter the Block name

**B**

Enter the room type

**AC**

Total College fee is 135000.0

5,

ego is a famous auditing company in the city. During auditing, the details of the employees will be verified, but only for selected employees.  Manually collecting the details will be difficult. So help them to collect the details of the employees and view their details based on their salary.

Help them to write a java program to find the employees whose salary is less than or equal to the salary specified by the auditors using Lambdas.

**Requirement 1:**Find the employee details

Lego company wants to identify the employees whose salary is less than or equal to the salary specified by the auditors. By using the method **"fetchEmployeeDetails"**, identify the employees whose salary is less than or equal to the salary specified by the auditors.

**Component Specification: Employee Interface- This is a Functional Interface.**

|  |  |  |
| --- | --- | --- |
| **Type(Interface)** | **Methods** | **Responsibilities** |
| EmployeeAudit | **public ArrayList<String> fetchEmployeeDetails (double salary)** | This method is used to identify the employees whose salary is less than or equal to the salary passed as an argument by using Lambda expressions. |

**Component Specification: UserInterface Class**

-         Add the employee details using a Map which holds the key as Employee name and value as Employees' salary, where the map is given as a private attribute with the getters and setters as a part of the code template.

-         Then implement the method **public static EmployeeAudit findEmployee()** which holds the employee details using Lambda Expressions.

|  |  |  |  |
| --- | --- | --- | --- |
| **Component Name** | **Type(Class)** | **Methods** | **Responsibilities** |
| Obtain Employee Details | UserInterface | public void **addEmployeeDetails**(String employeeName, double salary) | This method is used to add the employee details into a map. |
| Obtain Employee Details | UserInterface | public static EmployeeAudit **findEmployee**() | This method should return an EmployeeAudit object. To do this, implement the Lambda expression and identify the employees whose salary is less than or equal to the salary passed as an argument. |

If the returned list is empty, then display "**No Employee Found**".

Use appropriate collections to perform the above tasks.

**The signature of the above methods is given as part of the code template, do not change the method signature.**

In the UserInterface class create a main method with the menu as described in the sample Input and Output. When the user selects option **1. Add Employee details**, add the employeeName and salary to the **employeeMap**.

When the user selects option **2. Find Employee details**, it should display the employee name returned by the **findEmployee** method of **UserInterface**class. If no employee is present, then it should display **" No Employee Found".**

When the user selects option **3. Exit**, display the message **"Let's complete the session"** and terminate the program.

**Don't create an object for EmployeeAudit. Use the lambda expression.**

**In the UserInterface class write the main method and perform the given steps :**

-   Get the details of the Employees.

-   Invoke the static method in the **UserInterface** to identify whose salary is less than or equal to the salary specified

-   Capture the object of **EmployeeAudit** returned by the static method.

-   Display the result as shown in the sample output.

**Note**:

* In the Sample Input / Output provided, the highlighted text in bold corresponds to the input given by the user and the rest of the text represents the output.
* Please do not use System.exit(0) to terminate the program.
* Ensure to follow the object oriented specifications provided in the question.
* Ensure to provide the names for classes, attributes and methods as specified in the question.
* Adhere to the code template, if provided.

**Sample Input and Output 1:**

1.Add Employee Details

2.Find Employee Details

3.Exit

 Enter the choice

**1**

Enter the Employee name

**Faruq**

Enter the Employee Salary

**10000**

1.Add Employee Details

2.Find Employee Details

3.Exit

 Enter the choice

**1**

Enter the Employee name

**Benny**

Enter the Employee Salary

**20000**

1.Add Employee Details

2.Find Employee Details

3.Exit

Enter the choice

**2**Enter the salary to be searched

**15000**

Employee List

Faruq

1.Add Employee Details

2.Find Employee Details

3.Exit

Enter the choice

**3**

Let's complete the session

**Sample Input and Output 2:**

1.Add Employee Details

2.Find Employee Details

3.Exit

Enter the choice

**1**

Enter the Employee name

**Vikram**

Enter the Employee Salary

**25000**

1.Add Employee Details

2.Find Employee Details

3.Exit

Enter the choice

**2**

Enter the salary to be searched

**12000**

No Employee Found

1.Add Employee Details

2.Find Employee Details

3.Exit

Enter the choice

**3**

Let's complete the session

6.

Hareesh is fascinated with mathematics. He likes to play with numbers. He is crazy about finding different patterns using numbers. He wants to know about numbers like amicable numbers and numbers that create a  palindrome.

**Amicable numbers:** 220 and 284 are amicable numbers because divisors of 220 are - 1, 2, 4, 5, 10, 11, 20, 22, 44, 55 and 110 where the sum becomes 284 and the divisors of 284 are - 1, 2, 4, 71 and 142 where the sum becomes 220. Thus they make amicable numbers. But the product of 220 and 284 is 62480 which is not a Palindrome.

**Palindrome numbers:** Product of the numbers 122 and 221 produces a Palindrome. But the divisors of 122 are 1,2,61 sum of these numbers does not produce 221 and divisors of 221 are 1,13,17 sum of these numbers does not produce 122 thus they are not amicable numbers.

 Help him to write a java program to find whether the given pair of numbers are amicable numbers and if the product produces a palindrome or not using Lambdas.

**Requirement 1:**Check the Number Category

Hareesh wants to identify whether the number is an amicable number and if the product produces a palindrome. By using the method **checkNumberCategory**, the given numbers are identified whether the numbers are Amicable or if its product  produces a palindrome.

**Component Specification: NumberCategory Interface- This is a Functional Interface.**

|  |  |  |
| --- | --- | --- |
| **Type(Interface)** | **Methods** | **Responsibilities** |
| NumberCategory | **public boolean checkNumberCategory(int number1, int number2)** | This method is used to check whether the numbers passed as an argument are amicable orif it produces a palindrome product by using Lambda expressions. |

**Component Specification: UserInterface Class**

|  |  |  |  |
| --- | --- | --- | --- |
| **Component Name** | **Type(Class)** | **Methods** | **Responsibilities** |
| Check Amicability of the given numbers | UserInterface | **public static NumberCategory checkAmicable()** | This method should return a NumberCategory object. To do this implement the Lambda expression to identify whether the numbers received as a parameter are amicable. |
| Check whether the given numbers are Palindrome | UserInterface | **public static NumberCategory checkPalindrome()** | This method should return a NumberCategory object. To do this implement the Lambda expression to identify whether the product of the numbers received as parameter produces a palindrome. |

**Don't create an object for the NumberCategory. Use the lambda expression.**

**In the UserInterface class write the main method and perform the given steps :**

-  Get the values for the numbers.

-  Invoke the static methods (**checkAmicable()**, **checkPalindrome()**) in the main to perform the calculation.

-  Capture the object of NumberCategory returned by the static method. Using the reference invoke the **checkNumberCategory()** method.

-  Display the result as shown in the sample output.

**Note**:

* In the Sample Input / Output provided, the highlighted text in bold corresponds to the input given by the user and the rest of the text represents the output.
* Ensure to follow the object oriented specifications provided in the question.
* Ensure to provide the names for classes, attributes and methods as specified in the question.
* Adhere to the code template, if provided.
* Do not use System.exit(0) to terminate the program.

**Sample Input 1:**

**220**

**284**

**Sample Output 1:**

220 and 284 are amicable numbers

Their Product 62480 does not produce a Palindrome

**Explanation:**Divisors of 220 are 1+2+4+5+10+11+20+22+44+55+110=284

Divisors of 284 are 1+2+4+71+142=220, thus they make amicable numbers

**Sample Input 2:**

**221**

**122**

**Sample Output 2:**

221 and 122 are not amicable numbers

Their Product 26962 do produces a Palindrome

**Explanation:** Product of 221 and 122 produces 26962, which is a Palindrome.