

```
1 interface IBankNew{
2     boolean applyforCreditCard(Customer customer);
3 }
4
5 interface IBank extends IBankNew{
6     int CAUTION_MONEY = 2000;
7     String createAccount(Customer customer);
8     double issueVehicleLoan(String vehicleType, Customer customer);
9     double issueHouseLoan(Customer customer);
10    double issueGoldLoan(Customer customer);
11 }
12
13 class Customer {
14     private String name;
15     private String customerId;
16
17     public String getName() {
18         return name;
19     }
20
21     public void setName(String name) {
22         this.name=name;
23     }
24     public String getCustomerId() {
25
26         this.name=name;
27     }
28     public String getCustomerId() {
29         return customerId;
30     }
31     public void setCustomerId(String customerId) {
32         this.customerId= customerId;
33     }
34 }
35
36 class MumbaiBranch implements IBank {
37     public String createAccount(Customer customer){
38         return "Acc12345";
39     }
40     public double issueVehicleLoan(String vehicleType, Customer customer){
41         if(vehicleType.equals("bike")) {
42             return 100000;
43         }
44         else {
45             return 500000;
46         }
47     }
48     public double issueHouseLoan(Customer customer){
49         return 200000;
50     }
51 }
```

```

46     }
47     public double issueGoldLoan(Customer customer){
48         return 500000;
49     }
50     public boolean applyforCreditCard(Customer customer){
51         return true;
52     }
53 }
54
55 class Execute{
56     public static void main(String[] args){
57         IBank bank=new MumbaiBranch();
58         Customer customer = new Customer();
59         customer.setCustomerId("cust1001");
60         customer.setName("Ajay");
61         String accountNumber = bank.createAccount(customer);
62         System.out.println("Account number is..." +accountNumber);
63         double gloan = bank.issueGoldLoan(customer);
64         System.out.println("Gold loan amount is..." +gloan);
65         double hloan = bank.issueHouseLoan(customer);
66         System.out.println("House loan amount is..." +hloan);
67         double vloan = bank.issueVehicleLoan("bike", customer);
68         System.out.println("Vehicle loan amount is..." +vloan);
69         System.out.println("Caution money is..." +IBank.CAUTION MONEY);

```

```

IBank bank=new MumbaiBranch();
Customer customer = new Customer();
customer.setCustomerId("cust1001");
customer.setName("Ajay");
String accountNumber = bank.createAccount(customer);
System.out.println("Account number is..." +accountNumber);
double gloan = bank.issueGoldLoan(customer);
System.out.println("Gold loan amount is..." +gloan);
double hloan = bank.issueHouseLoan(customer);
System.out.println("House loan amount is..." +hloan);
double vloan = bank.issueVehicleLoan("bike", customer);
System.out.println("Vehicle loan amount is..." +vloan);
System.out.println("Caution money is..." +IBank.CAUTION MONEY);
IBankNew bank1 = new MumbaiBranch();
boolean credit = bank1.applyforCreditCard(customer);
System.out.println("Credit card request.." + credit);
}
}

```

## Execution Result

### Output:

Account number is...Acc12345  
Gold loan amount is...500000.0  
House loan amount is...200000.0  
Vehicle loan amount is...100000.0  
Caution money is...2000  
Credit card request..true

```
1 interface IBank {
2     int CAUTION_MONEY = 2000;
3     String createAccount(Customer customer);
4     double issueVehicleLoan(String vehicleType, Customer customer);
5     double issueHouseLoan(Customer customer);
6     double issueGoldLoan(Customer customer);
7 }
8 class Customer {
9     private String name;
10    private String customerId;
11
12    public String getName() {
13        return name;
14    }
15
16    public void setName(String name) {
17        this.name=name;
18    }
19    public String getCustomerId() {
20        return customerId;
21    }
22    public void setCustomerId(String customerId) {
23        this.customerId= customerId;
24    }
25 }
26 class MumbaiBranch implements IBank {
27     public String createAccount(Customer customer){
28         return "Acc12345";
29     }
30     public double issueVehicleLoan(String vehicleType, Customer customer){
```

```

28         return "Acc12345";
29     }
30     public double issueVehicleLoan(String vehicleType, Customer customer){
31         if(vehicleType.equals("bike")) {
32             return 100000;
33         }
34         else {
35             return 500000;
36         }
37     }
38     public double issueHouseLoan(Customer customer){
39         return 200000;
40     }
41     public double issueGoldLoan(Customer customer){
42         return 500000;
43     }
44 }
45
46 class Execute{
47     public static void main(String[] args){
48         IBank bank=new MumbaiBranch();
49         Customer customer = new Customer();
50         customer.setCustomerId("cust1001");
51         customer.setName("Ajay");
52         String accountNumber = bank.createAccount(customer);
53         System.out.println("Account number is..." +accountNumber);
54         double gloan = bank.issueGoldLoan(customer);
55         System.out.println("Gold loan amount is..." +gloan);
56         double hloan = bank.issueHouseLoan(customer);
57         System.out.println("House loan amount is..." +hloan);

```

```

33     }
34     else {
35         return 500000;
36     }
37 }
38 public double issueHouseLoan(Customer customer){
39     return 200000;
40 }
41 public double issueGoldLoan(Customer customer){
42     return 500000;
43 }
44 }
45
46 class Execute{
47     public static void main(String[] args){
48         IBank bank=new MumbaiBranch();
49         Customer customer = new Customer();
50         customer.setCustomerId("cust1001");
51         customer.setName("Ajay");
52         String accountNumber = bank.createAccount(customer);
53         System.out.println("Account number is..." +accountNumber);
54         double gloan = bank.issueGoldLoan(customer);
55         System.out.println("Gold loan amount is..." +gloan);
56         double hloan = bank.issueHouseLoan(customer);
57         System.out.println("House loan amount is..." +hloan);
58         double vloan = bank.issueVehicleLoan("bike", customer);
59         System.out.println("Vehicle loan amount is..." +vloan);
60         System.out.println("Caution money is..." +IBank.CAUTION_MONEY);
61     }
62 }

```



[Reset](#)[Execute](#)[Copy Code](#)

## Execution Result

### Output:

```
Account number is...Acc12345
Gold loan amount is...500000.0
House loan amount is...200000.0
Vehicle loan amount is...100000.0
Caution money is...2000
```

```
1  class Person{
2      private int salary = 5000;
3      public String name = "Jack";
4      protected int age = 24;
5      String email = "jack@samurai.com";
6
7      public void display(){
8          System.out.println("Name: " + name);
9          System.out.println("Age: " + age);
10         System.out.println("Email: " + email);
11         System.out.println("Salary: " + salary);
12     }
13 }
14
15 class Employee extends Person {
16     public void display(){
17         System.out.println("Name: " + name);
18         System.out.println("Age: " + age);
19         System.out.println("Email: " + email);
20     }
21 }
22
23 class Customer {
24     public void display(){
25         Person p = new Person();
26         System.out.println("Name: " + p.name);
27         System.out.println("Age: " + p.age);
28         System.out.println("Email: " + p.email);
29     }
30 }
```

```
28         System.out.println("Email: " + p.email);
29     }
30 }
31
32 class Execute{
33     public static void main (String[] args) {
34         Person p = new Person();
35         Employee e = new Employee();
36         Customer c = new Customer();
37         System.out.println("*****");
38         System.out.println("Person Class display method.");
39         System.out.println("*****");
40         p.display();
41         System.out.println("*****");
42         System.out.println("Employee Class display method.");
43         System.out.println("*****");
44         e.display();
45         System.out.println("*****");
46         System.out.println("Customer Class display method.");
47         System.out.println("*****");
48         c.display();
49     }
50 }
```

## Execution Result

Output:

\*\*\*\*\*

Person Class display method.

\*\*\*\*\*

Name: Jack

Age: 24

Email: jack@samurai.com

Salary: 5000

\*\*\*\*\*

Employee Class display method.

\*\*\*\*\*

Name: Jack

Age: 24

Email: jack@samurai.com

\*\*\*\*\*

Customer Class display method.

\*\*\*\*\*

Name: Jack

Age: 24

Email: jack@samurai.com

## Code in Java

```
1  class WrapperClassTester {
2
3      public static void main(String[] args) {
4
5          int i = 45;//primitive data int
6          Integer integer = new Integer(i);// Integer wrapper class instantiation
7          int i2 = integer.intValue();// unwrapping primitive data int from wrapper object
8          Integer integer2 = new Integer("23");
9
10         // all wrapper class except Character can take String in argument
11         System.out.println(integer2);
12         Integer intObj1 = new Integer(25);
13         Integer intObj2 = new Integer("25");
14         Integer intObj3 = new Integer(35);
15
16         //compareTo demo
17         System.out.println("Comparing using compareTo obj1 and obj2: " + intObj1.compareTo(intObj2));
18         System.out.println("Comparing using compareTo obj1 and obj3: " + intObj1.compareTo(intObj3));
19
20         // Equals demo
21         System.out.println("Comparing using compareTo obj1 and obj2: " + intObj1.equals(intObj2));
22         System.out.println("Comparing using compareTo obj1 and obj3: " + intObj1.equals(intObj3));
23         Float f1 = new Float("2.25f");
24         Float f2 = new Float("20.43f");
25         Float f3 = new Float(2.25f);
26         System.out.println("Comparing using compare f1 and f2: " + Float.compare(f1,f2));
27         System.out.println("Comparing using compare f1 and f3: " + Float.compare(f1,f3));
28
29         // Addition of Integer with Float
30         Float f = intObj1.floatValue() + f1;
31
32         System.out.println("Comparing using compareTo obj1 and obj2: " + intObj1.compareTo(intObj2));
33         System.out.println("Comparing using compareTo obj1 and obj3: " + intObj1.compareTo(intObj3));
34
35         // Equals demo
36         System.out.println("Comparing using compareTo obj1 and obj2: " + intObj1.equals(intObj2));
37         System.out.println("Comparing using compareTo obj1 and obj3: " + intObj1.equals(intObj3));
38         Float f1 = new Float("2.25f");
39         Float f2 = new Float("20.43f");
40         Float f3 = new Float(2.25f);
41         System.out.println("Comparing using compare f1 and f2: " + Float.compare(f1,f2));
42         System.out.println("Comparing using compare f1 and f3: " + Float.compare(f1,f3));
43
44         // Addition of Integer with Float
45         Float f = intObj1.floatValue() + f1;
46         System.out.println("Addition of intObj1 and f1: "+ intObj1 + "+" + f1 + "=" + f);
47         int x = Integer.parseInt("34");
48         System.out.println(x);
49         double y = Double.parseDouble("34.7");
50         System.out.println(y);
51     }
52 }
```



## Execution Result

### Output:

23

Comparing using compareTo obj1 and obj2: 0

Comparing using compareTo obj1 and obj3: -1

Comparing using compareTo obj1 and obj2: true

Comparing using compareTo obj1 and obj3: false

Comparing using compare f1 and f2: -1

Comparing using compare f1 and f3: 0

Addition of intObj1 and f1: 25+2.25=27.25

34

34.7

```
1      class Bank{
2          public static void main(String[] args){
3              String username = "Tendulkar";
4              int size = username.length();
5              if(size > 8 && size <15){
6                  char arr[]=username.toCharArray();
7                  int count=0;
8                  for(char c:arr){
9                      if(Character.isLetter(c)){
10                         ++count;
11                     }
12                 }
13                 if(count == size){
14                     System.out.println("valid username");
15                 }
16             }
17         }
18     }
```

## Execution Result

Output:

valid username

## Code in Java

```
1
2 class StringBuilderDemo{
3
4     public static void main(String[] args){
5
6         String firstName="Sachin";
7         String lastName="Tendulkar";
8         String fullName=firstName+lastName;
9         //'+'operator concatenates the string but creates a new object in the heap memory as sting is immutable
10        System.out.println(fullName);
11        StringBuilder sb=new StringBuilder(firstName);
12        String fName=sb.append(lastName).toString();//toString method converts StringBuilder to String
13        //StringBuilder is mutable, it appends to a single object
14        System.out.println(fName);
15    }
16 }
17 }
```

## Execution Result

Output:

SachinTendulkar  
SachinTendulkar

## Code in Java

```
1 class Except {  
2     public static void divide(int x, int y) {  
3         int z = x / y;  
4         System.out.println(z);  
5     }  
6  
7     public static void main(String[] args) {  
8         divide(10, 0);  
9     }  
10 }  
11
```

## Execution Result

### Runtime Exception

Exception in thread "main" java.lang.ArithmeticException: / by zero  
at Except.divide(myCode.java:3)  
at Except.main(myCode.java:8)

```

1  class ExceptionDemo {
2
3      public static int divide(int a,int b) {
4          return a/b;
5      }
6
7      public static void main(String[] args) {
8          try {
9              divide(9,0);
10         } catch (ArithmeticException exception) {
11             System.out.println(exception);
12             //exception.printStackTrace();
13             //System.out.println(exception.getMessage());
14             //System.out.println(exception.toString());
15         }
16         finally {
17             System.out.println("Inside finally");
18         }
19     }
20 }

```

## Execution Result

Output:

```

java.lang.ArithmeticException: / by zero
Inside finally

```

```

1 class UserInterface {
2     public static void divide(int x, int y) {
3         try {
4             if (y == 0)
5                 throw new Exception("The divisor should not be zero");
6             int z = x / y;
7             System.out.println(z);
8         } catch (Exception e) {
9             System.out.println(e.getMessage());
10        }
11    }
12
13    public static void main(String[] args) {
14        UserInterface.divide(10, 0);
15    }
16 }
17

```

## Execution Result

Output:

The divisor should not be zero

## Code in Java

```

1 class MyDivException extends Exception
2 {
3     public MyDivException(String message) {
4         super(message);
5     }
6 }
7
8 class Tester
9 {
10    public static void divide(int x, int y) throws MyDivException {
11        if(y == 0)
12            throw new MyDivException("The divisor should not be zero");
13        int z = x/y;
14        System.out.println(z);
15    }
16
17    public static void main(String[] args)
18    {
19        try
20        {
21            divide(6,0);
22        } catch(MyDivException e) {
23            System.out.println(e.getMessage());
24        }
25    }
26 }
27

```



## Execution Result

Output:

The divisor should not be zero

```
1 class Record<E> {
2     private E record;
3     public void display(E item) {
4         System.out.println(item);
5     }
6 }
7
8 class Student {
9     private int studentId;
10    private String studentName;
11
12    public Student(int studentId,String studentName)
13    {
14        this.studentId=studentId;
15        this.studentName=studentName;
16    }
17    public String toString()
18    {
19        return "Student: Id = " + studentId + " Name = " + studentName;
20    }
21 }
22
23 class GenericsDemo {
24     public static void main(String[] args)
25     {
26         Student s1 = new Student(101,"Robert");
27         Record<Integer> integerRecord = new Record<Integer>(); //integerRecord can be used to display only integers
28         integerRecord.display(12);
29         //integerRecord.display(s1); will give an error as we are trying to add a student class object
30         Record<Student> studentRecord = new Record<>(); //studentRecord can be used to display only Students
31         return "Student: Id = " + studentId + " Name = " + studentName;
32     }
33 }
34
35 class GenericsDemo {
36     public static void main(String[] args)
37     {
38         Student s1 = new Student(101,"Robert");
39         Record<Integer> integerRecord = new Record<Integer>(); //integerRecord can be used to display only integers
40         integerRecord.display(12);
41         //integerRecord.display(s1); will give an error as we are trying to add a student class object
42         Record<Student> studentRecord = new Record<>(); //studentRecord can be used to display only Students
43         studentRecord.display(s1);
44         //studentRecord.display(15); will give an error as we are trying to add an integer
45     }
46 }
```

## Execution Result

Output:

12

Student: Id = 101 Name = Robert